SCRATCH İLE PROGRAMLAMA ÖĞRETİMİNİN BİLİŞİM TEKNOLOJİLERİ ÖĞRETMEN ADAYLARININ MOTİVASYON VE BAŞARILARINA ETKİSİ

Osman EROL

(Doktora Tezi)

Mayıs 2015

SCRATCH İLE ROGRAMLAMA ÖĞRETİMİNİN BİLİŞİM TEKNOLOJİLERİ ÖĞRETMEN ADAYLARININ MOTİVASYON VE BAŞARILARINA ETKİSİ

Osman EROL

DOKTORA TEZİ

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı

Danışman: Doç. Dr. Adile Aşkım KURT

Eskişehir
Anadolu Üniversitesi Eğitim Bilimleri Enstitüsü
Mayıs 2015

JÜRİ VE ENSTİTÜ ONAYI

Osman EROL'un "SCRATCH ile Programlama Öğretiminin Bilişim Teknolojileri Öğretmen Adaylarının Motivasyon ve Başarılarına Etkisi" başlıklı tezi 25.05.2015 tarihinde, aşağıda belirtilen jüri üyeleri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı Bilgisayar ve Öğretim Teknolojileri Öğretmenliği Programında, Doktora tezi olarak değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	**İmza**
Üye (Tez Danışmanı)	: Doç.Dr. Adile Aşkım KURT	
Üye	: Prof.Dr. H.Ferhan ODABAŞI	
Üye	: Prof.Dr. Soner YILDIRIM	
Üye	: Doç.Dr. Yavuz AKBULUT	
Üye	: Doç.Dr. Serkan ŞENDAĞ	

Doç.Dr. Handan DEVECİ
Anadolu Üniversitesi
Eğitim Bilimleri Enstitüsü Müdür Vekili

ÖZET

SCRATCH İLE PROGRAMLAMA ÖĞRETİMİNİN BİLİŞİM TEKNOLOJİLERİ ÖĞRETMEN ADAYLARININ MOTİVASYON VE BAŞARILARINA ETKİSİ

Osman EROL
Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı
Anadolu Üniversitesi Eğitim Bilimleri Enstitüsü
Mayıs 2015

Danışman: Doç. Dr. Adile Aşkım KURT

Bu araştırmanın amacı Scratch ile programlama öğretiminin öğrencilerin motivasyon ve programlama başarısına etkisini incelemektir. Araştırmanın çalışma grubunu Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri bölümünde eğitim gören 52 ikinci sınıf öğrencisi oluşturmaktadır. Katılımcılar 26 öğrenci deney ve 26 öğrenci kontrol grubunda olacak şekilde yansız atama yapılarak iki farklı gruba ayrılmışlardır. Uygulamanın ilk yedi haftalık bölümünde programlama mantığının kazandırılması ve temel programlama yapılarının öğretilmesi amaçlanmıştır. Bu amaçla deney grubundaki katılımcılar Scratch ile oyun tasarımı etkinlikleri, kontrol grubunda yer alan katılımcılar ise mevcut ders programındaki haliyle akış diyagramları ile problem çözme etkinlikleri yapmışlardır. Uygulamanın ikinci yedi haftalık bölümde ise hem kontrol hem de deney grubunda aynı yöntem kullanılarak C# programlama dili öğretimi gerçekleştirilmiştir. Araştırmada veri toplama araçları olarak Başarı Testi, Güdülenme ve Öğrenme Stratejileri Ölçeği ve Odak Grup Görüşme Formu kullanılmıştır. Araştırmada 3 (ölçme zamanı)*2 (grup) faktöriyel desen kullanılmıştır. Hem deney grubunda hem de kontrol grubunda Başarı Testi ile Güdülenme ve Öğrenme Stratejileri ölçeği, öntest, sontest ve sontest 2 olmak üzere üç defa uygulanmıştır. Ayrıca hem deney hem de kontrol grubundaki katılımcılar ile ilk yedi haftalık uygulamanın sonunda ve C# programlama öğretiminin yapıldığı ikinci yedi haftalık uygulama sonunda odak grup görüşmeleri yapılmıştır. Araştırmada elde edilen verilerin analizinde çok değişkenli varyans analizi (MANOVA), tek

değişkenli varyans analizi (ANOVA) , bağımsız örneklem t-testi, bağımlı örneklem t-testi ve içerik analizi kullanılmıştır.

Araştırmada elde edilen bulgulara göre; katılımcıların hem motivasyon hem de başarı puanları; ölçüm zamanına göre, grup değişkenine göre (öğretim yönteminin türüne göre), ölçüm zamanı ve grup değişkeninin etkileşimine göre anlamlı farklılık göstermiştir. Buna göre yapılan basit etki analizi sonucunda katılımcıların motivasyon puanlarının öntestte her iki grupta benzer olduğu, sontest ve sontest 2'de ise deney grubu lehine anlamlı bir farklılık ortaya çıkmıştır. Ayrıca ölçüm zamanı bağlamında kontrol grubunda katılımcıların motivasyon puanlarının tüm uygulama sonunda azaldığı, deney grubunda ise arttığı görülmüştür. Katılımcıların programlama başarı puanları ele alındığında öntestte her iki grupta da benzer olduğu, sontest ve sontest 2' de ise deney grubu lehine anlamlı bir farklılık olduğu ortaya çıkmıştır. Ölçüm zamanı bağlamında ise hem kontrol hem de deney grubundaki katılımcıların programlama başarı puanlarının tüm uygulama sonunda arttığı görülmüştür. Ayrıca grup değişkeni bağlamında artışın deney grubu lehine anlamlı olarak farklı olduğu görülmüştür. Araştırmanın nitel verileri incelendiğinde deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinin eğlenceli ve kolay olduğunu, ders süresince yapılan etkinliklerin programlama mantığını kazandırmada ve motivasyonu artırmada etkili olduğunu, ancak bazı temel yapılar için yetersiz olduğunu dile getirmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme sürecinin zor ve sıkıcı olduğunu, ders süresince yapılan etkinliklerde ise aktif olamama ve uygulamanın olmaması gibi sınırlılıkların motivasyonlarını düşürdüğünü belirtmişlerdir. Ayrıca deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı sürecinde edindikleri deneyimin C# programlama öğrenme sürecinde programlama başarılarına katkısının olduğunu belirtmişlerdir. Araştırmanın sonunda uygulamaya ve araştırmalara yönelik önerilerde bulunulmuştur.

Anahtar Sözcükler: Programlama, oyun tasarımı, Scratch, akış diyagramları, motivasyon

ABSTRACT

THE EFFECTS OF TEACHING PROGRAMMING WITH SCRATCH ON PRE-SERVICE INFORMATION TECHNOLOGY TEACHERS' MOTIVATION AND ACHIEVEMENT

Osman EROL

Department of Computer Education and Instructional Technology
The Graduate School of Educational Sciences
May 2015

Advisor: Assoc. Prof. Dr. Adile Aşkım KURT

The purpose of this study was to investigate the effects of teaching programming with Scratch on students' motivation and programming achievement. Working group of the research consisted of 52 second graders studying in Mehmet Akif Ersoy University, Faculty of Education, Department of Computer Education and Instructional Technology. Participants were randomly assigned to two groups. There were 26 students in the experimental group and 26 students in the control group. In the first seven weeks of the application, gaining the programming logic and learning the basic programming structures were aimed. To this end, participants in the experimental group made game design activities with Scratch, while participants in the control group made problem-solving activities through the form of flow diagrams in the current curriculum. In the second seven-week section of the application, teaching C# programming language was carried out in both the control and the experimental groups through using the same teaching method. Data collection instruments used in the study were Achievement Test, Motivation and Learning Strategies Scale, and Focus Group Interview form. In the research design, 3(measuring time)* 2(groups) factorial design was used. In both the control group and the experimental group, achievement test and Motivation and Learning Strategies Scale were administered to the participants three times as pre-test, final test and final test 2. In addition, the focus group interviews were conducted with the participants in both control and experimental group at the end of the

first seven-week application and at the end of the second seven-week application that involved teaching C# programming. Data were analyzed through multivariate analysis of variance (MANOVA), univariate analysis of variance (ANOVA), independent sample t-test, dependent sample t-test and content analysis for qualitative data.

According to the findings obtained in the study, there were significant differences in participants' motivation and success scores with regard to measurement time, group variable (depends on type of the teaching method), and both measurement time and group interaction. According to the results of simple effect analysis, it was found that motivation scores of the participants were similar in both groups at the pre-test. However, in final test and final test 2, there was a meaningful difference in favor of the experimental group. In addition, in terms of the measurement time, motivation scores of the control group decreased at the end of all applications, while it increased in the experimental group. In terms of the programming achievement scores of the participants, the preliminary test was similar in both groups. In the final test and final test 2, a meaningful difference was observed in favor of the experimental group. In addition, in terms of the measurement time, programming achievement scores of participants of both control and experimental groups have increased at the end of the whole application. However, in terms of the increase in the grouping variable, a meaningful difference in favor of the experimental group can be observed.

Qualitative data revealed that participants in the experimental group found the game design with Scratch fun and easy. They considered the activities carried out during the course as effective in terms of gaining programming logic and increasing the motivation, but some basic structures were considered insufficient. Participants in the control group reported that flow diagrams and problem-solving processes were difficult and boring, and activities carried out during the course had disadvantages which reduced their motivation such as the inability to be active and to make practice. Participants in the experimental group further stated that the experience gained in the game design process with Scratch contributed to their programming success in learning process of C# programming. At the end of the study, recommendations were made for practice and further research.

Keywords: Programming, game design, Scratch, flow diagrams, motivation

ÖNSÖZ

Lisans eğitimim boyunca programlama dersleri benim için her zaman zor ve sıkıcı olmuştur. Yaklaşık yedi yıldır farklı isimlerdeki programlama derslerini yürütmekteyim. Ancak programlama derslerinde hiçbir zaman öğrencilerimin mutlu olduğuna şahit olamadım ve onlara programlamayı sevdirmekte hep başarısız oldum. Hem öğrencilerimin bu haklı isyanı hem de kendi deneyimlerim beni programlamada yeni yöntemler denemeye zorladı. Yaşadığım deneyimler bu çalışma için bir zemin hazırlamış oldu. Bu çalışma benim için yalnızca akademik bir çalışma olmadı. Aynı zamanda çok da güzel yeni bir deneyim oldu. Bu çalışma ile beş yıl önce başladığım, hayatımın önemli bir dönemi olan ve bana birçok noktada katkısı olan doktora eğitimim son bulmuş oldu.

Doktora eğitimim boyunca gerek aldığım derslerde, gerekse tez sürecinin her aşamasında yardımlarını esirgemeyen, bilgi ve deneyimlerini paylaşan danışmanım, saygıdeğer hocam Doç. Dr. Adile Aşkım KURT' a sonsuz teşekkürlerimi sunarım.

Tez izleme komitesinde yer alan ve değerleri görüşleriyle araştırmamda birçok katkısı olan, kendilerinden hem akademik anlamda hem de hayata bakış açısı bağlamında birçok şey öğrendiğim, desteklerini hiçbir zaman esirgemeyen çok değerli hocalarım Doç. Dr. Serkan ŞENDAĞ ve Doç. Dr. Yavuz AKBULUT' a teşekkürü bir borç bilirim. Tez savunma jürimde yer alan ve katkıları ile bana yol gösteren sayın hocalarım Prof. Dr. Ferhan ODABAŞI ve Prof. Dr. Soner YILDIRIM' a sonsuz teşekkürlerimi sunarım.

Doktora eğitimim süresince kendilerinden çok şey öğrendiğim Doç. Dr. Abdullah KUZU, Doç. Dr. Işıl KABAKÇI YURDAKUL, Yrd. Doç. Dr. Yusuf Levent ŞAHİN ve Yrd. Doç. Dr. Sema ÜNLÜER hocalarıma teşekkür ederim.

Eskişehir' e geldiğimde bana kapılarını açan ve misafir eden arkadaşım Muhterem DİNDAR' a, bu zorlu süreç içinde desteğini her daim hissettiğim değerli arkadaşım Eray YILMAZ' a ve güler yüzlü Anadolu Üniversitesi BÖTE ailesine teşekkür ederim.

Ayrıca sekiz yıldır çalıştığım ve bir aile ortamından farkı olmayan Mehmet Akif Ersoy Üniversitesi BÖTE bölümünde başta Yrd. Doç. Dr. Vesile Gül BAŞER GÜLSOY olmak üzere tüm değerli iş arkadaşlarıma teşekkür ederim.

Eğitim hayatım boyunca kendi sıkıntılarını bir kenara bırakıp bugünlere ulaşmamda maddi manevi desteğini hiç esirgemeyen anneme, babama ve kardeşlerime şükranlarımı sunarım. Tüm lisansüstü eğitimim boyunca tüm zorlukları ve güzellikleri birlikte paylaştığım, manevi desteğini sürekli hissettiğim, bana destek olan ve bu zorlu süreçte sabreden hayat arkadaşım, sevgili eşim Nurcan KUNT EROL' a ve varlığı ile bana mutluluk veren biricik kızım Zeynep Bilge' ye sonsuz teşekkür ederim.

Son olarak doktora eğitim boyunca beni destekleyen TUBİTAK Bilim İnsanı Destekleme Daire Başkanlığı'na teşekkür ederim

Osman Erol

Mayıs 2015

İÇİNDEKİLER

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KISALTMALAR LİSTESİ

AGFI	: Düzenlenmiş İyilik Uyum İndeksi (Adjusted Goodness of Fit Index)
BİT	: Bilgi ve İletişim Teknolojileri
CFI	: Karşılaştırmalı Uyum İndeksi (Comparative Fit Index)
GFI	: İyilik Uyum İndeksi (Goodness of Fit Index)
LOGO	: Language of Graphical Output
MIT	: Massachusetts Teknoloji Enstitüsü (Massachusetts Institute of Technology)
NFI	: Normlaştırılmış Uyum İndeksi (Normed Fit Index)
NNFI	: Normlaştırılmamış Uyum İndeksi (Non -normed Fit Index)
RMSEA	: Yaklaşık Hataların Ortalama Karekökü (Root Meansquare Error of Approximation)
Sd	: Serbestlik Derecesi
SRMR	: Standardize Edilmiş Artık Ortalamaların Karekökü (Standardized Root Mean Square Residuals)
YÖK	: Yükseköğretim Kurulu

BİRİNCİ BÖLÜM
GİRİŞ

Bilgi ve iletişim teknolojileri sektörü sürekli gelişen ve küresel düzeyde pazarı büyüyen dinamik alanlardan biridir. Özellikle de yazılım sektörünün bu pazar içindeki payı oldukça büyüktür. Bu nedenle yazılım geliştirebilen ve programlama becerisine sahip insan gücüne oldukça gereksinim duyulmaktadır. Ancak son yıllarda bilgisayar bilimleri alanında eğitim gören öğrencilerin sayısında önemli azalış görülmektedir (Başer, 2013; Heersink ve Moskal, 2010). Bunun önemli nedenlerinden biri birçok öğrencinin programlama dilleri dersini zor ve sıkıcı bir ders olarak görmeleri ve bu dersten başarısız olmalarıdır (Bennedsen ve Caspersen, 2008; Monroy- Hernandez ve Resnick, 2010; Robins, Rountree ve Rountree, 2003). Bazı araştırmacılara göre öğrencilerde bu olumsuz algıya ve başarısızlığa neden olan etmen programlama öğretiminde yaşanan sorunlardır (Deek ve Espinosa, 2005; Schulte ve Bennedsen, 2006; Soloway, 1986). Özellikle programlamaya yeni başlayanlar için temel ders sayılan "Programlamaya Giriş", "Programlama Dillerinin Temelleri" veya "Programlama Dilleri I" gibi farklı isimlerdeki programlama derslerinde amaç öğrencilere programlamaya ilişkin temel kavramların ve programlama mantığının öğretilmesidir (Özdener, 2008). Ancak birçok eğitimci programlama mantığını öğretmek yerine bunu bir programlama diliyle birlikte öğretmeyi tercih etmektedir (Deek ve Espinosa, 2005; Schulte ve Bennedsen, 2006). Bu durumda henüz başlangıç düzeyinde olan programcılar, hem bir programlama diline ait bütün kodları ve kuralları hem de programlamaya ilişkin kavramları (döngü, dizi, fonksiyon vb.) öğrenmek zorunda kalmaktadır. Bu nedenle öğrenenler bütün kodları ve kuralları bilseler bile programlama mantığını kavrayamadıkları için, bu kodları bir araya getirmekte ve herhangi bir programlama problemini çözmekte sorun yaşamaktadır (Cooper, Dann ve Pausch, 2003; Garner, Haden ve Robins, 2005; Porter ve Calder, 2004).

Söz konusu bu karmaşıklık programlama öğrenmeye bir engel oluşturmakta ve öğrencilerin motivasyonunu düşürmektedir (Allan ve Kolsar, 1997; Özoran, Çağıltay ve Topallı, 2012). Dolayısıyla öğrenciler programlama mantığını düşük düzeyde kavramakta ve başka bir programlama diline bunu aktaramamaktadır (Kinnunen ve Malmi, 2008; Kurland, Pea, Clemaent ve Mawby, 1989). Oysaki özellikle başlangıç

düzeyindeki programcıların eğitiminde belli bir programlama dilini öğretmek yerine programlama mantığını ve programlamaya ilişkin temel kavramları (döngü, fonksiyon, değişken, dizi vb.) öğretmenin daha yararlı olduğu görülmektedir (Bennedsen ve Caspersen, 2008; Linn ve Dalbey, 1985; Winslow, 1996). Bunun nedeni programlamanın temelinde programlama mantığının yatmasıdır. Aslında programlama dillerine ait kodlar ve komutlar birbirleriyle farklılık gösterse bile programlama mantığı tümünde benzerdir. Bu nedenle bir programlama diline geçmeden önce programlama mantığının çok iyi kavranmış olması önemlidir (Futschek, 2006).

Programlama mantığı algoritma geliştirerek öğrenilmektedir (Bennedsen ve Caspersen, 2008). Algoritma, teknik anlamda bir programlama probleminin çözümüne ilişkin işlemlerin adım adım bilgisayara iletilmesi (Vatansever, 2011), genel anlamda ise bir sorunun giderilmesi için işlemlerin uygulanması veya sonuca en hızlı şekilde ulaşılması biçimidir (Özkan, 2003). Bu nedenle programcı, programlamaya ilişkin bir problemi çözmeden önce problemin algoritmasını modellemeli ve çözüm basamaklarını belirlemelidir. Genellikle bu süreçte sözde kod denilen günlük konuşma dili ve akış diyagramı kullanılmaktadır. Ancak birçok öğrenci bu algoritmik yapıyı kavramada ve problemi algoritmaya dönüştürmede sorun yaşamaktadır (Winslow, 1996). Öğrenciler açısından bir programlama problemini basamaklara ayırmak ve modellemek, ayrıca tüm bu süreçlerde programlamaya ilişkin kavramları kullanmak soyut bir işlemdir. Bir başka deyişle öğrenciler bu süreci somutlaştırmada zorlanmaktadırlar (Cooper ve diğerleri, 2003; Futschek, 2006; Soloway ve Spohrer, 1989). Bu durum, yani sürekli soyut ifadelerin, teknik kavramların ve metin tabanlı ifadelerin yer alması öğrencilerin programlama sürecinden sıkılmasına yol açmaktadır (Ginat, 2003; Lahtinen, Ala-Mutka ve Järvinen, 2005; Mayer, 1981).

Algoritma öğretiminin bu denli soyut bir süreç olması ileride öğrencilerin programlama öğrenme sürecinde sıkıntılar yaşamasına neden olmaktadır. Bu nedenle algoritma öğretimini daha somut ve anlaşılır hale getirmek için eğitimciler Raptor, Flint, FCPro gibi akış diyagramı oluşturma ve modelleme araçları kullanmakta, ayrıca algoritmayı tasarlayarak hikayeleştiren veya görselleştiren Scratch ve Alice gibi araçlar da kullanmaktadırlar (Baldwin ve Kuljis, 2000; Cooper, Powers, McNally, Goldman, Proulx, Carlisle, 2006; Cooper ve diğerleri, 2003; Maloney, Resnick, Rusk, Silverman ve Eastmond, 2010).

Tüm bu yazılımların genel özelliği algoritma sürecini somutlaştırması, hızlı test etme olanağı ve kullanıcıya düzeltme olanağı sunmasıdır. Böylelikle programlama kavramları ve mantığı daha iyi anlaşılır hale gelmekte, problem çözme süreci kolaylaşmaktadır (Cooper ve diğerleri, 2006). Raptor, Flint, FCPro gibi akış diyagramı hazırlama yazılımları, öğrencilerin problem basamakları arasındaki bağlantıları görmelerine, hızlı test etmelerine ve yazılım hatalarını en aza indirmelerine olanak vermektedir (Baldwin ve Kuljis, 2000; Cooper ve diğerleri, 2003). Alice ve Scratch gibi yazılımlar ise tüm bu özelliklerinin yanı sıra oyun ve hikâye tasarlama olanakları sunarak algoritma sürecini daha anlaşılır hale getirmekte ve problem çözme sürecini somut bir ürüne dönüştürmektedir (Ramadhan, 2000; Utting, Cooper, Kölling, Maloney ve Resnick, 2010). Her ne kadar akış diyagramı hazırlama yazılımları programlama mantığını geliştirmek için etkili olsa da tasarım yapabilme olanağı sağlamaması önemli bir sınırlılıktır (Cooper ve diğerleri, 2003). Alice ya da Scratch gibi oyun ve hikâye tasarlama olanağı sağlayan araçlar, öğrencilerin daha somut ve özgün içerikler oluşturmalarına olanak tanımakta ve bu süreci daha ilgi çekici hale getirmektedir (Utting ve diğerleri, 2010). Özelikle oyun tasarımı süreci öğrenenlerin programlama mantığını kavramlarında yardımcı olabilmektedir (Cooper ve diğerleri, 2003; Moreno, 2012). Bunun nedeni ise öğrencinin tasarım sırasında daha etkin hale gelmesi, zihinsel olarak görüş ve kavramlar ile yakından ilgilenmesi, ürünle birlikte etkileşimde bulunması dolayısıyla daha çok öğrenmeye çalışmasıdır (Ke, 2014; Li, 2012; Mishra ve Girod, 2006; Resnick, 2007). Tasarım odaklı görselleştiriciler ele alındığında algoritma ve programlama öğretiminde en sık kullanılan araçlardan birinin Scratch olduğu söylenebilir (Resnick ve diğerleri, 2009).

Scratch; iki milyonu aşkın kullanıcı sayısı ve 50 farklı dil desteği ile kütüphanesinde binlerce projeyi barındıran ve diğer görselleştiricilere göre kullanımı daha kolay ve anlaşılır bir yazılımdır (Maloney ve diğerleri, 2010). Basit arayüzü ve sürükle bırak yapısı sayesinde Scratch tüm dünyada ilkokuldan üniversite öğrencilerine kadar farklı yaşlardaki öğrencilere algoritma mantığını öğretmek ve programlamayı sevdirmek için kullanılmaktadır. Aslında Scratch bir programlama dili olmaktan çok programlama mantığını öğreten ve genellikle programlamaya yeni başlayanların tercih ettiği etkili bir araçtır. Birçok çoklu ortam öğesini birlikte kullanmaya olanak tanıyan Scratch tasarım odaklı yapısı sayesinde oyun, hikâye ve animasyon tasarımına olanak

sağlamakta bu da öğrenmeyi daha eğlenceli hale getirmektedir (Genç ve Karakuş, 2011).

Algoritma ve programlama öğretiminde yaşanan sorunlar öğrencilerin programlama dersindeki motivasyonlarını düşürebilmekte, programlama eğitiminin zor bir süreç olduğuna ilişkin bir algı oluşturarak öğrencilerin programlamaya karşı olumsuz tutum geliştirmelerine neden olmaktadır (Anastasiadou ve Karakos, 2011). Dolayısıyla öğrencilerin motivasyonunun düşmesi ve oluşan olumsuz tutum öğrencilerin programlamaya ilişkin başarılarını olumsuz etkilemektedir (Lahtinen, ve diğerleri, 2005). Tüm bunlar dikkate alındığında Scratch basit arayüzü, kod gerektirmeyen yapısı, oyun tasarımına olanak tanıması ve her yaş için uygun olması özelliği ile programlama öğretiminde yaşanan zorlukları ve olumsuzlukları ortadan kaldırmak için etkili bir araç olarak kullanılabilir. Ayrıca Scratch ile programlama öğretiminde; programlamanın karmaşık yapısından kaynaklı öğrencilerde oluşan olumsuz tutum ortadan kaldırılabilir, öğrencilerin programlama derslerindeki motivasyonları artırılarak başarılı olmaları sağlanabilir. Bununla birlikte Scratch diğer programlama dillerine geçiş için de kullanılabilir (Malan ve Leitner, 2007). Böylelikle programlama mantığını kavrayan öğrenen, yeni bir programlama diline bilgilerini transfer edebilir (Wolz, Leitner, Malan ve Maloney, 2009). Bu nedenle Scratch gibi tasarıma dayalı görselleştiricilerin ve oyun tasarım sürecinin programlama mantığı ve programlama öğretiminde etkililiğinin araştırılması, programlama öğretiminde transfere etkisinin araştırılması, bununla birlikte öğrencilerin motivasyon gibi duyuşsal özelliklerine ve programlama becerileri gibi bilişsel özelliklerine etkisinin incelenmesi gerektiği söylenebilir.

Kuramsal Çerçeve

Programlama ve Programlama Öğretimi

En basit tanımıyla programlama; program geliştiricilerin yapmak istediği işleri bilgisayarın anlayabileceği şekilde komutlar yardımıyla bilgisayara anlatma ve aktarma sürecidir. Diğer bir ifadeyle analizi ve tasarımı yapılarak çözümüne ilişkin adımları oluşturulmuş bir problemin programlama dilleri kullanarak bilgisayar ortamına aktarılması işidir. Programlama dilleri ise programlama sürecini bilgisayar ortamında gerçekleştirmeyi sağlayan özel komut ve sembollerden oluşan yapay dillerdir

(Eryılmaz, 2003). İlk bilgisayarlar ile birlikte kullanıcı ve bilgisayar arasındaki iletişimi kurmak için makine dili kullanılmıştır. Makine dili ilk bilgisayarların geliştirilmesiyle ortaya çıkan birinci nesil programlama dilleri arasında yer almaktadır. Makine dili bilgisayarın çalışma mantığıyla 1-0 dizilerinden oluşan ve bilgisayara doğrudan arada bir editör ya da çevirici bulunmadan yazılan dillerdir. Makine dili öğrenmesi ve kullanılması zor bir dildir, bu nedenle günümüzde de halen kullanılan üçüncü nesil ve sonrasını kapsayan programlama dilleri ortaya çıkmıştır. 1954 yılında Fortran ile makine dilinden insanların kullandıkları dile yakın komutlar içeren yeni nesil programlama süreci başlamıştır. Bu diller her bilgisayarda çalışabilen ve öğrenilmesi makine dillerine göre daha kolay programlama dillerdir. Daha sonraları ise Algol, Cobol, Basic, Pascal ve C gibi diller ortaya çıkmıştır (Çamoğlu, 2009). Artık günümüzde bu dillerin daha üst sürümü olan diller kullanılmaktadır. Örneğin, programlamanın görsel öğeler ile yapıldığı Visiual Basic, Delphi ve nesneye dayalı programlamayı destekleyen C++, C# ve Java gibi birçok dil kullanılmaya başlamıştır.

Makine dilleriyle başlayan programlama sürecinde günümüze kadar 500'e yakın programlama dili geliştirilmiştir. Tüm bu dillerin kendine has söz dizimleri ve yazım kuralları vardır. Dolayısıyla bir programcının bu dillerin birçoğunu öğrenmesi zor görülmektedir. Programlama dillerine ait komut ve yazım kuralları farklılık gösterse de, çözüm için kullanılacak programlama mantığı tüm dillerde benzerdir. Bu nedenle bir programcının programlama dilinden önce programlama mantığını öğrenmesi gerekmektedir. Bayman ve Mayer (1988) programlama süreci ile ilgili edinilmesi gereken üç tür programlama bilgisinden söz etmektedir. Bunlar kavramsal bilgi (Conceptual), yazımsal bilgi (Syntactic) ve problem çözme bilgisidir (Strategic). Bu sınıflama Shneiderman ve Mayer (1979)'in yazımsal ve mantıksal programlama bilgisi modeline dayanmaktadır. Yazımsal ve mantıksal programlama bilgisi modelinde; yazımsal bilgi bir programlama diline ait komutları ve yazım kurallarını, mantıksal bilgi ise programlama dilinden bağımsız genel programlama yapılarını ve ilkelerini kapsamaktadır. Bayman ve Mayer (1988)'e göre yazımsal bilgi bir programlama dilinin yapısal kurallarını, kavramsal bilgi ise program içindeki temel kavramları anlama ve mantığını kavrama bilgisidir. Stratejik bilgi ise karşılaşılan problemi fark etme, tanımlama, mantıksal ve yazımsal bilgiyi kullanarak çözme bilgisini içermektedir. McGill ve Volet (1997) programlamada, öğrenenlerin kazanmaları gereken bu bilgileri

bildirimsel bilgi ve işlemsel bilgi şeklinde yeniden sınıflamıştır. Modelde söz edilen bildirimsel bilgi, bir programlama diline ait kuralların bilinmesi; işlemsel bilgi ise programlama dilinin kurallarına uyarak kod yazabilme olarak tanımlanmaktadır (McGill ve Volet, 1997). Bu modelde söz edilen sınıflamaya ilişkin bilgiler programlama dilleri öğrenim ve öğretim tablosunda (Tablo1) özetlenmiş ve örneklendirmiştir.

Tablo 1

Programlama Dilleri Öğrenim ve Öğretim Tablosu (McGill ve Volet, 1997)

	Bildirimsel Bilgi (Declarative)	**İşlemsel Bilgi (Procedural)**
Kavramsal Bilgi (Conceptual)	Program çalıştırıldığında hangi mantıksal işlemlerin yapıldığını anlama ve açıklama becerisi **Ör**: Yalancı (Pseudo) kodun ne iş yaptığını açıklayabilme.	Bir programlama problemine yönelik çözüm oluşturabilme. **Ör**: Bazı sayıların ortalamalarını bulan bir "Procedure" tasarımlayabilme
Yazımsal Bilgi (Syntactical)	Programlama dilinin yazım kuralları bilgisi **Ör**: Java programlama dilinde her komutun sonuna noktalı virgül işaretinin konulması gerektiğini bilme.	**Ör**: Java programlama dilindeki While komutunu yazım kuralı olarak doğru yazabilme
Stratejik Bilgi (Problem Solving)	Alışılmışın dışında bir problemle karşılaşıldığında buna ait bir çözüm üretmek için program tasarımı kodlama ve test edebilme becerisi.	

Bayman ve Mayer (1988) programcılığı yeni öğrenen bireyler üzerinde yaptığı çalışmalarında kavramsal bilgi ile öğrenenlerin problem çözme performansları arasında anlamlı bir ilişki bulmuşlardır. Buna göre öğrenenlerin programlamada başarılı olabilmeleri için temel programlama kavramlarını ve programlama mantığını iyi kavramaları gerekmektedir. Üst düzey bilişsel beceri gerektiren programlama becerisinin kazandırılması için temel programlama kavramlarının ve programlama mantığının kazandırılması önemlidir. Eryılmaz (2003) temel programlama kavramlarını değişkenler, karar verme ve kontrol işlemleri, döngüler ve diziler olarak adlandırmıştır. Bayman ve Mayer (1988)'in modeline göre bu temel kavramlar ya da yapılar farklı programlama dillerinde farklı kodlar şeklinde kullanılsa da tüm dillerde çalışma mantığı benzerdir.

Programlamaya ilişkin temel yapılar, programlama mantığı ve problem çözme süreci; programlamaya yeni başlayanlar için temel ders sayılan "Programlamaya Giriş", "Programlama Dillerinin Temelleri" veya "Programlama Dilleri I" gibi farklı isimlerdeki programlama derslerinde öğretilmektedir. Bu derslerde bir programlama

diline geçilmeden önce genellikle algoritma problemleri kullanılarak temel programlama yapılarının öğretilmesi ve programlama mantığının kazandırılması amaçlanmaktadır. Genellikle bu dersler programlama dilleri dersleri için ön koşul ders niteliğinde ve programcıların hazırbulunuşluk düzeylerini belirleyen derslerdir. Birçok eğitimci bu derslerde belirli bir programlama diline ait komutları ve kuralları öğretmeye odaklanmaktadır (Al-Imany, Alizadeh ve Nour, 2006; Deek ve Espinosa, 2005; Schulte ve Bennedsen, 2006; Özdener, 2008). Dolayısıyla programlamaya yeni başlamış olan programcılar hem mevcut programlama diline ait kodları ve kuralları hem de programlama mantığını öğrenmek zorunda kalmaktadırlar. Öğrenciler mevcut programlama diline ait bütün komutları ve kuralları bilseler bile programlama mantığını kavramada sorun yaşamakta ve yeni karşılaştıkları farklı bir programlama problemini çözmekte sorun yaşamaktadırlar (Cooper ve diğerleri, 2003; Garner ve diğerleri, 2005; Kinnunen ve Malmi, 2008). Bundan dolayı programlamaya yeni başlayanlar için doğrudan bir programlama dilinin öğretilmesi yerine programlama mantığının ve problem çözme sürecinin kazandırılması daha uygun bir süreçtir. Özellikle temel programlama derslerinde bir programlama diline geçilmeden önce temel programlama yapıları programlama mantığı çerçevesinde algoritma problemleri kullanılarak öğretilmektedir.

Programlama bir problem çözme sürecidir ve programcıdan beklenen en temel beceri, programlama problemini anlayarak en elverişli şekilde çözümleyebilmesidir (Munson, Moskal, Harriger, Karriker ve Heersink, 2011). Algoritma en temel tanımı ile bir problemin çözümü için izlenecek adımlar bütünü olarak tanımlanabilir (Eker, 2011). Teknik anlamda ise algoritma bir program probleminin çözümüne ilişkin işlemlerin adım adım bilgisayara iletilmesi olarak da tanımlanmaktadır (Vatansever, 2011). Günümüzde matematik ve bilgisayar biliminde birçok problemin çözümünde kullanılmaktadır. Algoritma problemleri için kullanılan dil belli kuralları olmayan ve konuşma diline yakın bir dildir. Algoritmalar, programcılara bir programlama problemine ilişkin çözümleme süreci oluşturmasına olanak veren basit ifadelerdir. Programcı bir programı tasarlamadan önce algoritmasını oluşturarak test eder ve adım adım bilgisayara aktarır. Dolayısıyla algoritmalar, programcıya bir problemi bilgisayara nasıl aktaracağına ilişkin ortamı sağlamaktadır. Aslında temel amaç algoritma problemleri ile programlamaya ilişkin temel yapıların ve programlama mantığının

problem çözme süreci içinde kazandırılmasıdır. Bu problemlerin çözümünde genellikle programlamanın temeli olan değişken tanımlama ve değer atama, karar verme ve kontrol yapıları, sayaç ve döngü yapıları, dizi oluşturma gibi temel programlamama işlemleri gerçekleştirilmektedir. Ancak bu işlemler soyut olduğu için birçok öğrenci programlama problemini anlama ve çözme sürecinde sorun yaşamaktadır (Lahtinen ve diğerleri, 2005; Özdener, 2008). Özellikle programlamaya yeni başlayanlar açısından bir programlama problemini basamaklara ayırmak ve modellemek, ayrıca tüm bu süreçlerde programlamaya ilişkin kavramları kullanmak oldukça zordur. Programlama eğitiminde bu soyut ve zor sürecin aşılması için algoritma problemlerinin ya da programlama problemlerinin çözümünde genellikle akış diyagramları kullanılmaktadır. Akış diyagramları algoritma problemlerinin sembolleştirilmiş halidir (Eker, 2011). Algoritma problemleri genel olarak basamaklı satır halinde yazılan konuşma dili şeklindedir. Akış diyagramları bu süreci görselleştirmektedir. Birçok programcı bir programlama problemini bilgisayarda çözümlemeden önce algoritmasını akış diyagramlarını oluşturarak test eder. Akış diyagramlarında her bir programlama yapısının gösterimi için farklı semboller kullanılır. Programın çalışma adımları akış diyagramları arasında akış yönü ile takip edilir. Böylelikle süreç görselleştirilmektedir. Ancak görselleştirme daha çok geometrik şekillere benzer semboller ile sınırlı kalmaktadır. Algoritmalar ve akış diyagramları programlama mantığı öğrenme sürecini somutlaştırmada yetersiz kalmakta ve öğrenciler bu süreçte programlama mantığı kavramakta sıkıntı yaşamaktadırlar (Garner, 2007; Garner ve diğerleri, 2005; Robins ve diğerleri, 2003).

Programlama öğretiminde yaşanan sıkıntılardan dolayı birçok öğrenci programlama dilleri dersini zor ve sıkıcı bir ders olarak görmekte ve bu dersten başarısız olmaktadır (Bennedsen ve Caspersen, 2008; Farkas ve Murthy, 2005). Hatta bilgisayar bilimleri alanında eğitim gören birçok öğrenci programlama dersleri yüzünden bölümünü bırakmakta ya da gelecek yaşamında programlama ile ilgili bir iş yapmayacağını belirtmektedir (Başer, 2013; Heersink ve Moskal, 2010). Bu bağlamda programlama öğretiminde motivasyonun önemli olduğu ve programlama öğretiminin güdülenmiş öğrenciler gerektirdiği söylenebilir (Forte ve Guzdial, 2005; Molis-Ruano, Sevilla, Santini, Haya, Rodriguez ve Sacha, 2014).

Programlama Öğretiminde Motivasyon

Motivasyon bir davranışın oluşması için en gerekli dürtüdür. Motivasyon bireyleri davranışa iten, bu davranışın düzeyini belirleyen, davranışa yön veren ve davranışın devamını sağlayan bir etmendir (Deci, Kostner ve Ryan, 2001). Özellikle öğrenme sürecinde motivasyonun rolü büyüktür. Bir öğrenme süreci ile ilgili öğrenenlerde motivasyon sağlanırsa öğrenmenin gerçekleşme oranı daha yüksektir. Öğrenme motivasyonunu etkileyen dışsal ve içsel etkenler bulunmaktadır. Dışsal etkenler, yani ödül, ceza, takdir edilme, baskı, rica, sevilme, kabul görme gibi dışsal uyaranlar motivasyonunu etkileyebilmektedir. İçsel etkenler yani içsel motivasyon ise öğrenenlerin öğrenmeye ve başarmaya karşı tutumu, ilgileri, dikkat düzeyleri ve kişilik özellikleri olabilir. Bir başka deyişle içsel motivasyon daha çok bireyin kendisinde öğrenme ihtiyacı ve arzusu oluşması durumudur. Ayrıca bir öğrenenin motivasyonu hedefe ulaşma ve başarıya yüklediği değer ile ilişkilidir. Eğer bir öğrenenin başarma şansı varsa motivasyonu yükselir, başarısızlık olasılığı varsa motivasyonu düşer (Schunk, 2003). Programlama derslerindeki motivasyon düşüklüğü de daha çok başarısızlıktan kaynaklanmaktadır (Forte ve Guzdial, 2005; Jenkins, 2002; Lahtinen, ve diğerleri, 2005; Molis-Ruano ve diğerleri, 2014; Porter ve Calder, 2004). Programlamanın zor ve karmaşık oluşu başarma duygusunu azaltmakta ve içsel motivasyona etki etmektedir. Deci ve arkadaşları (2001) öğrenenlerin özerk olduğu, yeteneklerini sergileyebildiği, merak uyandıran, eğlenceli ve hedefler ile iyi ilişkilendirilmiş bir öğrenme ortamında içsel motivasyonun sağlanabileceğini vurgulamaktadırlar. Ayrıca öğrenme ortamında öğrenenlerin pekiştirilmesi dışsal motivasyonu sağlayabilmektedir.

Programlama öğretiminde yaşanan sıkıntıları gidermek ve öğrencilerin programlama dersindeki motivasyonlarını artırmak için öğrenme ortamları; öğrenenlerin aktif olduğu, öğrenme sürecine kendisinin karar verdiği, daha somut ve eğlenceli, hedefler ile ilişkilendirilmiş şekilde tasarlanmalıdır. Programlama öğretiminde öğrenen başarısını ve motivasyonunu artırmak için görselleştiriciler kullanılmaktadır (Cooper ve diğerleri, 2006; Kelleher ve Pausch, 2005; Salleh, Shukur ve Judi, 2013). Bu görselleştiriciler, özellikle programlama için giriş niteliğindeki derslerde temel programlama mantığının kazandırılması için kullanılmaktadır.

Programlama Öğretiminde Görselleştirici Kullanımı

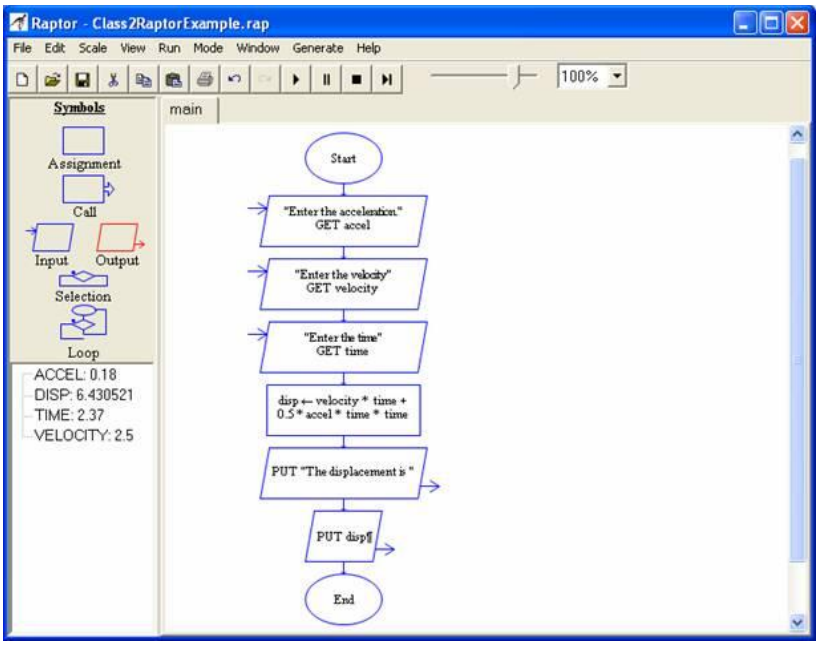
Programlama öğretiminde yaşanan sıkıntıları azaltmak, soyut ve zor bir süreç olan programlama öğretimini daha zevkli hale getirmek ve özellikle de programlamaya yeni başlayanlara programlamayı sevdirmek için birçok araç kullanılmaktadır. Bu araçların genel amacı programlama sürecini görselleştirerek programlamanın daha anlaşılır olmasını sağlamaktır (Bergin ve Martinez, 1996). Görselleştiricilerin en güçlü yönü soyut olan programlama kavramlarını şekiller ya da animasyonlar ile destekleyerek somut hale getirmesidir. Öğrenciler açısından soyut olan "değişken, döngü, eğer, dizi" gibi temel programlama yapılarının nasıl çalıştıklarını görselleştirerek somut hale getirirler. Ayrıca anında dönüt vermesi ile kullanıcılar programlama hatalarını en aza indirgeyebilmekte böylelikle problem çözme sürecine daha çok odaklanabilmektedirler. Bunun yanında programlamayı görselleştirme araçları ile öğrenme metin tabanlı öğrenmeye göre daha ilgi çekici olabilmektedir (Cooper ve diğerleri, 2006).

Görselleştiricilerden bazıları yalnızca süreci görselleştiren durağan araçlar iken, bazıları ise süreci canlandıran animasyonlu araçlardır. Son yıllarda kullanıcıları da sürecin içine katan ve onlara tasarım olanağı sunan görselleştiriciler geliştirilmiştir. Alanyazın incelendiğinde görselleştiriciler; program görselleştirme, algoritma animasyonları ve veri görselleştirme şeklinde farklı başlıklar altında isimlendirilmektedir (Bergin ve Martinez, 1996). Cooper ve arkadaşları (2006) ise görselleştiricileri; programlama çıktılarını görselleştiren araçlar, akış şeması modelleme araçları, görsel programlama araçları ve algoritma hikâyeleştirme araçları şeklinde isimlendirmişlerdir. Ancak görselleştirme amaçlı kullanılan yeni yazılımlar ele alındığında bu görselleştiricileri; statik (durağan) görselleştiriciler ve dinamik (hareketli) görselleştiriciler şeklinde ele almak mümkündür.

Statik (Durağan) Görselleştiriciler

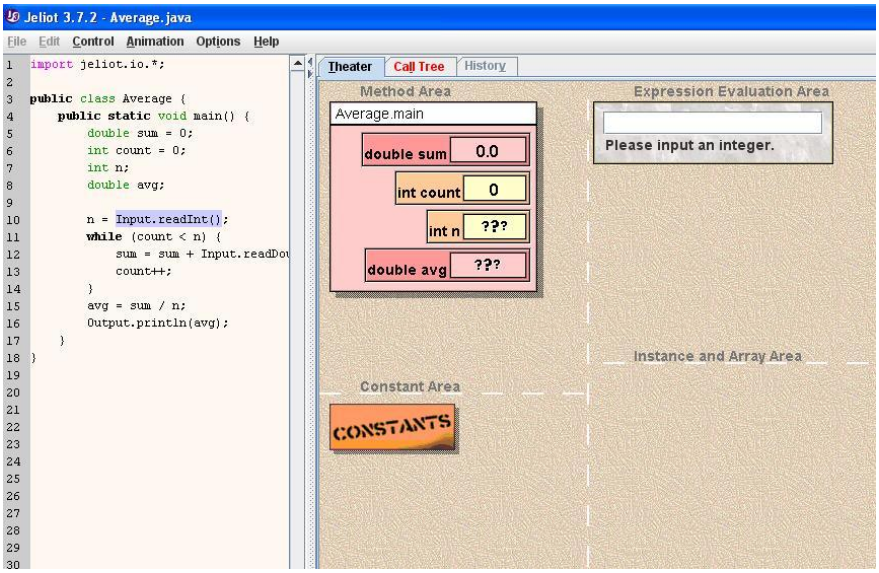
Bu tür görselleştiriciler daha çok oluşturulan programın ya da algoritma yapısının sonucunu grafiksel olarak kullanıcıya sunmaktadırlar. Algoritma görselleştirme ve program görselleştirme yazılımları özellikleri gereği durağan görselleştirme araçlarıdır. Algoritma görselleştirme araçları; kullanıcıya yazılım yardımıyla akış şeması oluşturma olanağı veren yazılımlardır. Kullanıcı akış diyagramlarını oluşturmak için programın kütüphanesinde yer alan diyagramları sürükle bırak ile hızlı bir şekilde

oluşturabilmektedir. Bu tür araçlar kullanıcılara oluşturduğu algoritma yapısını diyagramlar ile test etme ve program hatalarını en aza indirme olanağı sağlamaktadır (Cooper ve diğerleri, 2006). Flint, Raptor ve Visual Logic gibi araçlar bu tür görselleştiricilere örnek gösterilebilir. Raptor algoritma görselleştirme aracının arayüzü Şekil 1'de yer almaktadır.



Şekil 1: Raptor Arayüzü

Program görselleştirme araçları; daha çok verilerin grafiksel gösterimini sağlayan görselleştiricilerdir. Bu görselleştiriciler kullanıcıya kütüphanesinde yer alan görsel ya da yazılı kodların birleşiminden oluşan program çıktısının nasıl olduğunu kullanıcıya grafiksel olarak sunmaktadır (Bergin ve Martinez, 1996). Kullanıcı etkileşimi, programı kontrol etme ve verilerde değişiklik yapma ile sınırlıdır. Ville, Jeliot3 ve Bluej gibi araçlar bu tür görselleştiricilere örnek gösterilebilir. Jeliot3 program görselleştirme aracının arayüzü Şekil 2'de yer almaktadır.

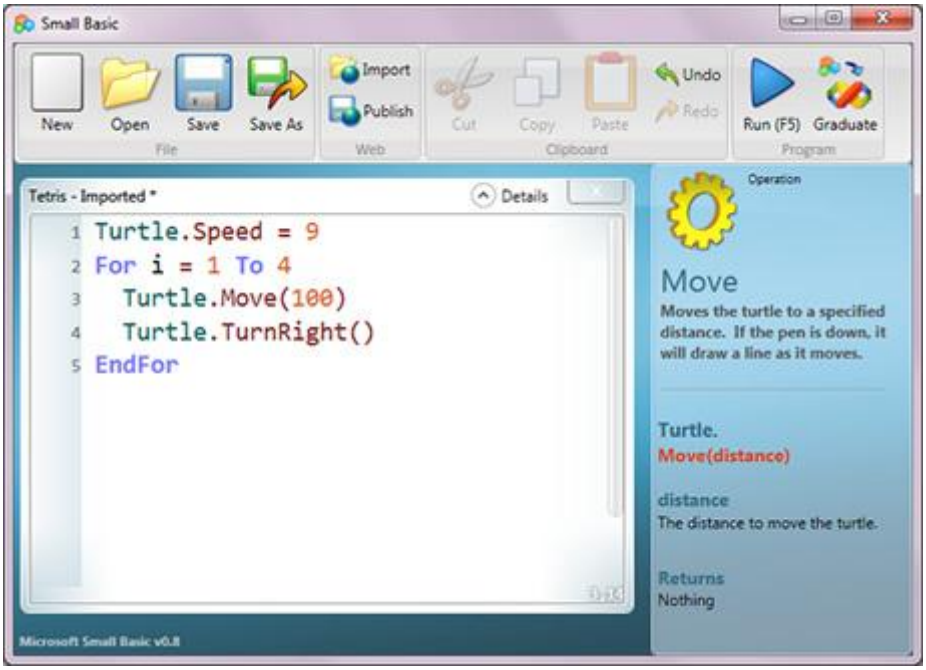


Şekil 2: Jeliot3 Arayüzü

Dinamik (Hareketli) Görselleştiriciler

Bu tür görselleştiriciler daha çok oluşturulan programın ya da algoritma yapısının sonucunu animasyon olarak kullanıcıya sunmaktadır. Algoritma ve programlama animasyon araçları, hikâyeleştirme ve tasarım araçları özellikleri gereği dinamik görselleştirme araçları olarak ele alınmaktadır.

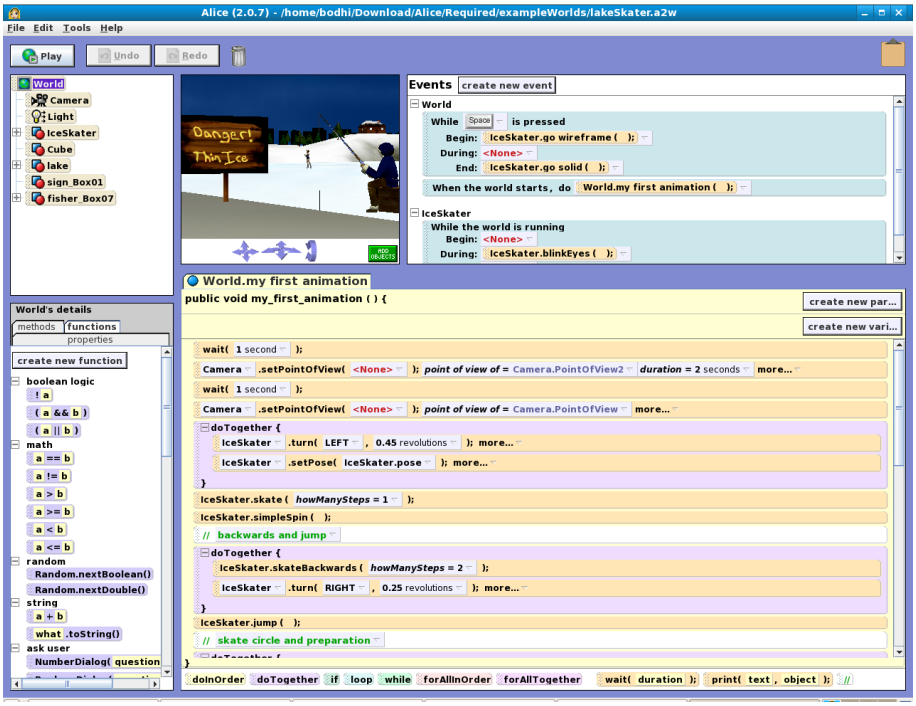
Algoritma ve programlama animasyon araçlarında görselleştirme yalnızca grafiklerle sınırlı değildir. Oluşturulan program ya da algoritma sonucu hareketli ve etkileşimli bir şekilde kullanıcıya iletilir. Diğer görselleştiricilerde de olduğu gibi kullanıcıya oluşturduğu program ya da algoritma yapısını test etme olanağı verir. Kullanıcı, görselleştirme sürecine kendisi karar verir ve neyi görselleştireceğini kendi belirler. Bu tür araçlarda durağan görselleştiricilere göre kullanıcı etkileşimi daha yüksek düzeydedir ve programlama hataları en aza indirgenmiştir. Kodlama hatalarından daha çok mantık hataları oluşabilmektedir. Görselleştirici çalıştığında kullanıcı program yapısının ya da algoritma yapısının nasıl çalıştığını hareketli bir şekilde görebilmekte, sürece müdahale edebilmekte ve hata ayıklama yapabilmektedir (Cooper ve diğerleri, 2006). Small Basic, AppInventor, JES, Samba ve TheSortAnimator II gibi araçlar bu tür görselleştiricilere örnek gösterilebilir. Small Basic algoritma ve programlama animasyon aracının arayüzü Şekil 3'te yer almaktadır.



Şekil 3: Small Basic Arayüzü

Bu tür programlarda hikâyeleştirme ve tasarım araçları ile sürecin görselleştirilmesinin yanında programlamanın karakter yardımıyla hikâye ya da oyun tasarımı yapılarak öğretilmesi amaçlanmaktadır. Kullanıcı programda yer alan karakteri kullanarak bir hikâye ya da oyun tasarımı yaptığından kullanıcı etkileşimi en üst düzeydedir. Kullanıcı tasarım sürecinde kendi oyun ya da hikâye senaryolarını belirler ve tasarım sürecine kendisi karar verir. Kullanıcının süreç içinde bu şekilde aktif olması kullanıcıların ilgisini çekmekte ve motivasyon düzeylerini üst düzeyde tutmaktadır (Cooper ve diğerleri, 2006; Papert, 1980). Ayrıca tasarım sürecinde resim, ses gibi çoklu ortam öğelerini kullanarak süreç daha ilgi çekici hale getirilebilmektedir.

Bu tür görselleştiricilerin birçoğunda programlama yapıları ve algoritma yapısı görselleştirilmiştir ya da konuşma dili şeklindedir. Kod yazmak yerine daha çok sürükle bırak yapısında olan bloklar kullanıldığından kodlama hatası en aza indirgenmiştir. Kullanıcı süreç içinde aktif olduğu için hataları ayıklayabilmekte, sürece müdahale edebilmekte ve tasarımı değiştirebilmektedir. Alice, Scratch, StageCastCreator ve ToonTalk gibi araçlar bu tür görselleştiricilere örnek gösterilebilir. Alice hikâyeleştirme ve tasarım aracına ait arayüz Şekil 4'te yer almaktadır.



Şekil 4: Alice arayüzü

Görüldüğü üzere programlama öğretiminde birçok görselleştirici araç kullanılmaktadır. Ancak son zamanlarda hikâye ve oyun tasarımına olanak veren görselleştiriciler kullanılmaktadır (Cooper ve diğerleri, 2006). Özellikle dijital oyun tasarımı ilgi çekici özelliği ve öğrenenler açısından sağladığı üstünlüklerle programlama öğretiminde etkili bir yöntem haline gelmiştir (Moreno, 2012; Wang, McCaffrey, Wendel ve Klopfer, 2006). Ayrıca oyunlar yapısı gereği döngü, karar verme yapıları gibi programlama yapılarını kullanmayı ve programlama yapmayı gerektirmektedir (Kafai, 2006). Bu nedenle dijital oyun tasarımı programlama öğretimi sürecinde kullanılması gereken etkili bir yöntem olarak ele alınabilir.

Dijital Oyun Tasarımı ve Eğitimde Kullanımı

Bilgisayar oyunları diğer ifadeyle dijital oyunlar; 1990 yıllarda masaüstü bilgisayarların yaygınlaşmasıyla popülerliği artmaya başlayan ve hayatın bir parçası haline gelen bilgisayar yazılımlarıdır (Oblinger, 2004). Özellikle 12-18 yaş çocukların tercih ettiği dijital oyunlar en çok zaman harcanan bilgisayar etkinliğinden biridir (Papastergio, 2009). Prensky (2001) dijital oyunları bu kadar eğlenceli ve zevkli yapan temel özelliklerini şu şekilde sıralamıştır:

- *Oyunlar kurallar içerir*: Oyunların kuralları ve sınırları vardır. Amaca nasıl ulaşılacağı, nasıl puanlanacağı ve seviyelerin nasıl geçileceği kurallar dâhilinde belirlenmiştir.
- *Oyunlar hedefler içerir*: Oyunların bir amacı vardır. Oyuncu o amaca ulaşmak için çaba harcar. Bu durum oyun içinde kullanıcıyı güdüleyen en önemli etmendir.
- *Oyunlar kullanıcıya dönüt sağlar:* Oyuncuyu yönlendiren ve sonuca götüren etkendir. Oyuncuyu daha istekli hale getirir.
- *Oyunlar çekişme/zorluk/yarışma içerir:* Her oyunun bir zorluk sınırı vardır. Oyunlar tasarımı gereği içinde çekişme ve yarış olan bir ortamdır. Çekişme oyuncuyu hırslandıran, oyuna olan bağlılığını artıran bir etmendir.
- *Oyunlarda etkileşim söz konusudur*: Oyunlar; oyuncu ile bilgisayar arasında ve oyuncu ile diğer oyuncular arasında etkileşim yaratan bir ortamdır. Etkileşim, oyuncu ve bilgisayar arasındaki geribildirimi sağlarken oyuncu ve diğer oyuncular arasındaki sosyal etkileşimi de sağlar.
- *Oyunlar hikâyeden oluşur:* Oyunlar bir senaryo dâhilinde tasarlanmaktadır. Oyun en başından, eğer sonlanıyorsa, en sonuna kadar ayrıntılı şekilde hikâyeleştirilmiştir. Oyun hakkında en genel bilginin yer aldığı bölümdür.

Garris, Ahlers ve Driskell (2002) bir oyunun karakteristik özelliklerini kurallar, hayal gücü, hedefler, duygusal uyaranlar, zorluk, gizem ve kontrol olarak sıralamaktadırlar. Malone (1981)'ye göre oyunların kullanıcıların motivasyonlarını artırabilmesi için bireysel seviyede zorluk, kontrol, merak, hayal gücü, kişiler arası rekabet ve işbirliği gibi özelliklerinin olması gerekmektedir. Dijital oyunları bu kadar çekici yapan etmenler; oyunların eğlenceli, zorlayıcı, hedefe yönelik, performans odaklı olması, çekişme içermesi, strateji geliştirme ve karar verme gibi beceriler gerektirmesi olarak sıralanabilir (Kiili, 2005; Koster, 2005).

Oyunlar tüm bu özellikleriyle eğitim sürecinde kullanılma potansiyeline sahiptir (Prensky, 2001). Eğitimciler öğrencilerin oyunlara olan tutkusunu programlama öğretmek için kullanabilirler. Bu bağlamda dijital oyunlar bir öğretim materyali olarak ele alınabilir (Gee, 2005). Oblienger (2004) ve Papestrergio (2009) oyunların eğitimde kullanılabilmesi için gereken nedenleri şu şekilde sıralamaktadırlar.

Dijital oyunlar;

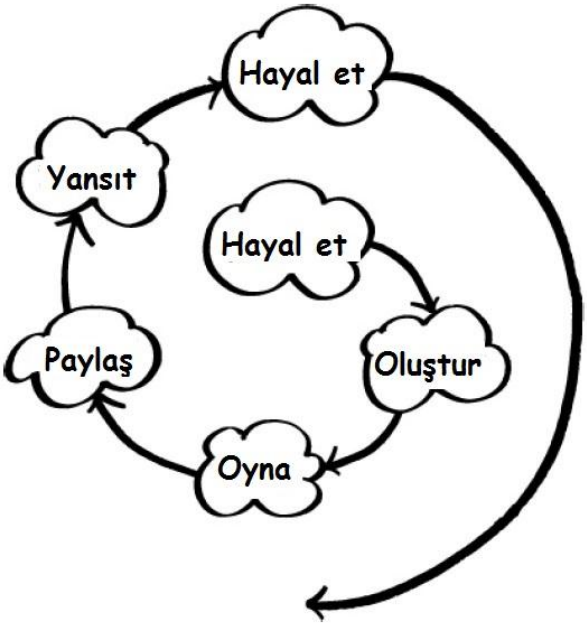
- birden çok algıya hitap eder, aktif katılım sağlar, deneyimleme olanağı verir,
- problem çözme becerilerini geliştirir,
- kullanıcıya mevcut bilgi ve becerilerini kullanma olanağı sağlar,
- yaparak öğrenme ve anında dönüt olanağı sunar,
- puanlama ve seviye atlama sayesinde bireysel olarak değerlendirme olanağı sağlar,
- birlikte oyun ortamı sağlayarak sosyal bir ortam yaratır,
- ilgi çekici yapısı ile yüksek kullanıcı motivasyonu sağlar.

Kafai (2006) eğitimde dijital oyunların kullanılması ile ilgili olarak, öğretici amaçlı kullanma (instructionist) ve yapıcı (constructionist) amaçlı kullanma olmak üzere iki yaklaşımdan söz etmektedir. Dijital oyunlarının öğretici amaçlı kullanması oyunların daha çok derslerde ya da ders dışı etkinliklerde, öğrenme sürecine dâhil edilmesi şeklindedir. Bu yaklaşımda; ticari ya da eğitsel amaçlı oluşturulmuş hazır bir bilgisayar oyununun öğretim sürecinde kullanılması söz konusudur (Kirriemuir ve McFarlance, 2004). Bir öğretmenin öğrencilerin ilgisini çekmek, dersi zevkli hale getirmek ve öğrenenleri ders sürecine dâhil etmek için "Age of Empires" isimli bilgisayar oyununu Dünya Tarihi dersinde kullanması bu duruma bir örnektir (Hayes ve Games, 2008). Dijital oyunlarının eğitimde kullanılması ile ilgili yapılmış çalışmaların büyük bir bölümü de bu şekildedir (Ke, 2014; Papastergio, 2009; Tüzün, Soylu, Karakuş, İnal ve Kızılkaya, 2009). Diğer bir yaklaşım ise yapıcı yaklaşımdır. Bu yaklaşımda temel amaç öğrenenlerin öğretme-öğrenme sürecinde dijital oyun tasarımı yapmasıdır. Eğitim etkinliği olarak oyun tasarlamak özellikle son zamanlarda basit tasarım araçlarının ortaya çıkmasıyla daha da yaygın hale gelmiştir. Bu yaklaşım, Papert (1991)'in yapılandırmacı yaklaşımına dayanmaktadır. Papert' in yapılandırmacı yaklaşımı (constructionism) Piaget'in bilişsel yapılandırmacı (constructivism) kuramına dayanan ve bireylerin kendi zihinsel modellerini oluşturduğu öğrenme yaklaşımıdır. Papert'in yapılandırmacı yaklaşımında daha çok gerçek ve paylaşılabilir ürünlerin inşa edilmesi süreci vardır. Papert (1980)'e göre en iyi öğrenme; bireylere doğrudan bilgiyi iletmek yerine, onlara bir ürün tasarlama ve oluşturma (inşa etme) olanağı verildiğinde gerçekleşmektedir. Bu ürünler bir maket, lego ve öğrenme materyali gibi elle tutulur

nesneler olabileceği gibi bir çokluortam, müzik ya da bilgisayar oyunu olabilir. Öğrenme bu ürünlerin oluşturulması sürecinde deneyimleme yoluyla gerçekleşmektedir (Harel ve Papert, 1991; Mishra ve Girod, 2006). Buna göre öğrenenler tasarım yaparken ya da inşa ederken içeriği öğrenmektedirler. Kısacası bu yaklaşımda öğrenenler eğitsel hedeflere ulaşmak için kendi oyunlarını inşa etmekte ve öğrenmede bu oyunların inşası sırasında gerçekleşmektedir (Kafai, 2006). Oyun yaparak öğrenenler bilgi ve becerilerini keşfettiği, sunduğu, test ettiği bir "mini dünya" inşa etmektedirler (Papert,1980).

"İnşa" kavramı Papert'in sürekli üzerinde durduğu bir kavramdır. Burada anlatılmak istenen tasarım sürecidir. Bu tasarım süreci sonunda birey bilgi ve becerilerini kullanarak anlamlı bir ürün oluşturmaktadır (Kafai, 2006). Tasarım; fikirlerin gerçek dünya ile yani teorinin uygulamayla birleştiği bir süreçtir (Genç ve Karakuş, 2011). Bu süreçte tasarımı gerçekleştiren birey aktif haldedir (Ke, 2014; Li, 2012). Bu durum öğrenenin motivasyonunu artırarak daha iyi öğrenmesini sağlamaktadır. Ancak burada dikkat edilmesi gereken nokta, tasarım sürecinde kazandırılmak istenen hedef ve öğretilmek istenen içeriğin ilişkilendirilmesidir (Ke, 2014).

Eğitimde oyunların hazır olarak kullanılması, öğrenenleri tam anlamıyla etkin bir katılımcı yapmamakta aksine onların yalnızca tüketici olarak kalmasına neden olmaktadır. Ancak oyun tasarımı edilgen bir deneyimlemenin ötesinde, öğrenenin etkin olduğu ve kendi öğrenme sürecini kontrol ettiği bir süreçtir. Bu nedenle oyun tasarlamak öğrenenlerin ilgisini çeken bir süreçtir (Smeets, 2005). Resnick (2007)'e göre de anlamlı öğrenme ancak tam anlamıyla bir katılım ve tam anlamıyla bir yaratma olduğu zaman gerçekleşmektedir. Bu nedenle de yaratıcı oyun tasarımı sürecini önermiştir. Resnick (2007)'e göre yaratıcı oyun tasarımı hayal etme, oluşturma, oynama, paylaşma ve yansıtma döngüsünde gerçekleşen bir süreçtir. Bu süreç Şekil 5'te yer almaktadır.



Şekil 5:Yaratıcı Tasarım Döngüsü (Resnick, 2007).

Yaratıcı tasarım döngüsü hayal etme süreci ile başlamaktadır. *Hayal etme;* tasarım sürecinin ilk aşamasıdır. Tasarımcıların yapmak istedikleri oyunu hayal etmesidir. *Oluşturma;* tasarımcıların hayal ettiklerini oluşturması ya da yaratması sürecidir. Hayal edilen fikir bu aşamada projeye ya da ürüne dönüşmektedir. *Oynama;* tasarlanan oyunun oynanması sürecidir. Oyunun ilk denendiği ve keşfedildiği aşamadır. *Paylaşma;* oluşturulan oyunun başkaları ile paylaşılması, ürünün ortaya atılması ve sunulmasıdır. Gerektiğinde diğerlerinin de tasarım sürecine dâhil edilmesi söz konusu olabilir. *Yansıtma*; geniş anlamda sürecin eleştirel olarak ele alınması, tasarımın iyileştirilmesi ya da mevcut deneyimlerin yeni tasarımlara aktarılması sürecidir. Yansıtma döngünün son aşaması değildir. Aksine bir sonraki tasarım döngüsünü tetikleyen yeni fikirler veren bir süreçtir çünkü bu döngü tekrar eden bir süreçtir. Bir sonraki basamak yeniden hayal etme sürecidir. Resnick (2007)'e göre öğrenme bu döngüsel süreç içerisinde etkin katılım, öz değerlendirme, yansıma, yaratıcı düşünme ve işbirliği ile birlikte anlamlı olmaktadır.

Oyun tasarım süreci tüm bu özellikleriyle öğretme-öğrenme sürecinde etkin şekilde kullanılmaktadır. Hayes ve Games (2008) oyun tasarımının öğretme-öğrenme sürecinde; içeriğin öğretilmesi amaçlı, bilgisayar bilimlerine olan ilgiyi artırma amaçlı, programlamayı öğretme amaçlı, oyun programlamayı ve tasarlamayı öğretme amaçlı kullanımından söz etmektedirler.

İçeriğin Öğretilmesi Amaçlı Kullanım: Bu süreçte; matematik, fen bilgisi, tarih gibi bir derse ilişkin bir konunun ya da içeriğin, öğrenenlere oyun tasarımı yaptırarak kazandırılması amaçlanmaktadır. Örneğin; öğrencilerin matematik dersinde kesirler konusuyla ilgili (Kafai, Franke, Ching ve Shih, 1998) ya da fen bilgisi dersinde çevre eğitimi ile ilgili oyun tasarlayarak içeriği öğrenmesi sağlanabilir (Baytak ve Land, 2011).

Bilgisayar Bilimlerine Olan İlgiyi Artırma Amaçlı Kullanım: Bu süreçte; oyun tasarımı ile özellikle genç yaştaki öğrencilerin bilgisayara ve bilgisayar bilimlerine karşı ilgilerinin artırılması ve ileride bilişim alanında kariyer yapma eğilimlerinin artırılması amaçlanmaktadır (Adalberg, 2013; Al-Bow ve diğerleri, 2009).

Oyun Programlamayı ve Tasarlamayı Öğretme Amaçlı Kullanım: Bu süreçte amaçlanan özellikle bilgisayar bilimleri eğitimi alan öğrencilere oyun programlamayı, oyun yapma mantığını ve tasarım yapmayı öğretmektir (Habgood ve Overmars, 2006).

Programlamayı Öğretme Amaçlı Kullanım: Bu süreçte amaç programlamaya yeni başlayanlara temel programlama mantığını öğretmek, algoritma mantığını kavratmak ve programlama becerilerini artırmaktır. Genellikle programlama öğretiminde kullanılan mevcut yöntemlerin yerine tercih edilen bir süreçtir (Cooper ve diğerleri, 2003; Moreno, 2012; Wang ve diğerleri, 2006).

Brennan (2011) ise oyun tasarım sürecini kompütasyonel düşünme ve yaratıcı kompütasyon süreci içinde ele almaktadır. Kompütasyonel düşünme en temel tanımıyla; bilgisayar okuryazarlığının ötesinde, bilgisayar kullanarak problem çözme sürecidir (Wing, 2006). Bilgisayar ve benzeri teknolojilerin sağladığı hesaplama, tasarlama, düzenleme gibi üstünlüklerin problem çözme sürecinde kullanılmasıdır. Wing (2008)'e göre kompütasyonel düşünme problem çözmeyi, yaratıcılığı, algoritmik düşünmeyi, programlama mantığını ve programlama becerisini kapsayan geniş bir kavramdır. Yaratıcı kompütasyon ise bilgisayarın sağladığı üstünlükleri bireylerin yaratıcı süreçte kullanmasıdır (Brennan, Balch ve Chung, 2014). Brennan (2011)'e göre oyun tasarımı; tasarımcılarda yaratıcılık, problem çözme gibi becerilerin yanında matematik, programlama, algoritmik düşünme gibi kompütasyonel düşünme becerilerini de geliştirmektedir.

Görüldüğü üzere dijital oyun tasarımı öğretme-öğrenme sürecinde etkin şekilde kullanılmaktadır. Ayrıca dijital oyunlar yapısı gereği bir programlama süreci

içermektedir. Bu nedenle dijital oyun tasarımı özellikle son zamanlarda programlama öğretimi sürecinde etkin şekilde kullanılmaya başlanmıştır (Cooper ve diğerleri, 2003; Moreno, 2012; Wang ve diğerleri, 2006).

Programlama Öğretiminde Dijital Oyun Tasarımı

Dijital oyun tasarımı ilgi çekici ve eğlenceli yapısıyla programlama öğretiminde de etkin şekilde kullanılmaktadır. Eğitimciler, özellikle programlamaya yeni başlayanlar için, sürece uyum sağlanması açısından, programlama derslerinde oyun tasarımı etkinliklerinin yapılması gerektiğini vurgulamaktadır (Gee, 2005; Moreno, 2012; Rajaravivarma, 2005). Böylelikle programlamaya yeni başlayanların oyun tasarımı yaparak motivasyonlarının artırılması ve programlamaya karşı olumlu bir tutum kazanmaları amaçlanmaktadır. Oyun tasarımı ile birlikte genellikle zor ve sıkıcı olarak görülen programlama dersi daha zevkli ve eğlenceli hale gelebilmektedir.

Yapılan çalışmalar incelendiğinde programlama öğretiminde oyun tasarımı sürecinin iki farklı şekilde kullanıldığı görülmektedir. Birincisi; yeniden düzenlenebilir oyunların (oyun modifikasyonu) programlama öğretiminde kullanılması sürecidir. Bu süreçte açık kaynak kodlu oyunlar ya da oyun sırasında kodlar yardımıyla düzeltme, yönlendirme ve desen değiştirme gibi müdahalelere izin veren oyunlar kullanılmaktadır (Becker ve Parker, 2007; Chaffin, Doran, Hicks ve Barnes,2009; Ladd, 2006). Bu süreçte Robocode, Colobot, Catacombs ve Saving Serra gibi açık kaynak kodlu oyunlar ya da hazır oyun motorları kullanılabilmektedir (Kazimoglu, Kiemani, Bacon ve MacKinnnon, 2011). Genellikle kod yazarak mevcut oyunların yeniden düzenlenmesi (modifikasyonu) söz konusudur. Ancak bu durum programlamaya yeni başlayanlar için karmaşık ve zor olarak algılanabilmektedir. Bu nedenle, özellikle temel programlama öğretiminin yapıldığı derslerde, düzenlenebilir oyunların kullanılması yerine kodların tamamen ortadan kaldırıldığı ve oyun tasarımının sürükle bırak ile yapıldığı ikinci bir yöntem kullanılmaya başlanmıştır. Bu yöntem birinci yönteme göre daha basit, eğlenceli ve yapıcıdır. Kodlama hatası ve karmaşıklık gibi öğrenenlerin süreçten sıkılmasına neden olacak etmenler tamamen ortadan kaldırılmıştır. Öğrenciler programlama sorunlarıyla yüzleşmek yerine oluşturdukları senaryoyu oyuna dönüştürme eğilimindedir. Bu süreçte oyun tasarımı daha çok programlamaya giriş düzeyinde bir dil öğretmek yerine; programlama mantığını kavrama, algoritma kurma, hata ayıklama, problem çözme becerisi kazandırma, programlamaya karşı motivasyon

sağlama ve korkuları yenme amaçlı kullanılmaktadır (Cooper ve diğerleri, 2003; Moreno, 2012; Wang ve diğerleri, 2006).

Oyun tasarımında öğrenen, öğrenme sürecinin tam merkezinde, karar verme pozisyonunda ve süreci yöneten kişidir. Her ne kadar programlama öğretiminde basit araçlar kullanılsa da oyun tasarımı yalnızca programlama sürecinden ibaret değildir. Bunun yanında tasarımcı, oyun senaryosunu ve karakterleri oluşturur, oyun kuralları planlar ve grafiklerini tasarlar. Tüm bunlar ele alındığında oyun tasarım süreci genellikle iş yükü çok, uzun aşamalı ve karmaşık bir süreç olarak görülmektedir. Tasarımcı bu tür zorluklarla mücadele etmektedir. Bu zorluklar çoğu zaman tasarımcıyı güdüleyen bir etmen olsa da tasarımın uzaması ya da başarısızlıkla sonuçlanması programlama öğrenme sürecini aksatmaktadır. Brandt, Messeter ve Binder (2008) bu zorlukların aşılmasında çok katılımcılı tasarım sürecinin etkili olduğunu vurgulamaktadırlar. Buna göre çok katılımcılı tasarım süreci bir grup tasarımcının işbirliği için bir arada toplandığı, açık ve basit kurallar içeren rollerin belirlendiği bir oyunu tasarlama süreci olarak ele alınır. Birlikte oyun tasarımı katılımcılar açısından eğlenceli bir araştırma süreci oluşturabilir, görevleri eşit dağılmış bir işbirliği zorluk seviyesini düşürebilir, verimli fikir üretimi ve etkili bir tasarım oluşturulmasına olanak verebilir. Ayrıca katılımcılar arasında oluşan iletişim, tasarım sürecinde farklı bakış açılarının oluşmasını sağlamaktadır (Muller, 2003). Buna göre programlama öğretiminde oyun tasarım süreci bireysel olarak ele alınabileceği gibi çok katılımcılı tasarım sağlanarak, işbirliğine dayalı olarak da ele alınabilir. Böylelikle programlama öğretim süreci daha da sadeleştirilerek, içeriği zengin, daha zevkli ve eğlenceli hale getirilebilir (Robertson ve Howells, 2008; Smeets, 2005).

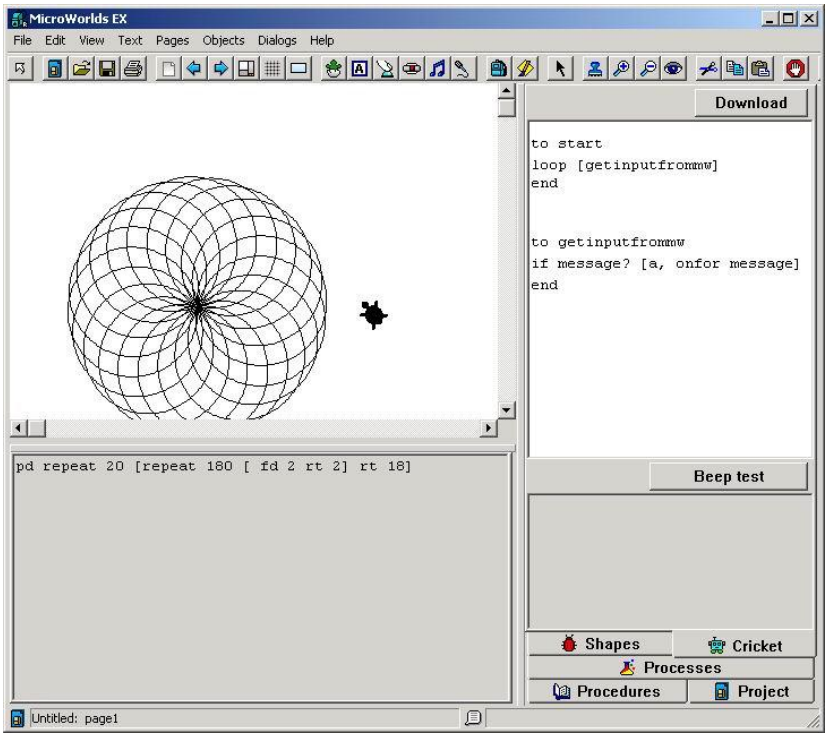
Massachusetts Institute of Technology (MIT) gençlerin ve programlamaya yeni başlayanların, hem bireysel hem de işbirliğine dayalı biçimde oyun gibi çokluortam öğeleri tasarlayarak, kolay programlama öğrenmeleri için Scratch programını geliştirmiştir (Maloney ve diğerleri, 2010).

Scratch

Scratch sözlükte "çizik" ve "tırmalamak" anlamına gelmektedir. Ayrıca Disk Jokeylerin (DJ) plaklar ile yaptıkları ses efekti olarak da adlandırılmaktadır. Scratch programını oluşturan geliştiriciler DJ'lerin yaptıkları ses efektlerini kullanarak yeni şarkılar yaratmasından esinlenerek bu programa "SCRATCH" adını vermişlerdir (Scratch,

2014). Ayrıca sözlükte "kedi tırmığı" anlamına da gelmesinden dolayı programın logosu ve program içinde yer alan karakter bir kedidir.

Scratch'in temeli 1960 yıllarda Seymour Papert tarafından MIT Yapay Zekâ Laboratuvarlarında geliştirilen Language of Graphical Output (LOGO) görsel çıktı programına dayanmaktadır. LOGO programının temel amacı her yaştaki çocuğa görsel çıktı tasarımı yaptırarak matematiksel kavramları öğretmektir. Bu programda kullanıcı programda yer alan "kaplumbağa" karakterini komut ekranına yazdığı kodlar ile yönlendirerek geometrik şekiller çizebilmektedir. Programda kullanılan kodlar küçük çocukların bile anlayacağı kolaylıkta ve günlük konuşma diline yakın komutlardır. Kullanıcı istediği geometrik şekli oluşturabilmek için kaplumbağa karakterini bu kodlar ile yönlendirmektedir. Böylece kullanıcı kendi geometri ve matematik dünyasını programlayarak inşa edebilmektedir (Papert, 1980). Papert (1987) bu durumu çocukların "mini dünyalarını" inşa etmeleri olarak açıklamaktadır. Ayrıca LOGO programı ile kullanıcı kendi öğrenme ve bilgi oluşturma sürecini yarattığı program ile oluşturabilmektedir. Şekil 6'da LOGO programının arayüzü bulunmaktadır.



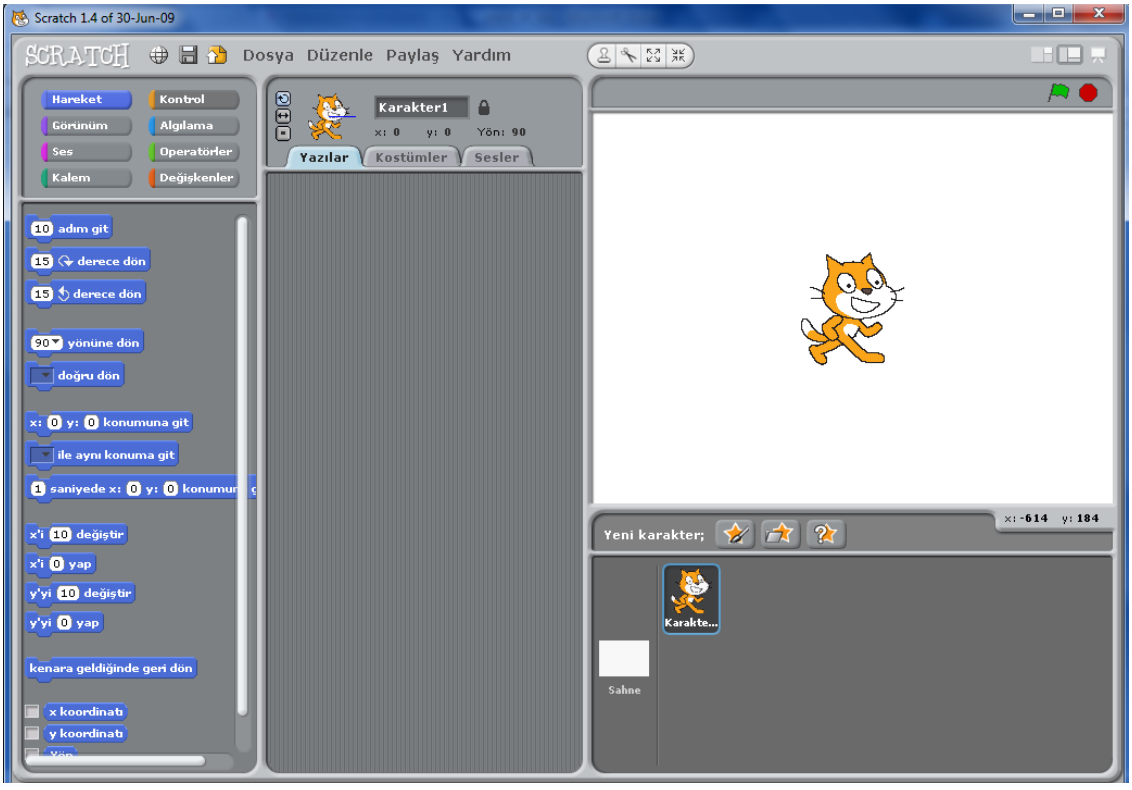
Şekil 6: LOGO Programı Arayüzü

Scratch programının en genel amacı programlamaya yeni başlayanlara programlamayı tanıtmak ve sevdirmektir. Kullanıcılar Scratch ile oyun, animasyon ya da dijital hikâyeler oluşturabilmekte, ayrıca oluşturdukları bu projeleri Scratch'in

çevrimiçi topluluk sayfasında diğer kullanıcılar ile paylaşabilmektedirler. Scratch 'in resmi sayfasındaki verilere göre 2015 yılı nisan ayı itibariyle kullanıcı sayısı 6.616.047 kişidir ve web sayfasında toplam 9.377.236 adet proje paylaşılmıştır (Scratch, 2015). Scratch'in temel özellikleri kolay arayüz, blok kod yapısı, hata ayıklama yapısı, çokluortam desteği, tasarım odaklı yapı, paylaşım ve işbirliği ve programlama yapılarına uygunluk olarak sıralanabilir. Bu özellikler aşağıda kısaca açıklanmıştır:

Kolay arayüz

Scratch programı oldukça basit bir arayüze sahiptir. Program her yaştan insanın özellikle küçük yaştaki çocukların kullanabileceği basitlikte ve üniversite düzeyinde öğrencilere programlama öğretmek için kullanılabilecek düzeyde tasarlanmıştır. Sürükle bırak yapısı ile basit düzeyde programlama ile yüksek düzeyde projeler tasarlanabilmektedir (Resnick ve diğerleri, 2009). Şekil 7'de Scrath programının arayüzü yer almaktadır.



Şekil 7: Scratch Arayüzü

Şekil 7'de görüldüğü üzere Scratch arayüzü; kodlar bölümü, kodlama paneli, sahne, karakterler paneli ve araç çubuğundan oluşmaktadır. Kodlar bölümünde her biri farklı renklerdeki kod blokları yer almaktadır. Kodlama paneli; kodlar bölümünde yer

alan kod blokların arka arkaya dizildiği ve tasarımın kodlandığı bölümdür. Sahne alanı; oluşturulan tasarımın görsel olarak canlandırıldığı yerdir. Program çalıştığında ekranda görünen bölüm burasıdır. Karakterler paneli ise canlandırmayı yapan karakter ya da karakterlerin bulunduğu bölümdür. Bu karakter Scratch'in kütüphanesinde yer alan bir karakter olabileceği gibi kullanıcının kendisinin oluşturabileceği bir karakter de olabilir (Çağıltay ve Fal; 2013). Ayrıca programda 50 farklı dil seçeneği mevcuttur.

Blok kod yapısı

Scratch programında karmaşık kodlar yerine sürükle bırak özelliğine sahip kod blokları kullanılmaktadır. Program tasarlamak için bu blokları arka arkaya dizmek yeterlidir. Bu bloklar farklı renklerde tasarlanmıştır ve her renk farklı kod bloklarını temsil etmektedir. Mavi renkte olanlar "Hareket" blok kodları, mor renkte olanlar "Görünüm" blok kodları, pembe renkte olanlar "Ses" blok kodları, koyu yeşil renkte olanlar "Kalem" blok kodları, sarı renkte olanlar "Kontrol" blok kodları, açık mavi renkte olanlar "Algılama" blok kodları, yeşil renkte olanlar "Operatörler" blok kodları ve turuncu renkte olanlar da "Değişkenler" blok kodlarıdır. Bu blok kodların üzerinde ne işe yaradığı yazılıdır. Bu kodlar bir programlama dilindeki karmaşık kod (syntax) yapısından daha çok günlük konuşma diline uygun şekilde tasarlanmıştır. Ayrıca yalnızca birbiriyle uyumlu bloklar birbiriyle kenetlenmektedir ve böylelikle hata yapma olasılığı azalmaktadır. Şekil 8'de bu blok yapıyla ilgili bir örnek yer almaktadır.

Şekil 8: Scratch Örnek Kod Bloğu

Hata ayıklama yapısı

Scratch programında blok kod yapısı sayesinde kod yazma problemi ortadan kalkmıştır. Bu durum kod yazma hatalarını (eksik kod, hatalı yazım, noktalama işaretleri gibi) ve kod ezberleme problemini ortadan kaldırmıştır. Ayrıca "adım adım çalışmayı başlat" seçeneği ile olası mantık hataları kolaylıkla görülebilmekte ve program çalışırken bloklar değiştirilerek müdahale edilebilmektedir.

Çokluortam desteği

Scratch programı kodlama panelinde arka arkaya dizilen blokların sonuçlarını sahne bölümünde canlandırmaktadır. Bloklar satır satır çalışmaya başladıkları anda sahnede yer alan karakter durumu canlandırır. Tasarımcı blok kodların çalışma mantığını ve bloklar yardımıyla oluşturulan hikâye ya da oyunun sonucu sahnedeki canlandırmayla birlikte görür (Maloney ve diğerleri, 2010). Ayrıca Scratch programında tasarlanan oyun ya da hikâyelerde resim ve ses gibi birçok çokluortam öğesi kullanılabilmektedir. Şekil 9'da Scratch' in canlandırma ekranı görülmektedir.

Şekil 9: Scratch Canlandırma Ekranı

Tasarım odaklı yapı

Scratch programı ile kullanıcılar kendi ilgi duyduğu hikâye, animasyon ya da oyunu tasarlayabilmektedirler. Özellikle tasarım odaklı yapısı ile oyun ve öğrenmeyi bir araya getirmektedir (Peppler ve Kafai, 2007). Böylelikle tasarım sürecinde kullanıcının etkin olması motivasyonu artırarak, öğrenmeyi daha da kolaylaştırmaktadır (Kafai, 2006). Ayrıca tasarımın kodlar, matematiksel kavramlar ve formüller yerine lego parçalarına benzeyen blokların kullanılması ve çokluortam öğeleri ile tasarımın zenginleştirilebilmesi süreci eğlenceli yapmaktadır. Scratch ile tasarım yapmak özellikle çocukların teknolojiyi sevmelerini, teknolojiyi daha etkili kullanmalarını sağlarken, yaratıcı düşünme, eleştirel düşünme ve işbirliği becerilerini geliştirmektedir (Brennan, 2011; Nelson ve Braafladt, 2012). Tasarım yalnızca hikâye ya da oyun gibi bir yazılımla sınırlı kalmamakta, algılayıcı (sensör) ve benzeri donanımlarla robot benzeri araçlar programlanabilmekte ve gerçek dünya ile iletişim sağlanabilmektedir.

Paylaşım ve işbirliği

Scratch programının diğer bir özelliği de yapılan projelerin kendi web sayfası üzerinden diğer kullanıcılar ile paylaşılabilmesidir. Böylelikle kullanıcılar bu web sayfası üzerinde çevrimiçi topluluklar oluşturarak görüş alışverişi yapabilmekte ve birlikte projeler oluşturabilmektedirler.

Programlama yapılarına uygunluk

Scratch programında kodlar yerine bloklar kullanılsa da bu bloklar programlama dillerinde kullanılan temel yapıları içinde barındırmaktadır. Değişken, koşullu ifadeler, döngü gibi programlama yapıları Scratch programında bloklar şeklinde yer almaktadır. Programın en temel amacı da bu yapıları kullanmayı öğreterek programlama mantığının kazandırılmasını sağlamaktır (Utting ve diğerleri, 2010). Her ne kadar Scratch 8-16 yaş aralığındaki çocuklara programlamayı sevdirmek ve öğretmek için kullanılsa da Harvard Üniversitesi, Berkley Üniversitesi ve California Üniversitesi gibi üniversiteler programlamaya giriş derslerinde Scratch'i kullanmaktadır (Resnick ve diğerleri, 2009). Scratch özellikle Java ve C gibi dillere geçiş için kullanılmaktadır.

Papert'in yapılandırmacı kuramına dayanan Scratch; basit yapısı, tasarım olanağı sağlaması ve programlama yapılarına uygunluğu gibi özellikleri ile oyun tasarımı ile programlama öğretiminde kullanılabilir (Resnick ve diğerleri, 2009; Resnick, 2012).

Yapılmış Çalışmalar

Bu bölümde programlama eğitiminde oyun tasarım etkinliklerinin öğrencilerin programlama başarısına ve motivasyonuna etkisini inceleyen çalışmalara yer verilmiştir. Çalışmalar incelenirken Scratch yazılımının yanı sıra oyun tasarımına olanak veren ve benzer özellikler gösteren araçlar da ele alınmıştır. Ayrıca Scratch ile oyun tasarım etkinliklerinin içerik öğretimi amaçlı bilgisayar bilimleri, matematik, fen bilgisi gibi derslerde akademik başarı ve motivasyonu artırmak amaçlı kullanımı, problem çözme ve yaratıcılık gibi becerileri geliştirme amaçlı kullanımını inceleyen çalışmalara da yer verilmiştir. Bu bağlamda incelenen çalışmalar "Programlama öğretiminde Scratch kullanımı", "Programlama öğretiminde diğer tasarım araçlarının kullanımı " ve " Oyun tasarımının içerik öğretiminde kullanımı" başlıkları altında toplanmıştır. Araştırmalar incelendiğinde bilgisayar bilimleri olarak nitelendirilen derslerde programlama mantığı, algoritma ya da temel programlama yapılarının ele alındığı görülmektedir. Bu nedenle bilgisayar bilimleri olarak nitelendirilen çalışmalar programlama öğretimi kapsamında değerlendirilmiştir.

Programlama Öğretiminde Scratch Kullanımı

Programlama öğretiminde Scratch kullanımıyla ilgili çalışmalar incelendiğine birçok çalışmanın ilköğretim ve lise düzeyinde olduğu, üniversite düzeyinde ise özellikle temel programlama derslerinde yapılmış çalışmalar olduğu görülmektedir. Gerçekleştirilen araştırmanın çalışma grubunu oluşturan öğrenciler üniversite öğrencileri olduğundan öncelikle üniversite düzeyinde ve yetişkinler ile yapılan çalışmalar incelenmiş daha sonra ilköğretim ve lise düzeyindeki çalışmalara yer verilmiştir.

Chiu (2014) yaptığı çalışmada Scratch ile programlama öğretiminin lise öğretmenlerinin programlama öz yeterliliklerine etkisini incelemiştir. Bu amaçla programlama ile ilgili hiçbir deneyimi olmayan öğretmenler haftada üçer saat olmak üzere Scratch ile oyun ve animasyon tasarımı yapmışlardır. Çalışma sonunda katılımcıların programlama öz yeterlilik öntest-sontest puanları arasında bir farklılık bulunmazken, katılımcıların süreç ile ilgili görüşleri incelendiğinde Scratch ile

programlama süreci ile ilgili pozitif tutum sergiledikleri, programlama için uygun bir araç olduğunu, eğlenceli ve ilgi çekici olduğunu ve sürükle bırak yapısı ile kolay öğrenilebilir olduğu belirlenmiştir.

Leiva ve Salas (2013) üniversite öğrencileri ile yaptıkları çalışmalarında Scratch ile oyun tasarım etkinliklerinin öğrencilerin programlama başarılarına etkisini incelemiştir. Bu amaçla Programlama Dilleri dersinde Veri türleri ve Modüller konusunu, kontrol grubunda araştırmacı tarafından geleneksel olarak nitelendirilen, derse devamın zorunlu olduğu, öğretmen öğrenci katılımlı laboratuvar uygulamaları ile deney grubunda ise katılımın gönüllük esasına dayandığı oyun tasarım etkinlikleri ile sürdürülmüştür. Çalışma sonunda her iki gruba da öğrencilerin programlama başarısını ölçen bir başarı testi uygulanmıştır. Ayrıca öğrencilerin, süreçten memnuniyetini ölçen; eğitimci kabiliyeti, öğrenme motivasyonu, materyalin uygunluğu gibi alt boyutlar içeren bir ölçek uygulanmıştır. Araştırma sonunda kontrol grubu %48' i başarılı iken, deney grubunda bu oran %92 olarak bulunmuştur ve gruplar karşılaştırıldığında deney grubu lehine anlamlı farklılık olduğu görülmüştür. Ayrıca öğrencilerin süreçle ilgili memnuniyeti karşılaştırıldığında yine deney grubu lehine anlamlı farklılık bulunmuştur. Araştırmacılar bu sonucu oyun tasarım etkinliklerinin öğrencilerinin motivasyonunu ve akademik başarısını artırdığı şeklinde yorumlamışlardır.

Ozoran, Çağıltay ve Topallı (2012) mühendislik öğrencileri ile yaptıkları çalışmalarında Bilgisayar Programlama derslerinde C programlama diline paralel olarak Scratch programı kullanmışlardır. Öğrenciler iki dönem boyunca ders sürecinde C programlama dillerini öğrenirken ders sonunda laboratuvar etkinliklerinde Scratch ile oyun tasarımı yapmışlardır. Dönem sonunda öğrencilerin programlama derslerindeki başarıları incelenmiş, öğrencilere ders sürecine ilişkin görüşlerini belirlemek amacıyla anket uygulanmış ve görüşme yapılmıştır. Çalışma sonunda bir yıl önce (2011) aynı dersi alan öğrencilerin final sınavı başarıları ile uygulamanın yapıldığı yılda (2012) dersi alan öğrenciler ile karşılaştırılmıştır. Final sınavlarında her iki yılda da aynı sorular sorulmuştur. Yapılan karşılaştırma sonucunda 2012 yılında Bilgisayar Programlama dersinde başarısız olan öğrenci sayısının geçen yıla göre %20 azaldığı görülmüştür. Ayrıca öğrenciler Scratch'i, C programlama diline göre daha eğlenceli ve görsel, temel programlama yapılarını öğrenmede katkı sağlayan, yaratıcılığı artıran ve oyunların çalışma mantığını daha iyi anlayan olarak gördüklerini dile getirmişlerdir.

Rizvi, Humphries, Major, Jones ve Lauzun (2011) üniversite birinci sınıf öğrencileri ile yaptıkları çalışmalarında CS0 (Bilgisayar Bilimleri) dersinde Scratch ile oyun tasarım etkinliklerinin öğrencilerin programlama öz yeterliliklerini ve programlamaya karşı tutumlarını etkisini incelemek ve devam dersi olan CS1(Bilgisayar Bilimleri) dersindeki programlama başarılarına etkisini incelemişlerdir. Bu bağlamda öğrencilere algoritma öğretimin yapıldığı CS0 (Bilgisayar Bilimleri) dersinde Scratch ile oyun ve animasyon tasarımı etkinlikleri yaptırılmıştır. İkinci yılda ise CS0 dersini alan Scratch grubu ve CS0 dersini almayan diğer bir grup programlama eğitimine devam etmişlerdir. İki yıl süren çalışmanın sonunda Scratch' in CS0 dersinde programlamaya karşı tutumu ve programlama öz yeterliliklerini anlamlı şekilde artırdığı, CS1 dersindeki programlama başarılarının da Scratch eğitimi almayan gruba göre daha çok arttığı görülmüştür. Araştırmacılar bu sonuçlara göre Scratch' in programlamanın temeli olabilecek bir derste öğretilmesi gerektiğini vurgulamışlardır. Ancak bu çalışmada deney grubunun programlama dersinden önce Scratch ile oyun tasarımı etkinlikleri yapması, kontrol grubunun ise hiçbir programlama deneyimi gerçekleştirilmemesi bir sınırlılık olarak ele alınabilir.

Genç ve Karakuş (2011) çalışmalarında öğrencilerin Eğitimde Bilgisayar Oyunları tasarımı dersinde matematiksel ve kompütasyonel becerilerini geliştirmek amacıyla Scratch programı kullanarak öğrenciler ile oyun tasarım etkinlikleri yapmıştır. Çalışmaya Fırat Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi (BÖTE) bölümünde eğitim gören 109 öğrenci katılmıştır. Öğrencilerden anket ve görüşme yoluyla veriler toplanmıştır. Çalışma sonunda öğrenciler Scratch ile oyun tasarım etkinliklerini beğendikleri, metinsel tabanlı programlamaya göre Scratch'in daha kolay olduğu, temel programlama dersleri için uygun fakat ileri programlama dersleri için uygun olmadığı şeklinde görüşte bulunmuşlardır. Ayrıca nitel verilerde ortaya çıkan sonuç ise öğrencilerin tasarım etkinliklerinde etkin olmaktan mutlu oldukları ve derse karşı motivasyonlarının yüksek olduğudur.

Westcott (2008) üniversite öğrencileri ile yaptığı çalışmasında Scratch ile programlama öğretiminin öğrencilerin C++ programlama dilleri başarısına olan etkisini incelemiştir. Kontrol grubunda programlama öğretimi C++ kodlarının öğretimi ve problem çözümü ile gerçekleşirken, deney grubunda C++ kodlarının öğretimi Scratch ile oyun tasarımı etkinliklerine entegre edilerek yapılmıştır. Deney grubunda öğrenciler

C++ problemlerinin benzeri oyunlar tasarlamışlar ve aynı oyunun C++ da yer alan kod karşılığını da öğrenmişlerdir. Araştırma sırasında öğrencilere değişken tanımlama gibi temel kavramların yer aldığı (Test 1), karar verme yapılarının yer aldığı (Test2 - if -else; Test 3 - Switch Case), döngü yapılarının yer aldığı (Test 4) ve tüm yapıların yer aldığı (Test 5) uygulanmıştır. Ayrıca tüm testlerde yalnızca C++ kodlarına değil algoritma problemlerine de yer verilmiştir. Çalışma sonunda değişken tanımlama gibi temel yapıların yer aldığı test sonuçları arasında farklılık çıkmazken; karar verme ve döngü yapıların yer aldığı testlerde ve ayrıca tüm yapıların yer aldığı testlerde deney grubu lehine anlamlı farklılık bulunmuştur. Çalışma sonucunda Scratch ile oyun tasarımı etkinliklerinin öğrencilerin algoritmik düşünme becerilerini geliştirdiğini ve bunu da bir programlama diline aktarabildiklerini vurgulamıştır.

Malan ve Leitner (2007) Harvard Üniversitesi yaz okulu kapsamında bilgisayar bilimleri dersinde Scratch programı öğretimi gerçekleştirmişlerdir. Dönem sonunda Scratch programının önceden programlama deneyimleri olanların ders başarısına etkisi incelenmiş ve çalışma sonunda Scratch'in bilgisayar bilimlerine ilişkin başarı ve tutumu artırdığı sonucuna ulaşılmıştır.

Çağıltay (2007) mühendislik öğrencileri ile yaptığı çalışmasında oyun tasarımının öğrencilerin üst düzey programlama dilleri dersindeki başarısına etkisini incelemiştir. Bu amaçla üst düzey programlama dersinde; bir önceki yıl oyun tasarım dersini alan öğrenciler ile almayan öğrencilerin ders sonundaki başarısını karşılaştırmıştır. Üst düzey programlama dersi sonunda öğrencilerin sınavlar ve projelerden aldığı puanlar ders başarı notu olarak değerlendirilmiştir. Yapılan analizler sonucunda oyun tasarım dersini alan öğrencilerin üst düzey programlama dilleri dersi başarısının almayan öğrencilere göre daha yüksek olduğu ve istatistiksel olarak anlamlı farklılık olduğu sonucuna ulaşılmıştır. Ayrıca oyun tasarım dersi sonunda yapılmış olan öğrenci görüşmelerinde oyun tasarımının, öğrencilerde süreç yönetme, karşılaşılan güçlüklerin üstesinden gelme gibi problem çözme becerilerini geliştirdiği; araştırma yapma, hataları ayıklama, deneme ve düzeltme gibi bağımsız öğrenmeyi desteklediği ve yaparak öğrenme olanağı sağladığı sonuçlarına ulaşılmıştır.

Koorsse, Ciliers ve Calitz (2015) lise öğrencileri ile yaptıkları çalışmada programlamaya yardımcı araçların öğrencilerin programlama kavramlarını öğrenmelerine etkisini incelemişlerdir. Bu amaçla Delphi programlama diline destek

amaçlı Scratch, B# ve Robomind programlarını bir yıl boyunca üç farklı deney grubunda uygulamıştır. Kontrol grubunda ise yalnızca Delphi programlama dilleri eğitimi gerçekleştirilmiştir. Araştırma sonunda üç programlama dilinin öğrencilerin programlama başarılarını artırdığı ancak kontrol grubu ile arasında anlamlı bir fark olmadığı görülmüştür. Ancak her üç programlama dili temel yapılar boyutunda incelendiğinde, döngü kavramı (for- repeat) başarıları açısından deney grubu lehine anlamlı fark olduğu görülmüştür. Öğrencilerin araçların kullanışlılığına ilişkin için görüşleri incelendiğinde; genel anlamda tüm araçları programlama kavramlarını anlama açısından kullanışlı gördüklerini, ayrıca Scratch programını temel yapıları anlama açısından kolaylaştırıcı olarak gördüklerini ifade etmişlerdir.

Nikou ve Economides (2014) lise öğrencileri ile yaptığı çalışmasında Scratch ve API Inventor görsel programlama araçlarının öğrencilerin programlamaya ilişkin motivasyonlarına etkisini incelemiştir. Çalışma sonunda her iki aracında öğrencilerin programlama dersinde içsel hedef yönelimi, görev değeri, öğrenme kontrol inancı ve öz yeterlik algısı motivasyonlarını artırdığı ancak dışsal motivasyonu değiştirmediği sonucuna ulaşılmıştır. Ayrıca çalışmada her iki aracın öğrencilerin programlamaya karşı olan ilgilerini artırdığı görülmüştür.

Giordano ve Mairona (2014) yaptıkları durum çalışmasında bir yıl boyunca lise öğrencilerinin programlama dersinde Scratch ve C programlama dili kullanma sürecini incelemiştir. Süreçte öğrenciler öncelikle ilk haftalarda temel programlama mantığını öğrenmek için yalnızca Scratch ile basit oyunlar ve programlar tasarlamışlar, daha sonra hem Scratch hem de C programlama dillerini paralel kullanmışlardır. Ayrıca Scratch kullanarak tasarladıkları basit programları, C programlama dilinde kod kullanarak yapmışlar ve sürecin son haftalarında yalnızca C programlama dilini kullanarak programlar oluşturmuşlardır. Çalışma boyunca öğrencilere dört kez sınav uygulanmış ve başarı gelişimleri incelenmiştir. Çalışma sonunda öğrencilerin programlama mantığını ve temel programlama yapılarını öğrenme süreçlerinin hem algoritma hem de C kodları bağlamında sürekli gelişme gösterdiği görülmüştür.

Tekerek ve Altan (2014) 6. sınıfta eğitim gören 60 öğrenci ile yaptığı çalışmasında Scratch'in algoritma (temel programlama) öğretimine etkisini incelemiştir. Araştırmada öğrenciler deney ve kontrol grubu olarak ayrılmış ve deney grubunda algoritma öğretimi, oyun tasarımı ile kontrol grubunda ise düz anlatım yöntemiyle

gerçekleştirilmiştir. Çalışma sonunda her iki grupta da öğrenci başarısının arttığı görülmüş; sontestler karşılaştırıldığında ise deney ve kontrol grubu arasında bir farklılık bulunmamıştır. Ayrıca cinsiyet değişkeni açısından kontrol ve deney gruplarının sontest puanları arasında anlamlı bir farklılık bulunmamıştır.

Kalelioğlu ve Gülbahar (2014) ilköğretim öğrencileri ile yaptığı çalışmasında Scratch ile programlama öğretiminin öğrencilerin problem çözme becerilerine olan etkisini incelemişlerdir. Çalışmada öğrenciler beş haftalık sürede basit programlar ve oyunlar tasarlamışlardır. Çalışma sonunda öğrencilerle Scratch ile programlama öğretimi hakkında odak grup görüşmeleri yapılmıştır. Nicel veriler incelendiğinde öğrencilerin öntest ve sontest problem çözme puanları arasında; öz güven, öz denetim ve kaçınma alt boyutları da dâhil olmak üzere anlamlı bir farklılık olmadığı sonucuna ulaşılmıştır. Nitel veriler incelendiğinde öğrencilerin Scratch ile programlama sürecini sevdiği, programlama yapmaya devam etme ve geliştirme isteklerinin olduğu belirlenmiştir.

Wyffles, Martens ve Lemmens (2014) ilköğretim öğrencileri ile yaptığı çalışmasında üç farklı grafik tabanlı programlama aracını öğrencilerin görüşleri bağlamında karşılaştırmıştır. Süreç içerisinde bir grup görsel yönü daha yüksek olan Scratch, bir grup Java kodlarını kullanma olanağı sağlayan Greenfoot, diğer grup ise robot programlama olanağı veren Dwengo programını kullanmışlardır. Çalışma sonunda öğrencilere programlama sürecini nasıl buldukları ve genel programlamaya ilişkin tutumlarına ilişkin görüşlerine başvurulmuştur. Buna göre Scratch grubu; programlama sürecini eğlenceli ve ilginç, programlamayı ise beklediklerinden daha kolay olarak nitelendirmişlerdir. Dwengo ve Grenfoot grubu ise programlama sürecini ilginç ve programlamayı beklediklerinden zor bulduklarını ifade etmişlerdir.

Hsu (2014) ilköğretim öğrencileri ile yaptığı çalışmada Scratch ile oyun tasarımı yapan öğrencilerin programlama yapılarını kavrama ve oyunlarında kullanma durumunu cinsiyete göre incelemiştir. Bu amaçla 46 öğrenci Scratch ile oyunlar tasarlamışlardır. Çalışma sonunda öğrencilere programlama yapılarını kapsayan bir test uygulanmış ayrıca tasarladıkları oyunlar analiz edilmiştir. Analiz sonucunda kız ve erkek öğrencilerin programlama yapılarını anlamada aralarında farklılık olmadığı, yalnızca sayaç mantığındaki döngü yapılarını kız öğrencilerin daha iyi anladığı ortaya çıkmıştır. Ayrıca oyun analizi sonucunda da kız ve erkek gruplarının tasarladıkları oyunlarda

temel programlama yapılarını içeren blokları benzer oranda kullandıkları, oyunların içerdiği kostüm, sahne, karakter vb. yapıları kullanma açısından da cinsiyetin belirleyici bir özellik olmadığı sonucuna ulaşılmıştır.

Salant, Armonia ve Ben-Aria (2013) lise öğrencileri ile yaptıkları çalışmalarında Scratch kullanarak Bilgisayar Bilimleri ilgili kavramlarının öğretilmesini amaçlamışlardır. İki yıl boyunca haftada iki saat olmak üzere farklı öğrenci gruplarında Scratch ile animasyon ve oyun tasarımı yapılmıştır. Araştırmada öntest, ara test ve sontest olmak üzere nicel veriler; gözlem ve görüşmeler aracılığıyla nitel veriler toplanmıştır. Araştırma sonunda öğrencilerin döngüler ve değişkenler gibi yapıların mantığını kavramakta zorlandığı, ancak şartlı ifadeler, zamanlama, olaylar gibi kavramların mantığını anlamakta zorluk çekmedikleri görülmüştür. Ayrıca öğrenciler Scratch ile tasarım etkinliklerinin zevkli olduğunu ve öğrenmeyi teşvik ettiğini vurgulamışlardır.

Adleberg (2013) Scratch ile oyun tasarımının kız öğrencilerin bilgisayar bilimlerine karşı olumsuz tutumlarına etkisini incelemeyi amaçladığı çalışmasını 98 ortaokul öğrencisiyle gerçekleştirmiştir. Scratch ile oyun tasarımı etkinlikleri sonunda katılımcılar arasında bilgisayar bilimlerine karşı tutum açısından cinsiyet bağlamında anlamlı bir farklılık olmadığı, yapılan gözlemlerle de kız öğrencileri güdüleyen etmenlerin akran onayı, öğretmen ve ebeveyn dönütleri ile Scratch etkinliklerindeki başarıların olduğu bulunmuştur.

Brigss (2013) durum çalışması olarak gerçekleştirdiği araştırmasında üç farklı okulda Scratch ile tasarım etkinliklerinin öğrenciler açısından faydasını incelemiştir. Bu amaçla üç okulda yer alan etkinlikler süresince gözlem, video kaydı ve görüşmeler y gerçekleştirilmiştir. . Çalışma sonunda Scratch ile oyun ve animasyon tasarımının, öğrenenlerin bağımsız öğrenmelerini desteklediği, mantıksal düşünmelerini geliştirdiği ve araştırma için teşvik ettiği sonuçlarına ulaşılmıştır. Bu süreçte öğrencilerin genellikle deneme ve yanılma yoluyla kendilerinin öğrendiği, birçok noktayı araştırdıkları veya akranlarına danıştıkları ortaya çıkmıştır. Ayrıca öğrencilerin süreç içinde oyun geliştirmeyi öğrendikleri ve bu süreç içinde birçok problem ve zorluğun üstesinden gelmek zorunda kaldığı görülmüştür. Öğretmenlerin ise bu süreçte daha çok geri planda kaldığı, bazı gruplarda tasarım konusunda katkıda bulunduğu ve öğrencilerden istek gelmesi durumunda onlara yardım ettiği görülmüştür.

Burke (2012) ortaokul öğrencileri ile yaptığı çalışmasında öğrencilerin Bilgisayar Bilimlerine karşı tutumlarını artırmak, programlama becerilerini geliştirmek, geleneksel okuryazarlık çalışmalarını dijital hikâye tasarımı ile farklılaştırmak ve öğrencilerin teknoloji okuryazarlığını artırmak için Scratch ile dijital hikâye ve oyun tasarımı etkinlikleri gerçekleştirmiştir. Araştırma sonunda öğrencilerin Scratch etkinliklerini informal öğrenme süreci olarak algıladıkları, döngü, karar verme gibi programlama yapılarını öğrendikleri, hikâye oluşturma ve eş zamanlama gibi dijital hikâye becerilerinin geliştiği sonuçlarına ulaşılmıştır.

Fadjo (2012) yaptığı çalışmasında somutlaştırma amaçlı Scratch kullanımının ortaokul öğrencilerinin kompütasyonel becerilerine etkisini incelemek için iki farklı çalışma yapmıştır. Birinci etkinlik çerçevesinde 56 ortaokul öğrencisi Scratch ile dijital hikâye tasarımı yapmışlar, ikinci etkinlik çerçevesinde ise 78 ortaokul öğrencisi Scratch ile oyun tasarımı etkinlikleri yapmışlardır. Çalışma sonunda birinci etkinlikte öğrencilerin daha çok eş zamanlama gibi kompütasyonel becerilerinin geliştiği, ikinci etkinlikte ise daha çok hata ayıklama, koşullu ifadeler, olaylar gibi programlama kavramlarını içeren kompütasyonel becerilerinin geliştiği görülmüştür. Bu sonuçlara göre Fadjo (2012) soyut olan kompütasyonel kavramların öğretiminde Scratch' in etkili olduğunu vurgulamıştır.

Osman, Zakaria, Loke ve Downe (2012) programlama eğitimi alan 591 lise öğrencisi ile yaptığı çalışmasında Visual Basic Express, Scratch ve PyGame programlarını motivasyon ve başarıya ulaştırma açısından karşılaştırmıştır. Öğrencilerden programlama sürecinde kullandıkları araçlar ile ilgili mükemmeliyetçilik düzeylerini belirtmeleri istenmiştir. Araştırma sonunda hem düşük hem de yüksek mükemmeliyetçilik düzeyine sahip gruplar Scratch ve PyGame araçlarını, Visual Basic Express yazılımına göre motivasyonu artırma ve başarıya ulaştırma açısından çok daha etkili bulmuşlardır. Bunu da Visual Basic Express programının kod tabanlı çalışması, Scratch ve PyGame araçlarının ise hem görsel olması hem de oyun tasarımına olanak vermesinden kaynaklandığını belirtmişlerdir. Araştırmacılar Scratch ve PyGame gibi görsel programlama araçlarının özellikle temel programlama düzeyindeki derslerde tercih edilebileceğini vurgulamışlardır.

Baytak ve Land (2011) yaptıkları durum çalışmasında kız öğrencilerin programlama becerilerini oyun tasarımı ile geliştirmeyi amaçlamıştır. Bu bağlamda

oyun tasarımı ile ilgili hiçbir becerileri olmayan ilköğretim öğrencileri farklı zaman dilimlerinde Scratch ile oyun tasarımı yapmışlardır. Süreç boyunca gözlemler yapılmış ve öğrencilerin yapmış oldukları oyunlar incelenmiştir. Çalışma sonunda tüm öğrencilerin oyun tasarım sürecini başarılı geçirdikleri, programlama kavramlarını (döngü, koşul, olay vb.) oyunlarında kullandıkları gözlenmiştir. Ayrıca kız öğrencilerin erkek öğrencilere göre oyunlarında, daha iyi organize edilmiş sayıca daha çok komut bloğu ve programlama yapılarını kullandığı ortaya çıkmıştır. Araştırmacılar bu durumun iyi organize edilmiş ve farklı kod blokları içeren programların, algoritması iyi düzenlenmiş ve daha düzgün çalışan programlar sonucunda ortaya çıktığını ifade etmişlerdir. Araştırmada ayrıca kız öğrencilerin en az erkek öğrenciler kadar oyun tasarım sürecini benimsediği, öğrendiği ve süreçte etkin olarak yer aldığı ve bu durumu yadırgamadıkları da ulaşılan önemli sonuçlardandır.

Alimisi ve Wniters (2010) 12-13 yaş aralığındaki dört öğrenci ile yaptığı küçük ölçekli durum çalışmasında gerçek hayat senaryoları üzerinden Scratch ile programlama yaparak koşullu ifadelerin öğretilmesini amaçlamışlardır. Çalışma süresince öğrenciler ile görüşmeler yapılmıştır. Öğrenciler Scratch' le programlama sürecini eğlenceli bir deneyim olarak ifade etmişler. Ayrıca birçok problemle baş etmek zorunda kaldıklarını, programı test etme aşamasında hata ile ilgili anında dönüt aldıklarını ve anında düzelttiklerini, bu durumun programı başarıya ulaştırmada etkili olduğunu ifade etmişlerdir. Ayrıca katılımcılar verilen ve koşullu ifadeleri kullanmayı gerektiren senaryoları başarılı şekilde Scratch ile programlaştırabilmişlerdir.

Gülmez (2009) ilköğretim öğrencileri ile yaptığı çalışmasında programlama öğretiminde görselleştirme araçları kullanımının öğrenci başarı ve motivasyonuna olan etkisini incelemiştir. Çalışmasında kontrol grubunda akış diyagramı yazılımı kullanmış, deney grubunda ise Scratch programı kullanmıştır. Çalışma sonunda öğrencilerin motivasyonları açısından anlamlı farklılık bulunmazken, algoritma başarıları açısından döngü ve koşullu ifadeler konusunda deney grubu lehine farklılık olduğu ortaya çıkmıştır.

Maloney, Peppler, Kafai, Resnick ve Rusk (2008) yaptıkları çalışmalarında "Bilgisayar Kulübü" etkinlikleri altında yaşları 8-18 arasında değişen gençler ile bir yıl boyunca okul dışı etkinlik olarak Scratch ile oyun tasarımı gerçekleştirmişlerdir. Çalışma kapsamında yapılan oyunlar programlama yapılarını (döngü, boolen mantığı,

koşullu ifadeler ve değişkenler vb.) içermesi açısından incelenmiş, ayrıca süreç ile ilgili öğrenci görüşlerine başvurulmuştur. Süreçte yapılan oyunlar incelendiğinde; %52'sinin döngü, %26'sının koşullu ifadeler, %10'unun boolen mantığı ve %9'unun değişkenleri içerdiği görülmüştür. Araştırmacılar oyun içerisinde programlama yapıları kullanılmasının programlama mantığı açısından önemli bir sonuç olduğunu vurgulamışlardır. Ayrıca öğrenci görüşleri incelendiğinde öğrencilerin Scratch'i eğlenceli buldukları, programlama ve matematikle ilişkilendirdikleri görülmüştür. Araştırmacılar Scratch'in programlama için yardımcı bir araç olabileceğini ve programlaya giriş derslerinde kullanılması gerektiğini vurgulamışlardır.

Malan ve Leitner (2007) yaptıkları çalışmalarında programlama deneyimi olmayan üniversite öğrencileri ile Harvard Üniversitesi yaz okulu kapsamında programlamaya giriş ve Java programlama diline geçiş amaçlı Scratch ile oyun ve animasyon tasarımı yapmışlardır. Çalışma sonunda öğrencilerle yapılan görüşmelerde; öğrencilerin Scratch'i eğlenceli buldukları, oyun yapmanın zevkli olduğu, karmaşık kodlar içermediği için kolay olduğunu belirlenmiştir. Java programla dili başarınıza Scratch etki etti mi?" sorusuna ise yapılan anket sonunda; öğrencilerin %76'sı olumlu etki etti derken, %8'i olumsuz etki ve %16'sı ise hiçbir etki etmedi cevabını vermiştir.

Programlama Öğretiminde Diğer Tasarım Araçlarının Kullanımı

Yapılan çalışmalar incelendiğinde Scratch programına benzer araçlarında programlama öğretiminde akademik başarı ve motivasyonu artırmak için kullanıldığı görülmüştür. Bu başlık altında programlama öğretiminde; Scratch ile benzer özellik gösteren görselleştiriciler ve tasarım araçlarının kullanıldığı çalışmalar ele alınmış, üniversite, lise ve ilköğretim düzeyindeki çalışmalar birlikte incelenmiştir.

Howland ve Good (2015) ortaokul öğrencileri ile yaptıkları çalışmada oyun tasarım sürecinin öğrencilerin kompütasyonel (programlama yapıları; çıktı, koşul, iç içe yapılar vb.) kavramları öğrenmesine etkisini incelemiştir. Öğrenciler bilgisayar derslerinde Flip adlı programlama aracı ile üç boyutlu rol yapma oyunu tasarlamışlardır. Çalışmada kompütasyonel kavramları ölçen soruların yer aldığı problemler öntest-sontest olarak uygulanmış ve oluşturulan oyunlar incelenmiştir. Araştırma sonunda oyun tasarım sürecinin öğrencilerin kompütasyonel becerilerini geliştirdiği (kavram ve programlama becerisi), cinsiyet değişkenine göre kızların daha başarılı olduğu ve daha karmaşık kodlar kullandıkları ortaya çıkmıştır.

Akcaoğlu ve Koehler (2014) "Oyun tasarımı kursu" altında bir grup ortaokul öğrencisi ile gerçekleştirdikleri çalışmalarında Microsoft Kodu programlama dilini kullanarak oyun tasarımı yapmışlardır. Araştırmada deney ve kontrol grupları oluşturulmuş ve oyun tasarımı etkinlilerinin öğrencilerin problem çözme becerilerine ve motivasyonlarına olan etkisi incelenmiştir. Araştırma sonunda öğrencilerin sistem analizi ve tasarımı, karar verme ve sorun giderme alt boyutlarında ayrıca problem çözme becerileri ve motivasyon düzeylerinin deney grubu lehine anlamlı farklılık olduğu belirlenmiştir.

Claypool (2013) yaptığı çalışmada DragonFly isimli oyun motorunu kullanarak üniversite öğrencileri ile programlama dersinde oyun tasarımı yapmışlardır. Bu kapsamda öğrenciler oyun motorunu kullanarak kodlama yapma yoluyla oyun tasarlamışlardır. Çalışma sonunda öğrencilerin programlama sürecinde oyun tasarım etkinliklerini eğlenceli buldukları, temel ve ileri düzey programlama yapıları ile oyun tasarım sürecini ve mantığını öğrendikleri sonucuna ulaşılmıştır.

Denner, Werner ve Ortiz (2012) ortaokulda eğitim gören 59 kız öğrenci ile yaptığı çalışmada öğrencilerle okuldan sonra haftada iki saat olmak üzere altı hafta boyunca Stagecast Creator programı ile oyun tasarımı yapmışlardır. Çalışma sonunda öğrencilerin yapmış olduğu 108 oyun incelenmiş ve öğrencilerin oyunlarda orta düzeyde karmaşık kod yapısı kullandıkları, oyunların orta düzeyde kullanışlı olduğu ve düşük düzeyde kod düzenlemesine sahip oldukları sonuçlarına ulaşılmıştır. Araştırmacılar Stagecast Creator gibi basit programlar sayesinde hiçbir programlama hazırbulunuşluğu olmayan ortaokul öğrencilerin kolaylıkla oyun tasarlayabileceklerini ve oyun tasarlamanın programlama mantığının öğretilmesi için destek olabileceğini vurgulamışlardır.

Basawapatma, Koh ve Repenning (2010) yaptıkları üç aşamalı çalışmada ortaokul öğrencilerine, üniversite öğrencilerine ve ortaokul öğretmenlerine oyun tasarımı ile temel bilgisayar bilimlerinin öğretilmesi amaçlamışlardır. Bu bağlamda katılımcılar AgentSheets aracını kullanarak oyun tasarım etkinlikleri gerçekleştirmişlerdir. Birinci uygulama sonunda yapılan gözlem ve görüşmelerde ortaokul öğrencilerinin sınıf içerisindeki katılımlarının ve bilgisayar bilimlerine olan ilgilerinin arttığı görülmüştür. İkinci uygulamada üniversite öğrencileri tasarladıkları oyunları ortaokul öğrencilerine oynatmışlar ve geribildirim alarak yeniden

düzenlemişlerdir. Uygulama sonunda öğrencilerin programlama becerilerinin ve oyun geliştirme becerilerinin arttığı görülmüştür. Üçüncü uygulamada oyun tasarımı yapan öğretmenler, oyun tasarım sürecinin programlama öğretiminde etkili bir öğretme yöntemi olabileceğini ifade etmişlerdir.

Carbonaro, Szafron, Cutumisu ve Schaeffer (2010) lise öğrencileri ile yaptıkları çalışmalarında ScriptEase aracı kullanarak oyun tasarım etkinlikleri gerçekleştirmişlerdir. Çalışma sonunda oyun tasarım etkinliklerinin; öğrencilerin üst düzey düşünme becerilerini ve soyut düşünme becerilerini geliştirdiği, bilgisayar bilimlerine giriş için eğlenceli olduğu sonucuna ulaşılmıştır. Ayrıca kız öğrencilerin erkek öğrenciler kadar oyun tasarım etkinliklerini eğlenceli buldukları ve hem üst düzey düşünme hem de soyut düşünme becerileri açısından erkeklerle aralarında farklılık olmadığı belirlenmiştir.

Akçay (2009) yaptığı çalışmasında ilköğretim 4. ve 5. sınıflarda yer alan bilgisayar dersinde Small Basic programlama dili kullanımın öğrenci motivasyonundaki etkisi ve bilgisayar dersinde Small Basic kullanımına ilişkin öğretmenler ve öğrencilerin görüşlerini incelemiştir. Çalışma sonunda bilgisayar derslerinde Small Basic kullanımının öğrencilerin derse olan motivasyonlarını artırdığı, öğretmenlerin bu programın derslerde kullanımının faydalı olduğuna ilişkin olumlu görüş bildirdikleri sonucuna ulaşılmıştır.

Al-Bow ve diğerlerinin (2009) yaptıkları çalışmada, oyun tasarımı ile lise öğrencilerinin ve öğretmenlerinin bilgisayar bilimlerine ve programlaya olan ilgilerini artırmayı amaçlamışlardır. Bu amaç doğrultusunda öğrenciler ve öğretmenler temel programlama yapılarını öğrenmek amacıyla oyunlar tasarlamışlardır. Çalışma sonunda hem öğretmenlerin hem de öğrencilerin programlamaya karşı özgüvenlerinin arttığı görülmüştür. Ayrıca öğrenciler oyun tasarlama yönteminin programlama öğretimine olumlu bir etkisinin olduğunu belirtmişlerdir.

Kurkovsky (2009) yaptığı durum çalışmasında mobil oyun tasarımın temel programlama yapılarını öğrenme sürecine etkisini incelemiştir. Çalışma sırasında katılımcılar ile görüşmeler yapmış ve süreci gözlemlemiştir. Çalışma sonunda mobil oyun tasarlama sürecinin öğrencilerin temel programlama yapılarını öğrenmelerinde ve programlama öğrenme sürecinde etkili bir motivasyon kaynağı olduğu sonucuna ulaşmıştır.

Robertson ve Howells (2008) ortaokul öğrencileri ile yaptıkları nitel çalışmada hazır bir oyun motoru kullanarak bilgisayar bilimleri kapsamında öğrenciler ile sekiz hafta boyunca oyun tasarımı gerçekleştirmiştir. Bu süreç içerisinde öğrenciler oyunu yaparken öğretmenden bağımsız, grup arkadaşlarına bağımlı biçimde oyun tasarımı yapmışlardır. Çalışma sonunda oyun tasarım sürecinin heyecan verici olduğu, programlama ve temel bilgisayar kavramlarını öğrenmede etkili olduğu, öğrenme motivasyonunu artırdığı, başarıya ulaştıran bir etmen olduğu ve öğrencilere öğrendiklerini yeni durumlara uyarlama olanağı verdiği sonuçlarına ulaşmışlardır.

Dönmez (2008) BÖTE bölümünde eğitim gören 43 öğrenci ile yaptığı çalışmasında, algoritma öğretimi için geliştirdiği görselleştiricinin öğrenci başarısına etkisini araştırmış ve bu süreçle ilgili öğrenci görüşlerini incelemiştir. Bu kapsamda Eğitimde Bilişim Teknolojileri-II dersini alan 43 öğrenci deney ve kontrol grubu olarak ikiye ayrılmış, kontrol grubunda yalnızca Visiual Basic programlama uygulamaları, deney grubunda ise buna ek olarak geliştirilen görselleştirici uygulanmıştır. Çalışma sonunda öğrenci başarıları açısından bir farklılık görülmemiştir. Bunun yanında öğrencilerin geliştirilen görselleştirici araçla ilgili düşüncelerinin olumlu olduğu görülmüştür.

Bishop - Clark, Courte ve Howard (2007) yaptıkları çalışmalarında programlama kavramlarının öğretildiği temel düzey bir derste araştırma kapsamında 154 öğrenciye 2,5 hafta boyunca Alice yazılımı ile görsel öğeler kullanarak animasyon ve oyun benzeri programlar yaptırmışlardır. Süreç sonunda programlama özgüven ve programlama kavramları öntest - sontest puanları karşılaştırıldığında öğrencilerin programlamaya ilişkin özgüvenlerinin ve programlama kavramlarını anlama düzeylerinin arttığı görülmüştür. Aynı zamanda nitel verilerinde bu sonucu desteklediği, öğrencilerin programlamadan keyif aldıkları görülmüştür.

Denner (2007) ortaokulda okuyan 126 öğrenci ile yaptığı çalışmasında oyun tasarımı ile kızların bilgisayar bilimlerine olan ilgisini ve ileride bilgisayar alanında kariyer yapma eğilimlerini artırmayı amaçlamıştır. Çalışma kapsamında "Kızlar Oyun Tasarlıyor" adlı etkinlikle öğrencilerle 12 hafta boyunca Flash programı ile oyun tasarımı gerçekleştirilmiştir. Çalışma sonunda kız öğrencilerin teknolojiye olan ilgilerinin arttığı ve ileride bilgisayar ile ilgili bir işte çalışmayı düşündükleri görülmüştür.

Kelleher (2006) kız öğrencilerin programlamaya karşı olumsuz tutumunu değiştirmek amacıyla yaptığı çalışmasında ortaokul öğrencilerinin Storytelling Alice yazılımını kullanarak üç boyutlu hikâye tasarlama süreçlerini incelemiştir. Çalışmada birinci grup programlama sürecinde Alice yazılımının hikâye tasarımına elverişli olmayan sürümünü kullanmış, diğer grup ise hikâye tasarımına elverişli sürümünü kullanmıştır. Çalışma sonunda Storytelling Alice yazılımını kullanan grubun programlama sürecinde daha çok zaman harcadıkları ve programlamaya ilişkin motivasyonlarının daha yüksek olduğunu bulmuştur.

Yucel, Zupko ve Seif El-Nasr (2006) yaptıkları çalışmalarında yeniden düzenlenebilir oyunlar kullanarak (oyun modifikasyonu) kız öğrencilerin Bilgi ve İletişim Teknolojileri (BİT) becerilerini ve motivasyonlarını artırmayı amaçlamıştır. Bu bağlamda kız öğrenciler hafta sonları Warcraft 3 isimli oyunu yeniden düzenlemişlerdir. Hem nitel (gözlem) hem de nicel (anketler ve oyunlar) veriler yeniden oyun düzenleme sürecinin kız öğrencilerin BİT becerilerini ve motivasyonlarını artırdığını göstermiştir. Bunun yanı sıra yeniden oyun düzenleme süreci ile öğrenciler arası etkileşim oluşmuş ve öğrenciler teknolojiyi tüketen konumdan üreten konuma gelmişlerdir.

Cooper ve arkadaşları (2003) üniversite öğrencileriyle yaptıkları çalışmada Bilgisayar Bilimleri 1 dersinde, programlama temeli olmayan bir grup öğrenciye Alice yazılımı ile etkileşimli animasyonlar hazırlatmışlardır. Ders sonunda Alice yazılımını kullananların ders başarılarının kullanmayan gruba göre daha iyi olduğu sonucuna ulaşılmıştır. Ayrıca Alice yazılımını kullanan öğrencilerin bilgisayar bilimleri alanında eğitim almaya devam etme eğiliminin arttığı belirlenmiştir.

Kafai ve arkadaşları, (1998) oyun tasarımının öğrencilerin öğrenme süreçlerine etkisini incelemiştir. Çalışmada Papert (1991)' in yapılandırmacı kuramına dayanarak ilköğretim 4. sınıf öğrencilerine kesirler konusuyla ilgili oyun tasarımı yaptırılmıştır. Tasarım sırasında öğrenciler hikâyeyi oluşturmuşlar, içeriği öğrenmişler ve oyunu programlamışlardır. Çalışma sonunda öğrencilerin kendi bağımsız öğrenme süreçlerini kontrol ettiklerini ve kendi öğrenme ortamlarını inşa ettikleri vurgulanmıştır.

Oyun Tasarımının İçerik Öğretiminde Kullanımı

Yapılan çalışmalar incelendiğinde Scratch ya da benzeri araçlar ile oyun ya da animasyon tasarımının yalnızca bilgisayar bilimleri ve programlama öğretiminde değil farklı disiplinlerde de akademik başarıyı ve motivasyonu artırmak için kullanıldığı

görülmüştür. Bu başlık altında oyun tasarımının farklı disiplinlerde akademik başarı ve motivasyonu artırmaya yönelik yapılan çalışmalar yer almaktadır.

Ke (2014) 64 ortaokul öğrencisi ile yaptığı çalışmasında oyun tasarımın matematik öğretimine etkisini incelemiştir. Çalışma sırasında öğrenciler Scratch kullanarak matematik oyunları tasarlamışlardır. Süreç boyunca öğrencilerden görüşme, gözlem ve anketle veriler elde edilmiştir. Çalışma sonunda hem nicel hem de nitel veriler oyun tasarlama sürecinin; öğrencilerin matematiğe karşı tutumlarını daha olumlu hale getirdiği ve içeriğin edinilmesinde etkili olduğu sonuçlarına ulaşılmıştır. Bunun yanında öğrencilerin programlama zorluklarıyla uğraşmalarından dolayı, oyun tasarımının bir müddet sonra matematik öğretiminin önüne geçtiği görülmüştür. Bu yüzden öğrencilerin programlama ve oyun tasarımını matematikten daha çok öğrendiği ortaya çıkmıştır. Araştırmacı bu nedenle eğer içerik öğretimi söz konusu ise oyun tasarımının yalnızca bir araç olması gerektiğini; öğretilmek istenen içeriğin tasarlanan oyun ile çok iyi ilişkilendirilmesi gerektiğini vurgulamıştır.

Kim, Chung ve Yu (2013) ortaokul öğrencileri ile yaptıkları çalışmada programlama eğitimi kapsamında Scratch ile oyun ve dijital hikâye tasarımının öğrencilerin yaratıcı problem çözme becerilerine olan etkisini incelemişlerdir. Bu bağlamda hem normal çocuklarda hem de üstün yetenekli çocuklar için ayrı ayrı Scratch ile tasarım etkinlikleri gerçekleştirmişlerdir. Çalışma sonunda normal çocuklar açısından yaratıcı problem çözme becerileri incelendiğinde deney ve kontrol grubu arasında anlamlı farklılık görülmezken; üstün yetenekli çocuklar açısından deney grubu lehine anlamlı farklılık ortaya çıkmıştır.

Navarrete (2013) yaptığı durum çalışmasında ortaokul öğrencilerinin yaratıcı oyun tasarımı etkinlikleri ile ilgili görüşlerini incelemiştir. Çalışma süresince öğrenciler Flash programı kullanarak sosyal sorunlar temalı dijital oyunlar tasarlamışlardır. Tasarım sırasında yalnızca tasarım ve programlama yapmamışlar aynı zamanda ilgili konuyu araştırmışlardır. Çalışma sonunda öğrencilerin oyun tasarım etkinliklerini tatmin edici ve ilgi çekici buldukları, oyun tasarım etkinliklerinin bilgiyi yapılandırma, ürüne dönüştürme ve anlamlı öğrenme olanağı sağladığı sonuçlarına ulaşılmıştır.

Hava (2012) ilköğretim 4. sınıf öğrencileri ile yaptığı çalışmada oyun tasarımının öğrencilerin matematik başarısına etkisini incelemiştir. Bu bağlamda deney grubunu oluşturan öğrenciler "Oyun yap ve Oyna" isimli uygulamayla kesirler

konusuyla ilgili oyunlar tasarlamışlar, kontrol grubu ise uygulamada yer alan kesirler konusuyla ilgili oyunları oynamışlardır. Araştırma sonunda her iki grupta da akademik başarı artış göstermiş ancak sontestler karşılaştırıldığında oyun tasarlayan grup ile yalnızca oyun oynayan grup arasında akademik başarıları açısından bir farklılık bulunamamıştır.

Baytak ve Land (2011) ilköğretim 5. sınıf öğrencileri ile yaptığı durum çalışmasında öğrencilerin çevre bilimleri konusunda Scratch ile oyun tasarımı süreçlerini incelemişlerdir. Araştırma sonunda öğrencilerin planlama, tasarlama, test etme ve paylaşma bağlamında oyun yapma gibi programlama becerilerinin geliştiği, konu ile ilgili informal bilgilerinin arttığı gözlemlenmiştir.

Calder (2010) matematiksel kavramların öğretiminde Scratch ile programlama sürecini incelemiş ve bu doğrultuda öğrencileri gözlemleyerek, onlarla görüşme yapmıştır. Çalışma sonunda Scratch'in matematik kavramların öğrenilmesinde güdüleyici olduğu ve problem çözme süreci açısından matematiksel düşüncenin geliştirilmesinde Scratch programının kullanılması gerektiği vurgulanmıştır.

Peppler ve Kafai (2007) yaptıkları etnografik çalışmada ortaokul öğrencilerinin okul dışı etkinlik olarak Scratch kullanarak medya tasarım sürecini incelemişlerdir. Öğrenciler Scratch kullanarak bilgisayar kulübü etkinlikleri altında oyunlar ve hikâyeler tasarlamışlardır. İnformal öğrenme sürecinin gerçekleştiği çalışmada öğrencilerin yaratıcı tasarım sürecinde nasıl medya oluşturdukları ve öğrenme davranışları incelenmiştir. Yapılan gözlemler sonucunda öğrencilerin Scratch ile tasarım yaparken öğrenme sürecinin bir parçası oldukları, medya üretimden zevk aldıkları ve yaratıcı davranışlar sergiledikleri ortaya çıkmıştır.

Alanyazın incelendiğinde programlama öğretiminde oyun tasarımı ile ilgili birçok çalışmanın yer aldığı bu çalışmaların genellikle deneysel çalışmalar olduğu, bunun yanında durum çalışmalarında da yer aldığı görülmektedir. Bu araştırmalarda oyun tasarım etkinlikleri için birçok farklı görselleştirici programın kullanıldığı, ancak son zamanlarda özellikle programlamayı sevdirmek amaçlı Scratch aracının kullanıldığı çalışmalar yapıldığı görülmektedir. Programlama öğretiminde Scratch kullanımına ilişkin çalışmaların genellikle lise ve alt kademelerde yapıldığı az da olsa üniversite düzeyinde ve yetişkinlerle yapılan çalışmaların da yer aldığı görülmektedir. Bununla birlikte çalışmalarda genellikle Scratch ile oyun tasarımını etkinliklerinin programlama

başarısına etkisi incelenmiş, bazı çalışmalarda ise başarının yanında motivasyona etkisi de incelenmiştir (Gülmez, 2009). Ancak alan yazında Scratch ile oyun tasarımı etkinliklerinde elde edilen deneyim ve bilgilerin yeni bir programlama dili öğretiminde devam edip etmediği ya da transfer edilip edilmediğiyle ilgili yeterli çalışma olmadığı görülmektedir (Ozoran ve diğerleri, 2012; Rivzi ve diğerleri, 2011; Wescott, 2008).

Amaç

Bu araştırmanın temel amacı Scratch ile programlama öğretiminin bilişim teknolojileri öğretmen adaylarının motivasyon ve programlama başarısına etkisini incelemektir. Bu amaç doğrultusunda aşağıdaki araştırma sorularına yanıt aranmıştır:

1. Katılımcıların programlama dili öğrenmeye yönelik motivasyonlarını ve akademik başarılarını artırmada Scratch ile öğrenen grup ile mevcut ders programına göre öğretimin gerçekleştirildiği grup arasında anlamlı farklılık var mıdır?
2. Katılımcıların aldıkları programlama eğitimleri hakkındaki görüşleri nelerdir?

Önem

Günümüzde bilgi ve iletişim sektöründe özellikle yazılım geliştirme alanında yetişmiş insan gücüne gereksinim duyulmaktadır. Ancak programlama eğitiminde yaşanan sorunlar özellikle başlangıç düzeyindeki programcıların programlamaya karşı olumsuz tutum geliştirmelerine ve programlama derslerinden başarısız olmalarına neden olmakta, hatta birçoğu da dersi ya da ilgili bölümü bırakmaktadır (Kinnunen ve Malmi, 2008). Bu zorluğun giderilmesi için başlangıç düzeyindeki programlama derslerinde temel programlama yapılarının ve programlama mantığının etkili şekilde öğretilmesi gerekmektedir. Eğitimciler bu zorluğun giderilmesi için farklı görselleştiriciler kullanmaktadırlar. Bu görselleştiricilerin bazıları yalnızca programlama sürecini görselleştirerek çıktı sağlarken, bazıları da oyun ve hikâye gibi çokluortam tasarımına olanak vermektedir. Programlama öğretiminde oyun veya hikâye tasarımı, öğrenenlerin kendi öğrenme süreçlerini oluşturduğu, aktif katılımın sağlandığı zevkli ve eğlenceli bir öğrenme ortamı oluşturmaktadır (Gee, 2005; Rajaravivarmasi, 2005). Bir görselleştirici araç olan Scratch basit arayüzü ve tasarım olanağı sağlamasıyla oyun tasarımı ile programlama öğretimi sürecinde kullanılmaktadır (Malan ve Leitner, 2007). Bundan dolayı Scratch ile oyun tasarım etkinlikleri; öğrenenlerin programlamaya olan ilgisini ve

motivasyonlarını artırabilir, temel programlama sürecini daha iyi kavrayarak programlama mantığını daha iyi öğrenmelerini sağlayabilir.

Bu çalışma, Scratch kullanarak oyun tasarımı ile programlama öğretiminin;

- öğrenci motivasyonuna etkisi konusunda alanyazına katkı sağlayacak olması,
- temel programlama yapıları bağlamında öğrenci başarısına etkisi konusunda alanyazına katkı sağlayacak olması,
- temel programlama eğitiminde edinilen bilgi ve becerilerin farklı bir programlama diline transferinin sağlanması konusunda alanyazına katkı sağlayacak olması,
- sağladığı üstünlükler ve sınırlılıklar konusunda öğrenci görüşleri bağlamında nitel veriler sağlayacak olması,
- programlama öğretiminde kullanılan yöntemlere farklı bir bakış açısı kazandırması,
- programlamayı sevdirme amaçlı ilkokul, ortaokul ve liselerde Bilgisayar dersleri kapsamında Scratch kullanımını teşvik etme olasılığı,
- MEB bünyesinde, beşinci ve altıncı sınıf öğretim programlarında yer alan Bilişim Teknolojileri ve Yazılım dersi içeriğine katkı sağlaması,
- MEB bünyesinde Mesleki ve Teknik Eğitim okullarında Mesleki Eğitim ve Öğretim Sistemini Güçlendirme Projesi (MEGEP) kapsamında geliştirilen ders modüllerine katkı sağlaması;
- programlama eğitimi veren ön lisans (Bilgisayar Programcılığı vb.) ve lisans düzeyindeki (BÖTE vb.) bölümlerde eğitim gören öğrencilere programlamayı sevdirme ve programlamaya devam etmelerini sağlama olasılığı,
- programlama eğitimi veren ön lisans (Bilgisayar Programcılığı vb.) ve lisans düzeyindeki (BÖTE vb.) tüm bölümlerin temel programlama derslerini yeniden düzenlemelerine olanak sağlaması,
- BÖTE bölümünde eğitim gören öğretmen adaylarının programlama öğretimi konusunda farklı bir deneyim kazandırması, bu konuda farkındalık sağlanması ve bunu ileride öğretmenlik mesleğinde kullanmalarına teşvik etme olasılığı,

açılarından önemlidir.

Sınırlılık

Bu araştırma,

1. Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi bölümü birinci sınıfta öğrenim gören ve Programlama I dersini alan 52 öğrenci ile,
2. çalışma süresince kullanılan programlama başarı testleri, programlama öğrenmeye karşı tutum ölçeği, motivasyon ölçeği ve görüşme formlarının kapsadığı niteliklerle,
3. Programlama Dilleri I dersi kapsamında hazırlanan Scratch, akış diyagramları ve C# öğrenme etkinliklerinin kapsadığı niteliklerle sınırlıdır.

Tanımlar

Programlama: Programlama iyi bir şekilde analizi ve tasarımı yapılmış bir problemin çözümüne dair adımlar ile çözüm sürecinin bir programlama dili ile bilgisayar ortamına aktarım işi (Eryılmaz, 2003).

Algoritma: İyi tanımlanmış kuralların ve işlemlerin adım adım uygulanmasıyla bir sorunun giderilmesi veya sonuca en hızlı biçimde ulaşılması işlemi (Özkan, 2003)

***Akış Diyagramı*:** Algoritmaların içerdiği işlemlerin geometrik şekiller ile ifade edilmesi (Eker, 2011).

Scratch: Algoritma ve programlama öğretiminde kullanılan, kullanımı kolay, hikâye, animasyon ve oyun tasarımına olanak tanıyan grafik tabanlı bir yazılım (Maloney ve diğerleri, 2010).

İKİNCİ BÖLÜM
YÖNTEM

Bu bölümde araştırma modeli, çalışma grubu, veri toplama araçları, uygulama süreci ve verilerin analizine ilişkin bilgiler yer almaktadır.

Araştırma Modeli

Bu çalışmada öntest - sontest kontrol gruplu deneysel model kullanılmıştır. Deneysel desenli çalışmalar değişkenler arasındaki nedenselliği test etmeyi amaçlayan, araştırmacı tarafından değişken veya değişkenlerin kontrol altına alınabildiği ve sonuçların gözlendiği çalışma türleridir (Büyüköztürk, Çakmak, Akgün, Karadeniz ve Demirel, 2011). Deneysel çalışmalarda genellikle oluşturulan gruplara seçkisiz atama yapılmakta ve dışsal değişkenler kontrol altına alınmaktadır (Hovardoğlu, 2000). Öntest-sontest kontrol gruplu modele göre oluşturulan çalışmanın bağımlı değişkenlerini programlama başarısı ve motivasyon oluştururken; bağımsız değişkenlerini Scratch ile programlama öğretimi yöntemi oluşturmaktadır. Tablo 2'de araştırmada kullanılan modelin simgesel görünümü yer almaktadır:

Tablo 2

Araştırma Deseni

Gruplar	**Atama**	**Öntest**	**Uygulama**	**Sontest**	**Uygulama**	**Sontest 2**
Deney	R	$BT_{ÖT}$ $GÖS_{ÖTS}$	SPÖ	BT_{ST} $GÖS_{ST}$ G1	CPDÖ	BT_{ST2} $GÖS_{ST2}$ G2
Kontrol	R	$BT_{ÖT}$ $GÖS_{ÖT}$	ADPÖ	BT_{ST} $GÖS_{ST}$ G1	CPDÖ	BT_{ST2} $GÖS_{ST2}$ G2

R: Seçkisiz Atama
SPÖ: SCRATCH ile Programlama Öğretimi
ADPÖ: Akış Diyagramları ile Programlama Öğretimi
CPDÖ: C# Programlama Dilleri Öğretimi

$BT_{ÖT}$: Başarı Testi – Öntest

$GÖS_{ÖT}$: Güdülenme ve Öğrenme Stratejileri Ölçeği– Öntest

BT_{ST}: Başarı Testi – Sontest

$GÖS_{ST}$: Güdülenme ve Öğrenme Stratejileri Ölçeği – Sontest

G1: Görüşme 1

BT_{ST2}: Başarı Testi- Sontest 2

$GÖS_{ST2:}$ Güdülenme ve Öğrenme Stratejileri Ölçeği – Sontest 2

G2: Görüşme 2

Çalışma Grubu

Araştırmanın katılımcılarını Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi bölümünde eğitim gören ve 2014-2015 eğitim öğretim yılı Güz döneminde Programlama 1 dersini alan 52 üniversite öğrencisi oluşturmaktadır. Katılımcılar 26 öğrenci deney ve 26 öğrenci kontrol grubunda olacak şekilde yansız atama yapılarak iki farklı gruba ayrılmışlardır. Katılımcıların deney ve kontrol gruplarına seçkisiz atama yapılarak eşit olasılıklı atanmaları sağlanmıştır. Atama sürecinde:

- Programlama deneyimi açısından öğrencilerin mezun olduğu lise türüne göre (genel liseler/meslek liseleri) öğrencilerin gruplar arası eşit dağılımına dikkat edilmiştir.
- Katılımcılar başarı öntest puanları ve motivasyon öntest puanlarına göre denk çiftler oluşturularak sıralanmıştır.
- Gruplara atama bu denk çiftler arasından yansız atama ile yapılmıştır.

Tablo 3'de çalışmada yer alan katılımcıların demografik özellikleri yer almaktadır.

Tablo 3

Katılımcılara Ait Demografik Özellikler

		Grup (N=52)	Deney Grubu (n=26)	Kontrol Grubu (n=26)
Cinsiyet	Kadın	*F*	14	11
		%	53.84	42.31
	Erkek	*F*	12	15
		%	46.16	57.69
Lise Türü	Meslek/ Teknik Liseler	*F*	18	20
		%	69.23	76.92
	Genel/ Anadolu Liseleri	*F*	8	6
		%	30.77	23.08

Tablo 3'te görüldüğü gibi deney grubunun %53.84' ü kız öğrencilerden, %46.16'sı ise erkek öğrencilerden oluşmaktadır. Kontrol grubunun ise %42.31' i kız öğrencilerden, %57.69 ise erkek öğrencilerden oluşmaktadır. Mezun olunan lise türleri incelendiğinde deney grubunda yer alan katılımcıların %69.23'ü meslek ya da teknik liseden mezun iken , %30.79' u genel ya da Anadolu liselerinden mezun olmuştur. Kontrol grubunda ise katılımcıların %76.92' si meslek ya da teknik lise mezunu, %23.08' i ise genel ya da meslek lisesi mezunudur.

Veri Toplama Araçları

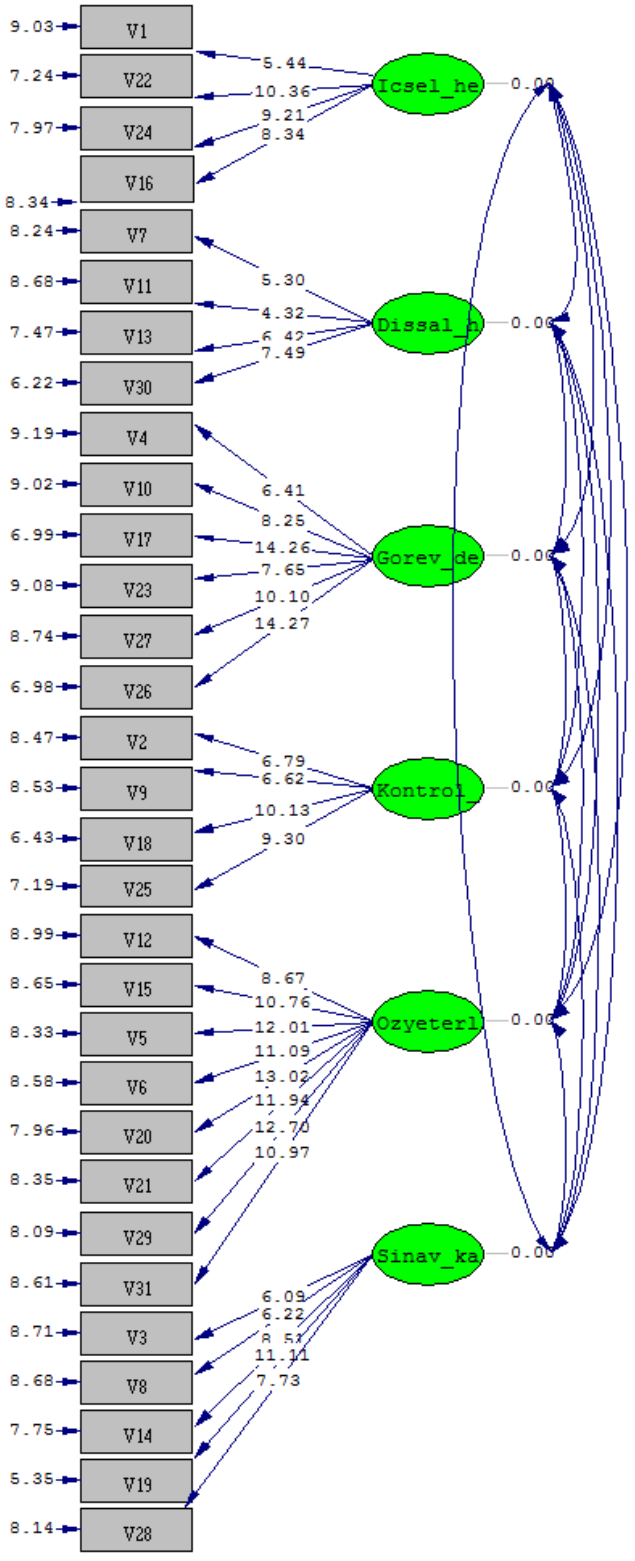
Bu bölümde araştırmada veri toplama sürecinde kullanılan Güdülenme ve Öğrenme Stratejileri Ölçeği, Başarı Testi ve Odak Grup Görüşme Formu hakkında bilgiler yer almaktadır.

Güdülenme ve Öğrenme Stratejileri Ölçeği

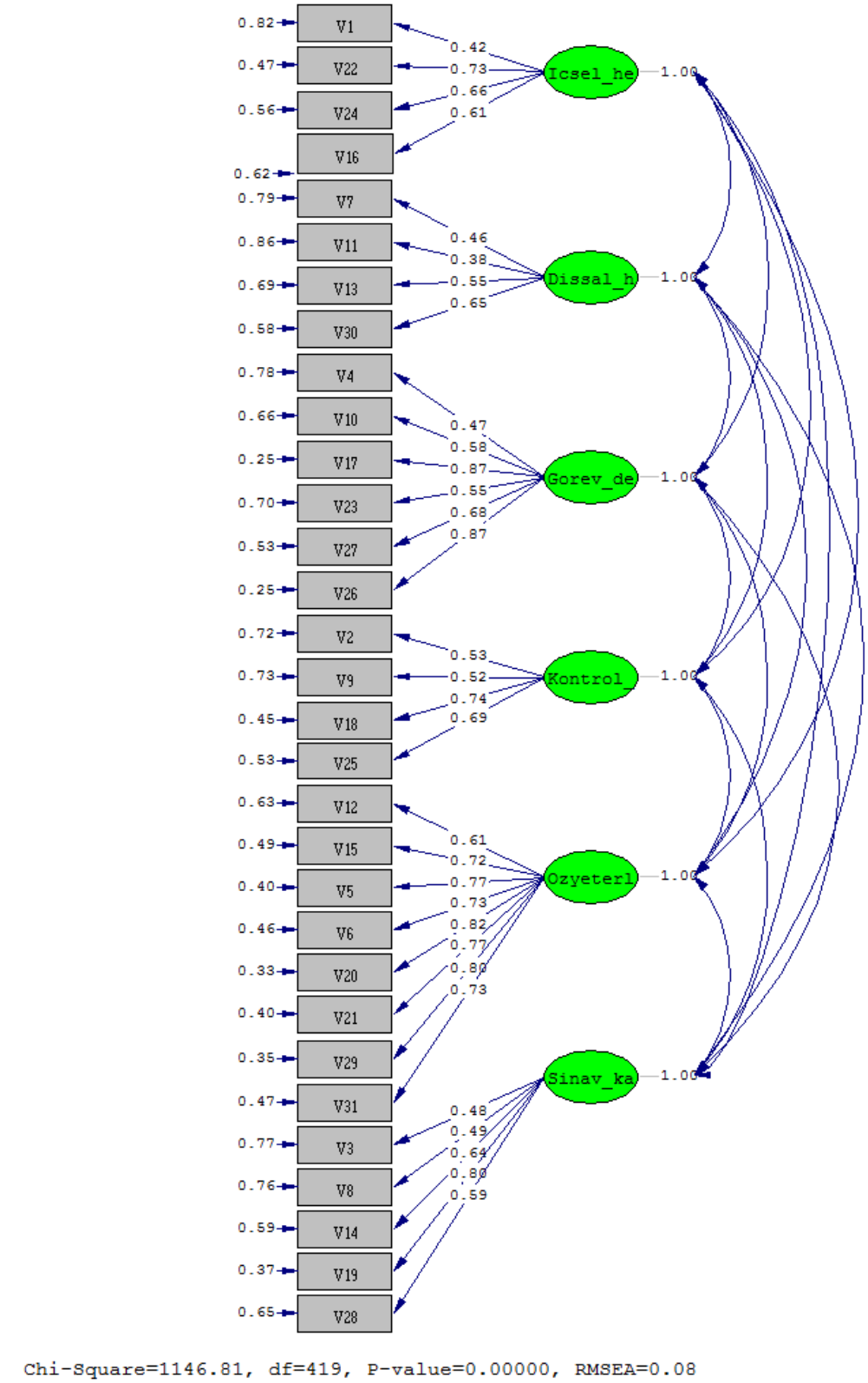
Katılımcıların motivasyon düzeylerini belirlemek için Pintrich, Smith, Garcia ve McKeachie (1993) tarafından geliştirilen; Büyüköztürk, Akgün, Özkahveci ve Demirel (2004) tarafından Türkçeye uyarlanan MSLQ olarak da bilinen "Güdülenme ve Öğrenme Stratejileri Ölçeği" gerekli izinler alınarak kullanılmıştır (EK B). Ölçek motivasyon ve öğrenme stratejileri olmak üzere iki farklı alt ölçekten oluşmaktadır (EK C). Motivasyon bölümünde 31 madde ve altı alt boyut bulunurken öğrenme stratejileri bölümünde 50 madde ve dokuz alt boyut bulunmaktadır. Bu çalışmada ölçeğin motivasyon bölümü kullanılmıştır. 7'li Likert tipi ölçekte her bir madde "1- Benim için

kesinlikle yanlış" ile "7- Benim için kesinlikle doğru" arasında değişen değerler almaktadır. Ölçek "içsel hedef düzenleme", "dışsal hedef düzenleme", "görev değeri", "öğrenmeye ilişkin kontrol inancı", "öğrenme ve performansla ilgili öz yeterlik" ve "sınav kaygısı" olmak üzere altı alt boyuttan oluşmaktadır. Türkçe uyarlama çalışmasında alt faktörlerin iç tutarlılık kat sayıları 0.52 ile 0.86 arasında değişen değerler almıştır (Büyüköztürk ve diğerleri, 2004).

Bu çalışma kapsamında uygulanmadan önce ölçeğin yapı geçerliliğini doğrulamak amacıyla doğrulayıcı faktör analizi yapılmıştır. Bu amaçla ölçek 2013-2014 bahar yarıyılında Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi BÖTE bölümünde eğitim gören 178 öğrenciye uygulanmıştır. Yapılan doğrulayıcı faktör analizi sonucunda elde edilen gizil değişkenlerin gözlenen değişkenleri açıklama oranlarına ilişkin t değerleri Şekil 10'da, standardize edilmiş kat sayılar ve hata varyansları Şekil 11'de verilmiştir.



Şekil 10. Gizil değişkenlerin gözlenen değişkenleri açıklama oranlarına ilişkin t değerleri



Şekil 11. Gözlenen değişkenlerin hata varyansları

Şekil 10' da görüldüğü gibi gizil değişkenlerin gözlenen değişkeni açıklama durumlarına ilişkin t değerleri oklar üzerinde gösterilmiştir. Parametre tahminleri eğer t değerleri 1.96'yı aşarsa .05 düzeyinde ve 2.56'yı aşarsa .01 düzeyinde anlamlıdır (Çokluk, Şekercioğlu ve Büyüköztürk, 2010). Bu durumda Şekil 10 incelendiğinde tüm değerlerin. 01 düzeyinde anlamlı olduğu görülmektedir. Ayrıca Şekil 11' de yer alan gözlenen değişkenlerin hata varyansları incelendiğinde yüksek hata varyansına sahip hiçbir değişken gözlenmemiştir. Tablo 4'de ölçeğe ilişkin uyum indeksleri yer almaktadır.

Tablo 4

Ölçek Modeline İlişkin Uyum İndeksleri Tablosu

Uyum İndeksi	**Değer**	**Kriter Değer**	**Durum**	**Kaynak**
p	0.000	$0.05 \leq p \leq 1.00$		
χ^2/sd	2.737	$0 \leq \chi^2/sd \leq 3$	Mükemmel uyum	Kline, 2005; Sümer, 2000
RMSEA	0.080	$0 \leq RMSEA \leq 0.08$	İyi uyum	Hooper, Coughlan ve Mullen, 2008; Sümer, 2000
SRMR	0.078	$0 \leq SRMR \leq 0.08$	İyi uyum	Brown, 2006; Hu ve Bentler, 1999
NFI	0.91	$0.90 \leq NFI \leq 1.00$	İyi uyum	Tabachnick ve Fidell, 2001
NNFI	0.90	$0.90 \leq NNFI \leq 1.00$	İyi uyum	Tabachnick ve Fidell, 2001
CFI	0.89	$0.90 \leq CFI \leq 1.00$	İyi uyuma yakın	Sümer, 2000; Tabachnick ve Fidell, 2001
GFI	0.91	$0.90 \leq GFI \leq 1.00$	İyi uyum	Hooper, Coughlan ve Mullen, 2008; Sümer, 2000
AGFI	0.84	$0.90 \leq GFI \leq 1.00$	İyi uyuma yakın	Kelloway, 1998; Sümer, 2000

χ^2 :1146.81; sd: 419

Tablo 4' de yer alan uyum indeksleri incelendiğinde ilk olarak beklenen kovaryans matrisi ile gözlenen kovaryans matrisi arasındaki farkın anlamlı olduğu görülmektedir ($\chi^2_{(419)}$ 1146.81; p<.01). Ancak bu durum arzu edilen bir durum değildir, çünkü değerinin anlamlı çıkmaması gerekmektedir. Örneklem büyüklüğünden kaynaklı olarak birçok doğrulayıcı faktör analizi çalışmasında bu değerin anlamlı çıkması normaldir. Bu nedenle bu durum göz ardı edilebilmektedir (Çokluk ve diğerleri, 2010). Bir diğer uyum indeksi değeri ise χ^2 ve χ^2/sd değerleridir. χ^2değeri tek başına değerlendirilen bir değer değildir. χ^2/sd değeri ile birlikte ele alındığında χ^2değerinin sd ye oranı 2.737 bulunmuştur. Bu değer Kline (2005)'a ve Sümer (2000)' e göre mükemmel uyuma karşılık gelmektedir. Yol şemasında yer alan RMSEA (yaklaşık hataların ortalama karekökü) uyum indeksi değeri incelendiğinde 0.080 değerinin iyi uyuma işaret ettiği görülmektedir (Hooper, Coughlan ve Mullen, 2008; Sümer, 2000). Standardize edilmiş RMR uyum indeksi değeri ise 0.078 olarak hesaplanmıştır. Bu değer de iyi uyuma karşılık gelmektedir (Brown, 2006; Hu ve Bentler, 1999). Normlaştırılmış uyum indeksi (NFI) ve normlaştırılmamış uyum indeksi değerleri (NNFI) incelendiğinde her ikisinde de 0.90 ve üzerinde iyi uyuma rastlanmıştır (Tabachnick ve Fidell, 2001). Karşılaştırmalı uyum indeksi (CFI) incelendiğinde 0.89 olduğu görülmektedir. Bu değer CFI değerinin iyi uyuma yakın olduğunu göstermektedir (Sümer, 2000; Tabachnick ve Fidell, 2001). Son olarak iyilik uyum indeksi (GFI) ve düzenlenmiş iyilik uyum indeksi (AGFI) incelendiğinde GFI'nın 0.91, AGFI'nın ise 0.84 olduğu görülmektedir. GFI değerinin 0.90 ve üzerinde olması iyi uyuma karşılık gelmekte iken (Hooper ve diğerleri, 2008; Sümer, 2000) AGFI'nın 0.90'a yakın bir değer olması ise iyi uyuma yakın bir uyum olduğunu göstermektedir (Kelloway, 1998; Sümer, 2000). Ortaya çıkan bu duruma göre modelin iyi bir uyuma sahip olduğu söylenebilir.

Başarı Testi

Öğrencilerin programlama ile ilgili akademik başarılarını ölçmek için araştırmacı tarafından başarı testi geliştirilmiştir (EK D). Bu başarı testi Temel Programlama Yapılarını kapsayan (Değişken, Karar verme ve Kontrol, Döngü, Dizi) çoktan seçmeli soruların yer aldığı ve açık uçlu programlama problemlerinin yer aldığı iki ayrı bölümden oluşmaktadır. Başarı testinin çoktan seçmeli test maddelerinin yer aldığı

bölümünde 20 soru, açık uçlu problemlerin yer aldığı bölümde ise dört soru yer almaktadır.

Çoktan seçmeli soruların yer aldığı başarı testinin oluşturulması için kavrama ve uygulama düzeyindeki sorulardan oluşan bir madde havuzu oluşturulmuştur. Testin kapsam ve görünüş geçerliliği için bir ölçme değerlendirme uzmanı ile programlama ve BÖTE alanından üç uzmanın görüşlerine başvurulmuştur. Uzman görüşü doğrultusunda madde havuzunda yer alan bazı sorular yeniden düzenlenmiş, seçeneklerde ya da madde köklerinde değişiklikler yapılmıştır. Başarı testinin ilk taslak formunda 25 soru yer almıştır. Bu 25 soruluk ilk taslak form pilot uygulama yapılmadan önce Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi BÖTE bölümünde eğitim gören 10 öğrenciye uygulanmıştır. Öğrencilerden test sırasında anlaşılmayan soruları araştırmacıya belirtmeleri istenmiştir. Test yaklaşık 40 dakika sürmüştür. Testi ilk bitiren öğrenci 32 dakikada son bitiren öğrenci ise 40 dakikada bitirmiştir. Öğrencilerin testi bitirme süreleri göz önüne alındığında pilot uygulamada test için 45 dakika süre öngörülmüştür. Pilot uygulamada ayrıca öğrencilerden gelen dönütlere göre anlaşılmayan sorularda yeniden düzenleme yapılmıştır.

Pilot uygulama sonrası düzenlenerek son hali verilmiş olan 25 soruluk test formu güvenirlik çalışması için Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi BÖTE bölümünde eğitim gören Programlama Dilleri I dersini almış 114 öğrenciye uygulanmıştır. Uygulama sonrası madde analizi yapılarak her bir maddenin ayırt edicilik (r) ve güçlük indisleri (p) hesaplanmış ve elde edilen değerler Tablo 5'te verilmiştir.

Tablo 5

Başarı Testine İlişkin Madde Analizi Sonuçları

Madde No	**Madde Ayırt edicilik indisi (r)**	**Ayırt Edicilik**	**Madde Güçlük İndisi (p)**	**Güçlük**	**Durum**
1	.10	Testten Atılmalı	.87	Kolay	Testten çıkartıldı
2	.29	Düzenlenebilir	.69	Orta	Yeniden düzenleme
3	.42	Çok iyi	.34	Zor	-
4	.52	Çok iyi	.35	Zor	-
5	.26	Düzenlenebilir	.45	Orta	Yeniden düzenleme
6	.48	Çok iyi	.40	Orta	-
7	.29	Düzenlenebilir	.79	Kolay	Yeniden düzenleme
8	.32	İyi	.39	Zor	-
9	.42	Çok iyi	.63	Orta	-
10	.13	Testten Atılmalı	.94	Kolay	Testten çıkartıldı
11	.06	Testten Atılmalı	.19	Zor	Testten çıkartıldı
12	.55	Çok iyi	.47	Orta	-
13	.21	Düzenlenebilir	.23	Zor	Yeniden düzenleme
14	.48	Çok iyi	.73	Kolay	-
15	.74	Çok iyi	.56	Orta	-
16	.65	Çok iyi	.55	Orta	-
17	.52	Çok iyi	.65	Orta	-
18	.29	Düzenlenebilir	.44	Orta	Yeniden düzenleme
19	.48	Çok iyi	.56	Orta	-
20	.71	Çok iyi	.48	Orta	-
21	.58	Çok iyi	.32	Zor	-
22	.42	Çok iyi	.53	Zor	-
23	.23	Düzenlenebilir	.44	Orta	Yeniden düzenleme
24	.01	Testten Atılmalı	.26	Zor	Testten çıkartıldı
25	.19	Testten Atılmalı	.16	Zor	Testten çıkartıldı

Madde güçlük indisi, her bir maddenin doğru yanıtlanma oranını göstermektedir ve "0" ile "1" arasında değerler almaktadır. Bulunan değer sıfıra yaklaştıkça maddenin zor olduğu, 1'e yaklaştıkça maddenin kolay olduğu söylenebilir. Testin ortalama madde güçlük indisinin 0,50 civarında olması ise arzu edilen bir durumdur (Çepni ve diğerleri, 2008). Madde ayırt edicilik indisi ise, testin ölçmeyi amaçladığı özelliğe yüksek düzeyde sahip bireylerle, düşük düzeyde sahip bireyleri ayırt etme derecesidir. Madde ayırt edicilik indisi "-1" ile "+1" arasında değerler alabilmektedir. Madde ayırt edicilik indisinin sıfıra yaklaşması, maddenin ayırt ediciliğinin düşük, +1'a yaklaşması ayırt ediciliğinin yüksek olması demektir. Madde ayırt edicilik indisinin negatif çıkması ise bireyleri ters ayırt ettiğini ve testten çıkartılması gerektiğini göstermektedir (Özçelik, 2010). Madde güçlük indisi düşük çıkan ve madde ayırt edicilik indisi negatif olan test maddeleri testten çıkarılmaktadır. Analiz sonucunda beş soru maddesinin ayırt edicilik indisi .20'nin altında olduğu için testten çıkartılmıştır. Madde ayırt edicilik indisi .20 ve .30 arasında olan altı madde uzman görüşü doğrultusunda yeniden düzenlenmiştir. Ayırt edicilik indisi düşük maddeler testten çıkartıldıktan sonra 20 maddelik testin ortalama ayırt edicilik indisi r=.44 olarak bulunmuştur.

Testte yer alan maddelerin güçlük indisi incelendiğinde 20 maddenin güçlük indisi .23 ile .79 arasında değişmektedir. Ortalama güçlük indisi ise p=51 olarak hesaplanmıştır. Bu durumda testin orta güçlükte bir test olduğu söylenebilir. Bununla birlikte testin güvenilirliğini ölçmek için ise KR20 iç tutarlılık kat sayısı hesaplanmıştır. KR-20 testi elde edilen test puanları arasındaki iç tutarlılığı ölçen ve özellikle başarı testlerinde kullanılan güvenilirlik testidir. Geliştirilen başarı testinin KR20 güvenirlik katsayısı .712 olarak hesaplanmıştır. Bu durumda testin güvenilir ölçüm yaptığı söylenebilir (Büyüköztürk, 2009).

Başarı testinin kararlılık katsayısını hesaplamak için test tekrar test yöntemi kullanılmıştır. Geliştirilen başarı testi, iki hafta ara ile Programlama dersini almış olan Mehmet Akif Ersoy Üniversitesi, Bilgisayar ve Öğretim Teknolojileri Öğretmenliği Bölümü, 3. sınıfta öğrenim gören 61 öğrenciye uygulanmıştır. Elde edilen iki ölçüm arasındaki ilişki Pearson Momentler Çarpımı Korelasyon Katsayısı ile hesaplanmıştır. Gerçekleştirilen uygulama sonucunda korelasyon katsayısı değeri .723 bulunmuştur. Elde edilen kararlılık katsayısı .70 ve üzerinde olduğu için testin güvenilir bir ölçüm yaptığından söz edilebilir (Büyüköztürk ve diğerleri, 2011).

Başarı testinin açık uçlu programlama problemlerinin yer aldığı ikinci bölümün oluşturulması için analiz ve sentez düzeyinde soruların yer aldığı 30 soruluk bir madde havuzu oluşturulmuştur. Oluşturulan problemler programlama ve BÖTE alanında konu alanı uzmanı üç kişinin görüşlerine sunulmuştur. Uzmanlardan bu problemleri zorluk ve uygunluk seviyesine göre puanlanmaları istenmiştir. Uzmanlardan gelen dönütlere göre başarı testi için en uygun dört problem seçilmiş ve uzman görüşlerine göre yeniden düzenlenmiştir. Bu dört problem uygunluk, zorluk ve konulara (Değişken, Karar verme ve Kontrol, Döngü, Dizi) göre dağılım gözetilerek seçilmiştir. Bu sorularda katılımcıların problemin çözümüne ilişkin hem sözde kodlarını (algoritma) hem de C# kodlarını yazmalarını isteyen iki alt seçenek yer almaktadır. Puanlama için yine uzman görüşü doğrultusunda değerlendirme formu hazırlanmış ve puanlama farklı üç puanlayıcı tarafından ayrı ayrı yapılmıştır. Puanlamanın güvenilirliği için Kendall W uyum testi ile puanlayıcılar arasındaki uyum test edilmiştir. Kendall W uyum testi ikiden çok değerlendirmecinin bir grup üzerinde yaptığı değerlendirmeleri sıralama yaparak, aralarında anlamlı derecede uyum olup olmadığını test eden non-parametrik bir testtir (Can, 2013). Kendall W ile hesaplanan değerlendiriciler arası uyum puanlar yüzde ile yapılan değerlendirmeye göre daha nesneldir. Buna göre açık uçlu problemler için yapılan analiz sonucunda üç farklı değerlendiricinin 52 katılımcı için yaptıkları değerlendirmeler arasında; hem sontest ($W = 0.911$, $p<.01$) hem de sontest 2 ölçümü sonunda ($W = 0.901$, $p<.01$) istatistiksel olarak anlamlı derecede uyum olduğu görülmüştür.

Odak Grup Görüşme Formu

Deney ve kontrol grubunda yer alan katılımcıların uygulamalar hakkında görüşlerinin belirlenmesi için araştırmacı tarafından yarı yapılandırılmış görüşme formu hazırlanmıştır (EK E). Hazırlanan form ile katılımcıların;

- uygulamaya ilişkin,
- gerçekleştirilen öğretme- öğrenme etkinliklerine ilişkin,
- motivasyonlarına olumlu ya da olumsuz etki eden durumlara ilişkin,
- süreç sonunda programlamaya karşı bakış açılarına ilişkin,
- süreç sonunda elde ettikleri kazanımları yeni durumlara transfer edebilme durumlarına ilişkin,

- gelecekte programlamaya devam edip etmeme durumlarına ilişkin görüşlerinin belirlenmesi amaçlanmıştır.

Bu amaçlar doğrultusunda görüşme soruları ve sonda sorular hazırlamış, programlama ve BÖTE alanından üç uzmanın görüşü alınarak form uygulamaya hazır hale getirilmiştir. Odak grup görüşmeleri hem deney hem de kontrol grubunda; sontest ve sontest 2 ölçümleri sonunda olmak üzere iki kez gerçekleştirilmiştir. Odak grup görüşmeleri; deney ve kontrol grupları ile ayrı ayrı olmak üzere, altı ile sekiz katılımcıdan oluşan gruplar ile yapılmıştır. Her iki odak grup görüşme sürecinde de tüm katılımcılar ile görüşme yapılmıştır. Görüşmelerden önce katılımcılar bilgilendirilmiş ve kendilerinden gerekli izinler alınmıştır (EK F). Görüşmeler yaklaşık olarak 45 dakika ile 60 dakika arasında sürmüştür.

Uygulama Süreci

Uygulama süreci, uygulama öncesi hazırlık işlemleri ve uygulamanın gerçekleştirilmesi sırasında yapılan işlemler başlıklarında ele alınmıştır.

Uygulama Öncesi Hazırlık İşlemleri

Bu başlık altında konu ve katılımcıların nasıl belirlendiği ve içeriğin hazırlanma sürecine ilişkin bilgiler yer almaktadır.

Konu seçimi ve katılımcıların belirlenmesi

Uygulama 2014-2015 eğitim öğretim yılı güz döneminde, Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, BÖTE bölümü lisans programında yer alan Programlama Dilleri I dersinde gerçekleştirilmiştir. Programlama Dilleri I dersinin temel amacı temel programlama yapılarının ve programlama mantığının öğretilmesidir. Ders içeriğinde ise algoritma ve akış diyagramları başta olmak üzere, değişken, sabit, girdi ve çıktı eylemleri, döngüler, karar verme yapıları, fonksiyonlar ve diziler gibi temel programlama yapıları yer almaktadır (Yükseköğretim Kurulu, 2007).

Uygulama için Programlama Dilleri I dersinin seçilme nedeni, programlama öğretimi için temel sayılacak bir giriş dersi niteliğinde olmasıdır. Ayrıca araştırmada Scratch ile programlama öğretiminin öğrencilerin motivasyon, programlama başarısı ve transfere etkisini incelemek amaçlanmaktadır. Bundan dolayı, Programlama Dilleri I dersinin amacı ve içeriği araştırmanın amacına ulaşmak için uygun bir derstir. Böylelikle alan yazında da söz edilen; temel programlama mantığı öğretiminde yaşanan

sıkıntılar (Cooper ve diğerleri, 2003; Garner ve diğerleri, 2005; Schulte ve Bennedsen, 2006) giderilebilir ve öğrenenlerin programlama dilleri derslerindeki başarısı ve motivasyonu artırılabilir. Bu amaç doğrultusunda uygulama 2014-2015 öğretim yılı güz döneminde, Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümünde eğitim gören ve Programlama Dilleri I dersini ilk kez alan 52 öğrenciyle gerçekleştirilmiştir. Programlamaya ilişkin temel kavramlar ve programlama mantığının öğretimi; kontrol grubunda yer alan 26 öğrenciye akış diyagramlarıyla; deney grubunda yer alan 26 öğrenciye ise Scratch programıyla gerçekleştirilmiştir.

İçeriğin Hazırlanması

Bu bölümde kontrol ve deney grubunda gerçekleştirilen ders içeriğinin hazırlanma süreci ele alınmıştır.

Kontrol grubu ders içeriğinin hazırlanması. Kontrol grubunda katılımcıların programlama mantığını ve temel programlama yapılarını (değişken, karar verme ve kontrol, döngü ve dizi gibi) mevcut ders içeriğinde olduğu şekliyle, yani akış diyagramları kullanılarak öğrenmeleri amaçlanmaktadır. Bu amaç doğrultusunda; ilk olarak ders içeriği ile ilgili kazanımlar uzman görüşü doğrultusunda ve YÖK (2007)' ün Bilgisayar ve Öğretim Teknolojileri Öğretmenliği lisans programı ders içeriği kılavuzuna göre belirlenmiş ve bu kazanımlara uygun olarak ders etkinlikleri uzman görüşü doğrultusunda hazırlanmıştır. Ayrıca kontrol grubuna uygulanan ders etkinlikleri deney grubuna paralel olarak hazırlanmış ve kontrol grubuna göre düzenlenmiştir. Hazırlanan etkinlikler temel programlama yapılarını kullanmayı gerektiren yapılandırılmış programlama problemlerini içermektedir. Katılımcılar bu problemlerin çözümünde akış diyagramlarını kullanmışlardır. Hazırlanan ders içeriği son olarak uzman görüşü doğrultusunda yeniden düzenlenmiş ve uygulama öncesi hazır hale getirilmiştir (EK H).

Deney grubu ders içeriğinin hazırlanması. Deney grubunda katılımcıların; programlama mantığını ve temel programlama yapılarını (değişken, karar verme ve kontrol, döngü ve dizi gibi) Scratch programı kullanılarak öğrenmeleri amaçlanmaktadır. Bu amaç doğrultusunda; ilk olarak ders içeriği ile ilgili kazanımlar

uzman görüşü doğrultusunda ve YÖK(2007)'ün Bilgisayar ve Öğretim Teknolojileri Öğretmenliği lisans programı ders içeriği kılavuzuna göre belirlenmiş ve bu kazanımlara uygun olarak ders etkinlikleri yine uzman görüşü doğrultusunda hazırlanmıştır. Ayrıca deney grubunda uygulanan ders etkinlikleri kontrol grubuna paralel olarak hazırlanmış ve deney grubuna göre düzenlenmiştir. Hazırlanan etkinlikler kontrol grubunda da olduğu gibi temel programlama yapılarını kullanmayı gerektiren yapılandırılmış programlama problemlerini içermektedir. Ancak katılımcılar bu problemleri çözmek için Scratch programını kullanmışlardır. Bu nedenle programlama problemleri Scratch programının yapısı gereği yeniden düzenlenmiş ve bu problemler katılımcılara bir oyun senaryosu şeklinde sunulmuştur. Ayrıca her ders sonunda katılımcıların tüm süreçlerine (senaryo, karakter, sahne, tasarım) kendilerinin karar verdiği ve altı hafta sonunda bitirmeleri gereken bir oyun tasarımlamaları için serbest tasarım etkinlikleri eklenmiştir. Bunun nedeni Scratch' in diğer görselleştiricilerden farklı olarak oyun tasarımına olanak veren yapısı ile katılımcıların öğrenme sürecine aktif katılımını sağlayarak, oyun ve öğrenmeyi bir araya getirme olanağı sağlamasıdır. Hazırlanan ders içeriği son olarak uzman görüşü doğrultusunda yeniden düzenlenmiş ve uygulama öncesi hazır hale getirilmiştir (EK G).

C# Ders içeriğinin Hazırlanması. Hem kontrol hem de deney grubunda katılımcıların; C# programlama dillerini, editör üzerinden kod yazarak öğrenmeleri amaçlanmıştır. Bu amaç doğrultusunda; ilk olarak ders içeriği ile ilgili kazanımlar uzman görüşü doğrultusunda ve YÖK (2007)' ün Bilgisayar ve Öğretim Teknolojileri Öğretmenliği lisans programı ders içeriği kılavuzuna göre belirlenmiş ve bu kazanımlara uygun olarak ders etkinlikleri uzman görüşü de alınarak hazırlanmıştır. Hazırlanan etkinlikler hem kontrol grubunda hem de deney grubunda; temel programlama yapılarını ve C# kodları kullanmayı gerektiren programlama problemlerini içermektedir. Her iki grupta da aynı örnek problemler ve aynı yöntem (gösterip yaptırma tekniği ve sunuş yoluyla öğretim) kullanılmıştır. Hazırlanan ders içeriği son olarak uzman görüşü doğrultusunda yeniden düzenlenmiş ve uygulama öncesi hazır hale getirilmiştir (EK I).

Uygulamanın Gerçekleştirilmesi Sırasında Yapılan İşlemler

Araştırma kapsamındaki uygulama hem deney hem de kontrol grubu için Mehmet Akif Ersoy Üniversitesi, Eğitim Fakültesi, BÖTE Bölümüne ait olan bilgisayar

laboratuvarında gerçekleştirilmiştir. Laboratuvarda 30 adet bilgisayar, bir adet ana bilgisayar ve bir adet projeksiyon yer almaktadır. Uygulamaya başlamadan önce uygulamanın yapılacağı ortam hazır hale getirilmiştir. Laboratuvarda yer alan bilgisayara Scratch programı ve Visiual Studio 2012 programı kurulmuştur.

Uygulama öncesi katılımcılara uygulama hakkında bilgi verilmiş ve araştırmaya gönüllü katıldıklarına ilişkin onayları yazılı olarak alınmıştır (EK M). Öntestler uygulandıktan sonra seçkisiz atama yapılarak deney ve kontrol grupları oluşturulmuştur.

Uygulama süreci; testlerin toplanma süreci de dâhil olmak üzere toplam 14 hafta sürmüştür. İlk 7 haftalık bölümde katılımcılara, temel programlama yapılarının öğretilmesi amaçlanmıştır. Kontrol grubunda temel programlama yapılarının öğretimi mevcut programda yer alan şekliyle yani akış diyagramları kullanılarak gerçekleştirilmiştir. Bu doğrultuda düz anlatım yöntemi kullanılarak temel yapıların öğretimi yapılmış ve programlama problemleri akış diyagramları kullanılarak çözümlenmiştir. Deney grubunda ise bu süreç Scratch programı ile tasarım etkinlikleri yapılarak gerçekleştirilmiştir. Her iki grupta da benzer problemler ve örnekler kullanılmıştır. Deney grubunda ise tüm bunlardan farklı olarak katılımcılardan kendilerinin belirlediği bir oyunu altı haftalık süreç içerisinde (senaryo, karakter, sahne, tasarım) tasarlamaları ve sunmaları istenmiştir. Bu amaçla katılımcılar en çok iki kişilik gruplar halinde her haftanın son bir saatinde belirledikleri oyunu tasarlamak için serbest tasarım etkinliği gerçekleştirmişlerdir. Oyun tasarımı ile ilgili her türlü süreç; oyun senaryosunun belirlenmesi, karakterlerin oluşturulması, sahnelerin hazırlanması ve tasarımın gerçekleştirilmesi katılımcılara bırakılmıştır. Katılımcılar tüm bu süreçleri kendileri yönetmiştir. Katılımcılar bu süreçte hem arkadaşları hem de öğretici ile iletişime geçmişler, aldıkları dönütlere göre tasarımlarını gerçekleştirmişlerdir. Altıncı haftanın sonunda bu oyunları sınıfta sunmuşlar ve Scratch' in resmi sayfasında paylaşımda bulunmuşlardır.

Uygulamanın ikinci yedi haftalık bölümde ise hem kontrol hem de deney grubunda C# programlama dili öğretimi gerçekleştirilmiştir. Her iki grupta da sunuş yoluyla öğretim yöntemi ve gösterip yaptırma tekniği kullanılarak, benzer problemler ve örnekler uygulanmıştır. Bir başka deyişle C# programlama dilleri öğretimi süresince her iki grupta da aynı yöntem uygulanmıştır.

Araştırma süresince hem deney hem kontrol grubundaki uygulamalar aynı araştırmacı tarafından gerçekleştirilmiştir. Uygulama boyunca izlenen süreç Tablo 6'da yer almaktadır.

Tablo 6

Uygulama Süreci

Tarih	**Kontrol Grubu**	**Deney Grubu**
08- 12 Eylül 2014	1. Uygulama Hakkına Bilgilendirme a. Katılımcı Onayının alınması 2. Öntestlerin uygulanması (1. Ölçüm) a. Başarı Testi b. Güdülenme ve Öğrenme Stratejileri Ölçeği 3. Seçkisiz Atama ile Grupların Oluşturulması a. 26 Öğrenci Deney Grubu b. 26 Öğrenci Kontrol Grubu	
15-19 Eylül 2014	**1.Haftanın Uygulanması** 1. Temel Kavramlar 2. Akış Diyagramları ve Semboller	**1.Haftanın Uygulanması** 1. Temel Kavramlar 2. Scratch ve Yapısı
22-26 Eylül 2014	**2.Haftanın Uygulanması** 1. Değişkenler 2. Girdi - Çıktı Eylemleri 3. İlgili problemlerin akış diyagramı ile çözümü	**2.Haftanın Uygulanması** 1. Değişkenler 2. Girdi - Çıktı Eylemleri 3. İlgili problemlerin SCRATCH ile tasarlanması 4. Oyun Tasarımı a. Senaryoların belirlenmesi
29 Eylül - 03 Ekim 2014	**3.Haftanın Uygulanması** 1. Kontrol ve Karar Verme Eylemleri 2. İlgili problemlerin akış diyagramı ile çözümü	**3.Haftanın Uygulanması** 1. Kontrol ve Karar Verme Eylemleri 2. İlgili problemlerin SCRATCH ile tasarlanması 3. Oyun Tasarımı a. Karakterlerin Tasarımı
06 - 10 Ekim 2014	**4.Haftanın Uygulanması** 1. Döngü Eylemleri 2. İlgili problemlerin akış diyagramı ile çözümü	**4.Haftanın Uygulanması** 1. Döngü Eylemleri 2. İlgili problemlerin SCRATCH ile tasarlanması 3. Oyun Tasarımı a. Sahne Tasarımı
13-17 Ekim 2014	**5.Haftanın Uygulanması** 1. İç içe Döngü Eylemleri 2. İlgili problemlerin akış diyagramı ile çözümü	**5.Haftanın Uygulanması** 1. İç içe Döngü Eylemleri 2. İlgili problemlerin SCRATCH ile tasarlanması 3. Oyun Tasarımı a. Oyun Tasarımının gerçekleştirilmesi b. Dönüt ve Yeniden Düzenleme
20- 24 Ekim 2014	**6.Haftanın Uygulanması** 1. Dizi Eylemleri 2. İlgili problemlerin akış diyagramı ile çözümü	**6.Haftanın Uygulanması** 1. Dizi Eylemleri 2. İlgili problemlerin SCRATCH ile tasarlanması 3. Oyun Tasarımı

	a. Oyun Tasarımının sonlanması b. Oyunun sunulması ve yeniden düzenleme c. Oyunun paylaşımı
27 – 31 Ekim 2014	**7. Haftanın Uygulanması** 1. Sontestlerin uygulanması (2. Ölçüm) a. Başarı Testi b. Güdülenme ve Öğrenme Stratejileri Ölçeği 2. Odak Grup Görüşmelerin Yapılması a. Kontrol Grubu b. Deney Grubu
3 - 7 Kasım 2014	**1. Haftanın Uygulanması** 1. Temel Kavramlar a. C#' a Giriş b. Derleyiciler ve Visiual Studio hakkında 2. Değişkenler a. Veri türleri b. Dönüşümler c. Operatörler 3. İlgili problemlerin derleyicide çözümü
10- 14 Kasım 2014	**2. Haftanın Uygulanması** 1. Console.Write() Yapısı 2. Console.WriteLine() Yapısı 3. Console. Read() Yapısı 4. Console. ReadLine() Yapısı 5. İlgili problemlerin derleyicide çözümü
17- 24 Kasım 2014	**3. Haftanın Uygulanması** 1. IF ELSE Yapısı 2. Switch Case Yapısı 3. İlgili problemlerin derleyicide çözümü
24 -28 Kasım 2014	**4. Haftanın Uygulanması** 1. For Yapısı 2. While Yapısı 3. Do - While Yapısı 4. İlgili problemlerin derleyicide çözümü
01-05 Aralık 2014	**5. Haftanın Uygulanması** 1. Arrays 2. İlgili problemlerin derleyicide çözümü
08 -12 Aralık 2014	**6. Haftanın Uygulanması** 1. Metotlar 2. Dosya İşlemleri 3. İlgili problemlerin derleyicide çözümü
15-19 Aralık 2014	**7. Haftanın Uygulanması** 1. Sontestlerin uygulanması (2. Ölçüm) a. Başarı Testi b. Güdülenme ve Öğrenme Stratejileri Ölçeği 2. Odak Grup Görüşmelerin Yapılması a. Kontrol Grubu b. Deney Grubu

Verilerin Analizi

Araştırma süresince elde edilen verilerin analizinde öncelikle frekans, yüzde, ortalama ve standart sapma gibi betimsel istatistikler hesaplanmıştır. Tüm analizlerde anlamlılık düzeyi .05 olarak ele alınmıştır. Ancak birden çok karşılaştırma gerektiren analizlerde

Bonferroni düzeltmesi yapılarak I. tip hata olasılığı en aza indirilmiştir. Ayrıca farklılığın büyüklüğünü belirlemek için eta kare etki büyüklüğü değeri incelenmiştir. Cohen' e (1998) göre etki büyüklüğü değeri .01 ile .06 arasında ise küçük, .06'dan .14'e kadar ise orta, .14 ve üzeri ise büyük bir etki olarak kabul edilmektedir. Farklılığın bulunmadığı durumlarda ise istatistiksel güç rapor edilmiştir. Buna göre istatistiksel güç .80'in altında ise; istatistiksel gücün yetersiz olduğu ve daha büyük örneklemlerle bu değerin anlamlı çıkmasının mümkün olduğu söylenebilir (Akbulut, 2010, Ellis, 2010).

Başarı testinin çoktan seçmeli bölümünün geliştirilmesi aşamasında; madde güçlük indisi, madde ayırt edicilik indisi, KR 20 iç tutarlık kat sayısı ve test tekrar yöntemi ile elde edilen verilerin analizi için Pearson Momentler Çarpımı Korelasyon Katsayısı hesaplanmıştır. Başarı testinin açık uçlu programlama problemlerinin yer aldığı ikinci bölümünde ise puanlamanın güvenilirliği için Kendall W uyum testi ile puanlayıcılar arasındaki uyum test edilmiştir.

Güdülenme ve Öğrenme Stratejileri Ölçeği için uygulamadan önce yapılan doğrulayıcı faktör analizinde χ^2, χ^2 /sd değerleri ve RMSEA, SRMR, NFI, NNFI, CFI, GFI ve AGFI uyum indeksleri incelenmiştir.

Katılımcıların programlama dili öğrenmeye yönelik motivasyonlarını ve akademik başarılarını artırmada Scratch ile öğrenen grup ile mevcut ders programına göre öğretimin gerçekleştirildiği grup arasında anlamlı bir farklılığın olup olmadığını belirlemek için 3 x 2 karma desen MANOVA testi kullanılmıştır. Bu analizde iki farklı ders yöntemi gruplar içi faktörü, ölçme araçlarına ait öntest, sontest ve sontest 2 şeklindeki ölçüm zamanları ise yinelenen ölçüm faktörünü oluşturmaktadır. MANOVA testinin ön şartlarının sağlanıp sağlanmadığını kontrol etmek için aşağıdaki işlemler gerçekleştirilmiştir (Akbulut, 2010):

Örneklem büyüklüğü; testin yapılabilmesi için her hücrede en az bağımlı değişken sayısı kadar (Akbulut, 2010; Pallant, 2001) yani bu araştırma için en az iki katılımcı olması gerekmektedir. Hem deney grubunda hem de kontrol grubunda Güdülenme ve Öğrenme Stratejileri Ölçeğini ve başarı testinin öntest, sontest ve sontest 2 testlerini yanıtlayan 26'şar öğrenci yer almaktadır. Bu durumda sözü edilen şart sağlanmıştır.

Normallik; MANOVA testinin yapılabilmesi için hem tek değişkenli hem de çok değişkenli normallik şartlarının sağlanması gerekmektedir. Tek değişkenli normallik

şartı için gerekli olan değerler (çarpıklık, basıklık değerleri ve histogram, Q-Q grafiği) incelenmiş her bir grupta tüm bağımlı değişkenlerin normal dağılama yakın dağılım gösterdiği görülmüştür (EK K). Çok değişkenli normal dağılım için ise Mahalanobis uzaklık değerleri ve uç değerler incelenmiştir. Uzaklık değerleri incelendiğinde (Tablo 7) hiç bir değerin kritik değer olan 22.46'dan yüksek olmadığı görülmektedir (Akbulut, 2010; Pearson ve Hartley, 1958).

Tablo 7

Mahalanobis Uzaklık Değerleri

Mahalanobis Uzaklığı		**Katılımcı**	**Değer**
En Yüksek	1	1	15.61142
	2	4	13.91276
	3	21	11.99456
	4	44	11.83170
	5	39	9.33649
En Düşük	1	51	.93485
	2	28	1.73632
	3	19	1.84175
	4	2	1.89100
	5	5	2.10645

Doğrusallık; bu şartın test edilmesi için veri seti bağımsız değişkenler grup bazında bölünerek saçılım grafikleri incelenmiştir. Yapılan inceleme sonunda hem başarı testinin hem de Güdülenme ve Öğrenme Stratejileri Ölçeğinin matriste eşleştirilen saçılım grafiklerinin elips şekline olduğu görülmüştür (EK L).

Varyans - Kovaryans matrisinin homojenliği; bu ön koşul için Box' ın Varyans-Kovaryans Matrislerinin homojenliği test edilmiş, elde edilen değerler Tablo 8'de sunulmuştur.

Tablo 8

Varyans - Kovaryans Matrisinin Homojenliği Testi

	Box' s M	**Sd1**	**Sd2**	**F**	**P**
Değer	21.513	21	9194.996	.892	.603

Tablo 8 'e göre anlamlılık değeri 0.603 olarak bulunmuştur. Bu değer .05 düzeyinde varyans - kovaryans matrisinin homojenliği şartının sağlandığını göstermektedir.

Hata varyanslarının homojenliği; bağımlı değişkenlerin her biri için hata varyanslarının homojenliği Levene F testiyle test edilmiş ve elde edilen değerler Tablo 9'da verilmiştir.

Tablo 9

Hata Varyanslarının Homojenliğine İlişkin Levene F Testi Sonuçları

	F	**Sd1**	**Sd2**	**P**
GÖS Öntest	3.205	1	50	.069
Başarı Öntest	.335	1	50	.565
GÖS Sontest	.135	1	50	.715
Başarı Sontest	.006	1	50	.940
GÖS Sontest 2	.349	1	50	.557
Başarı Sontest 2	.275	1	50	.602

Tablo 9 incelendiğinde her iki bağımlı değişkenin tüm ölçümlerinde p değeri .05'den büyük olduğundan hata varyanslarının homojen olduğu görülmüştür.

Çoklu doğrusal bağıntı ve tekillik; bağımlı değişkenler arasında korelasyon değerinin .90 ve üzerinde olması tekillik sorununa neden olmaktadır (Akbulut, 2010). Öntest, sontest ve sontest 2 ölçümleri için her bir bağımlı değişkende Pearson korelasyon testi yapılmış ve korelasyon için .258 ve .448 arasında değişen değerler elde edilmiştir. Bu değerlere göre bağımlı değişkenler arasında tekillik sorunu yoktur.

Araştırma sürecinde odak grup görüşmesiyle toplanan nitel verilerin analizi için içerik analizi yapılmıştır. İçerik analizinde temel amaç, elde edilen verileri daha derinlemesine açıklayabilecek kavramlara ve ilişkilere ulaşmaktır. İçerik analizinde önce veriler kavramsallaştırılır ve birbirine benzeyen kavramlar temalaştırılarak bir

araya getirilir. Böylelikle elde edilen veriler tümevarımsal bir yaklaşımla örgütlenerek yorumlanır (Yıldırım ve Şimşek, 2000). Bu süreç verilerin kodlanması, temaların bulunması, kodların ve temaların organize edilmesi ve bulguların tanımlanarak yorumlanması şeklinde gerçekleşmektedir. Odak grup görüşmesinde elde edilen ses dosyaları araştırmacı tarafından bilgisayar ortamına yazılı metin olarak aktarılmıştır. Elde edilen dijital veriler her bir görüşme sorusu için önce kodlanmış daha sonra temalaştırılmıştır. Bu sürece araştırmacıdan farklı olarak bu konuda bir uzman daha dâhil edilmiştir. Hem araştırmacı hem de uzman tarafından yapılan analiz sonuçları görüş birliği sağlanması açısından bir araya gelinerek yeniden gözden geçirilmiştir. İki kodlayıcı arasında ki oluşturulan temalar açısından tutarlılığın hesaplanması için Miles ve Huberman' ın (1994) "Görüş birliği/(Görüş ayrılığı +Görüş birliği)*100" formülü kullanılmıştır. Birinci odak grup görüşmesinde güvenilirlik; her bir görüşme sorusu için %80 ile %92 arasında, ikinci odak grup görüşmesinde ise her bir görüşme sorusu için %84 ile %93 arasında değişen oranlarda hesaplanmıştır. Yıldırım ve Şimşek (2000) birden çok kodlayıcının yer aldığı durumlarda en az %80 düzeyinde bir güvenilirlik yüzdesine ulaşılması gerektiğini vurgulamaktadır. Buna göre iki kodlayıcı arasında görüş birliğine varıldığı söylenebilir. Nitel bulgular yorumlanırken doğrudan alıntılar ile desteklenmiştir. Doğrudan alıntılarda öğrenci kimliği gizli tutulmuş gerçek isimlerinin yerine önceden tanımlanmış kod isimler kullanılmıştır. Kod isimler deney ve kontrol grubunda yer alma durumuna göre katılımcı numarası verilerek yapılmıştır. Örneğin deney grubunda yer alan birinci katılımcı için "DGK-1", kontrol grubunda yer alan birinci katılımcı için "KGK-1" şeklinde kod isim kullanılmıştır.

ÜÇÜNCÜ BÖLÜM

BULGULAR VE YORUMLAR

Bu bölümde araştırmanın alt problemlerine ilişkin çözümlemeler sonucunda elde edilen bulgular ve yorumlar yer almaktadır.

Motivasyon ve Başarı Puanlarının Farklı Öğretim Yöntemi ve Ölçüm Zamanına Göre Değişimine İlişkin Bulgular

Bu bölümde araştırmanın birinci alt problemi olan katılımcıların programlama dili öğrenmeye yönelik motivasyonlarını ve akademik başarılarını artırmada Scratch ile öğrenen grup ile mevcut ders programına göre öğretimin gerçekleştirildiği grup arasında anlamlı bir fark olup olmadığı sorusuna yanıt aranmıştır. Katılımcıların deney ve kontrol grubuna göre ölçme araçlarından öntest, sontest ve sontest 2 testlerinden aldıkları puanlara ait betimsel istatistikler Tablo 10'da verilmiştir.

Tablo 10

Motivasyon ve Programlama Başarısı Öntest, Sontest ve Sontest 2 Puanlarına İlişkin Betimsel İstatistikler

	Grup	**n**	$\bar{X}$	**Ss**
GÖS (Motivasyon) Öntest	Kontrol	26	151.92	22.509
	Deney	26	153.04	16.153
	Toplam	52	152.48	19.405
Başarı Öntest	Kontrol	26	6.62	5.507
	Deney	26	7.58	6.236
	Toplam	52	7.10	5.845
GÖS (Motivasyon) Sontest	Kontrol	26	143.00	22.293
	Deney	26	159.35	22.165
	Toplam	52	151.17	23.506
Başarı Sontest	Kontrol	26	20.32	8.709
	Deney	26	29.51	9.357
	Toplam	52	24.92	10.081
GÖS (Motivasyon) Sontest 2	Kontrol	26	144.04	23.787
	Deney	26	164.85	19.713
	Toplam	52	154.44	24.046
Başarı Sontest 2	Kontrol	26	39.62	20.413
	Deney	26	58.64	23.027
	Toplam	52	49.13	23.589

Tablo 10' da görüldüğü üzere hem motivasyon (GÖS) hem de başarı puanları açısından üç farklı ölçüm zamanında da deney grubu puanlarının kontrol grubuna göre daha yüksek olduğu görülmektedir. Ölçme araçlarından edinilen puanların grup

değişkeni ve ölçme zamanı (öntest-sontest- sontest 2) bağlamında farklılığının test edilmesi için karma desen MANOVA testi uygulanmıştır. Faktöriyel desen olarak gerçekleştirilen analizde; kullanılan iki farklı öğretim yöntemi (Scratch ile oyun tasarımı ve akış diyagramları ile problem çözme) gruplar arası faktörü, yinelenen üç ölçüm (Ölçüm Zamanı-öntest, sontest, sontest 2) ise gruplar içi faktörü oluşturmaktadır. Gruplar arası farklılık incelenirken, her bir bağımlı değişken sayısı kadar analiz gerçekleştirildiği için anlamlılık düzeyi yeniden düzenlenmiştir. Grup değişkeni için anlamlılık düzeyi Bonferroni düzeltmesi ile .05/2=.025 şeklinde ele alınmıştır. Ölçüm zamanı ve grup değişkenine göre veri toplama araçlarından alınan puanlara ilişkin MANOVA sonuçları Tablo 11'de sunulmuştur.

Tablo 11

Ölçüm Zamanı ve Grup Değişkenine Göre Motivasyon ve Programlama Başarısı Ortalama Puanlarına İlişkin Çok Değişkenli Varyans (MANOVA) Analizi Sonuçları

Varyansın kaynağı	Wilks' Λ	F	Denence sd	Hata sd	p	η^2
Grup	.800	6.106	2.000	49.000	.004	.200
Ölçüm Zamanı	.172	56.434	4.000	47.000	<.001	.828
Ölçüm Zamanı * Grup	.712	4.755	4.000	47.000	.003	.288

Tablo 11' de görüldüğü üzere; katılımcıların motivasyon ve başarı puanları, grup değişkenine göre anlamlı bir farklılık göstermektedir (Wilks' Λ = .800, $F_{(2,49)}$=6.106,p<.025). Buna göre deney ve kontrol gruplarında uygulanan öğretim yöntemi türünün, katılımcıların motivasyon ve başarı puanlarına etki ettiği söylenebilir. Etki büyüklüğü değeri ise .200 olarak hesaplanmıştır. Buna göre büyük bir etkiden söz edilebilir (Cohen, 1998). Diğer bir ifadeyle motivasyon ve başarı puanlarındaki varyansın %20' si grup değişkeniyle yani kullanılan öğretim yöntemi ile açıklanabilir.

Tablo 11 incelendiğinde ölçüm zamanı değişkenine göre de katılımcıların motivasyon ve başarı puanlarının anlamlı bir farklılık gösterdiği görülmektedir (Wilks' Λ = .172, $F_{(4,47)}$=56.434, p<.001). Buna göre katılımcıların motivasyon ve başarı

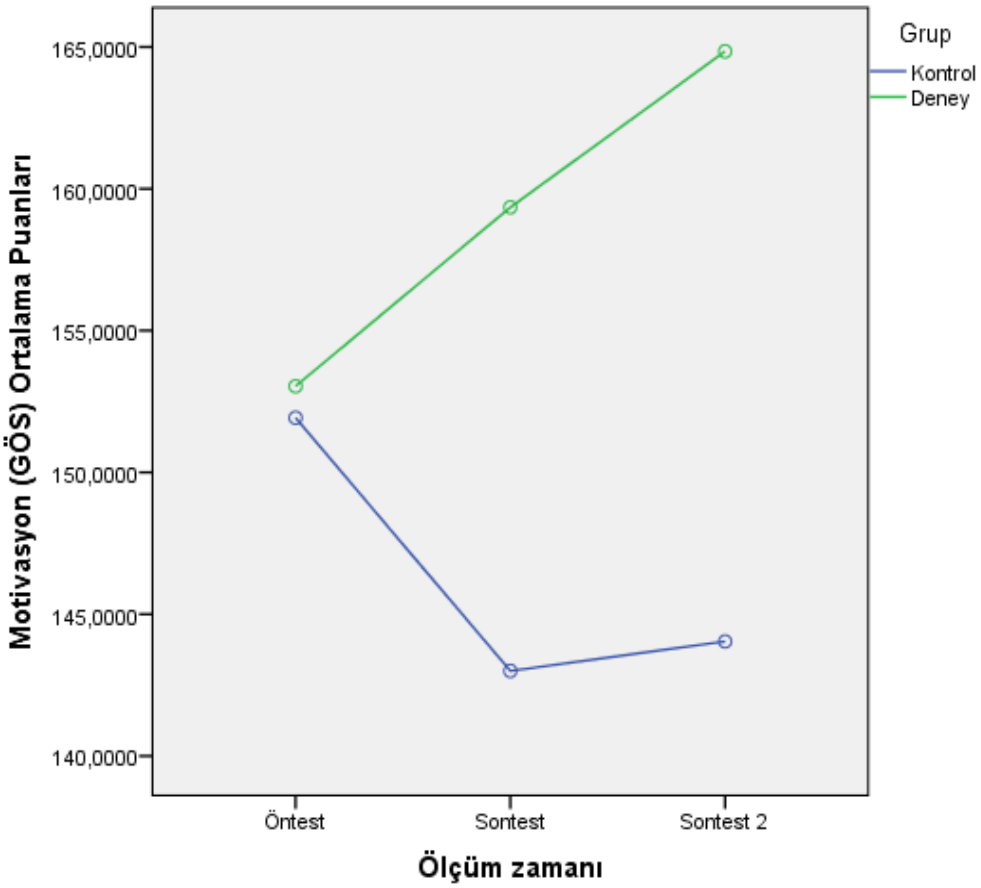
puanlarının öntest, sontest ve sontest 2 ölçümlerine göre farklılık gösterdiği söylenebilir. Etki büyüklüğü değeri .828 olarak hesaplanmıştır. Buna göre büyük bir etkiden söz edilebilir (Cohen, 1998). Diğer bir ifadeyle motivasyon ve başarı puanlarındaki varyansın %82.8'i ölçüm zamanıyla açıklanabilir. Diğer bir bulgu ise ölçüm zamanı ve grup değişkenin (ölçüm zamanı*grup) ortak etkisine göre öğrencilerin motivasyon ve programlama başarısı puanlarının anlamlı farklılık göstermesidir (Wilks'Λ = .712, $F_{(4,47)}$=4.755, p<.025). Buna göre katılımcıların öntest, sontest ve sontest 2 testlerinden aldıkları motivasyon ve başarı puanlarının; grup değişkenine göre bir başka deyişle kullanılan öğretim yöntemine göre farklılık gösterdiği söylenebilir. Ölçüm zamanı ve grup (ölçüm zamanı*grup) etkileşiminin katılımcıların motivasyon ve başarı puanlarında farklılık oluşturması önemli bir bulgudur. Bu nedenle motivasyon ve programlama başarı puanları; grup değişkeni bağlamında tek tek değerlendirilmek yerine ölçüm zamanı ve grup değişkenin etkileşimi bağlamında incelenmiştir. Bu durum bulguların daha sağlıklı yorumlanması açısından önemlidir (Huck, 2000). Etki büyüklüğü değeri .288 olarak hesaplanmıştır. Buna göre büyük bir etkiden söz edilebilir (Cohen, 1998). Diğer bir ifadeyle motivasyon ve programlama başarı puanlarındaki varyansın %28.8' i ölçüm zamanı ve grup değişkeninin etkileşimi ile açıklanabilir. Ölçüm zamanı ve grup değişkeninin birlikte etkileşiminin daha iyi incelenebilmesi için motivasyon ve programlama başarısı değişkenleri ayrı ayrı ele alınarak 3x2 karma desen ANOVA yapılmıştır. Motivasyon değişkenine ilişkin tek değişkenli varyans analizi sonuçları Tablo 12' de verilmiştir.

Tablo 12

Ölçme Zamanı ve Grup Değişkenine Göre Motivasyon Ortalama Puanlarına İlişkin Tek Değişkenli Varyans Analizi Sonuçları

Varyansın Kaynağı	**KT**	**Sd**	**KO**	**F**	**P**	**η^2**
Gruplar arası						
Grup	6346.314	1	6346.314	6.320	.015	.112
Hata	50204.526	50	1004.091			
Gruplar içi						
Ölçüm Zamanı	281.590	1.750	160.920	.802	.437	.016
Ölçüm Zamanı*Grup	2771.897	1.750	1584.052	7.897	.001	.136
Hata	17550.513	87.494	200.591			
Toplam	77154.84	141.994				

Tablo 12'den de görüldüğü gibi katılımcıların motivasyon puanları grup değişkenine göre anlamlı farklılık göstermektedir ($F_{(1,50)}$ =6.320, p<.025, η^2=.112). Buna göre katılımcıların motivasyon puanlarının grup değişkeni yani öğretim yöntemi türüne göre farklılaştığı söylenebilir. Ayrıca etki büyüklüğü değeri incelendiğinde büyük bir etkiden söz edilebilir (Cohen, 1998). Ölçüm zamanı değişkenine göre ise motivasyon puanları arasında anlamlı farklılık bulunmamaktadır ($F_{(1.750,87.494)}$ =160.920, p=.437, güç=.174). Ölçüm zamanı ve grup değişkenin ortak etkisi incelendiğinde ise motivasyon puanları arasında anlamlı farklılık olduğu görülmektedir ($F_{(1.750,87.494)}$ =1584.052, p<.01, η^2=.136). Buna göre ölçüm zamanına göre kontrol ve deney grubunun motivasyon puanlarının değiştiği söylenebilir. Ortak etkinin daha ayrıntılı incelenebilmesi için Şekil 11'de yer alan ölçüm zamanına göre deney ve kontrol grubunun motivasyon puanlarını gösteren grafik incelenmiştir.



Şekil 11: Ölçüm zamanı ve gruplara göre motivasyon ortalama puanları grafiği

Şekil 11' de görüldüğü üzere deney grubunda motivasyon puanı sürekli artarken, kontrol grubunda önce azalmış daha sonra ise durağan hale gelmiştir. Grup bağlamında incelendiğinde ise öntest puanlarının birbirine yakın sontest ve sontest 2 puanlarının ise deney grubunda daha yüksek olduğu görülmektedir. Gözlenen bu durumların istatistiksel açıdan anlamlı olup olmadığının test edilmesi için basit temel etki analizi yapılmıştır.

İlk olarak grupların kendi içindeki motivasyon puanları ölçüm zamanına göre karşılaştırılmıştır (gruplar içi karşılaştırma). Bu bağlamda hem kontrol hem de deney grubunda; motivasyon öntest, sontest ve sontest 2 ortalama puanları arasında anlamlı farklılık bulunup bulunmadığını test etmek amacıyla her bir grupta ayrı ayrı olmak üzere bağımlı örneklemler t-testi yapılmıştır. Analiz her bir grupta ve her bir ölçme zamanı arasında ayrı ayrı yapıldığı için Bonferroni düzeltmesi yapılmıştır. Bundan dolayı anlamlılık düzeyi yapılan analiz sayısına bölünerek .025/6 =.004 olarak ele

alınmıştır. Analiz sonucuna göre kontrol grubunda öntest ile sontest arasında ($t_{(25)}$=2.563, p=.017), sontest ile sontest 2 arasında($t_{(25)}$=-.331, p=.743) ve öntest ile sontest 2 arasında ($t_{(25)}$=2.034, p=.053) anlamlı farklılık bulunamamıştır. Deney grubunda da öntest ile sontest arasında ($t_{(25)}$=-1.380, p=.018), sontest ile sontest 2 arasında ($t_{(25)}$=-2.074, p=.049) ve öntest ile sontest 2 arasında ($t_{(25)}$=-2.943, p=.007) anlamlı farklılık bulunamamıştır. Ancak bu durum Bonferroni düzeltmesi sonucu oluşan II. tip hatadan dolayı ortaya çıkmış olabilir. Aslında yapılan karşılaştırmada anlamlı fark çıkabilecek iken anlamlılık düzeyinin daraltılmasından dolayı anlamlı fark bulunamamış olabilir. Ortalama puanlar incelendiğinde ise kontrol grubunda katılımcıların motivasyon puanlarının birinci ölçümden ikinci ölçüme doğru azaldığı, ikinci ölçümden üçüncü ölçüme doğru arttığı görülmektedir. Ancak üçüncü ölçüm sonunda katılımcıların motivasyon puanlarının birinci ölçüme göre azalmış olduğu görülmektedir. Buna göre kontrol grubunda katılımcıların motivasyon puanlarının tüm uygulama sonunda azaldığı söylenebilir. Deney grubunda ise katılımcıların motivasyon puanlarının birinci ölçümden (öntest) ikinci ölçüme (sontest) doğru ve ikinci ölçümden (sontest) üçüncü ölçüme (sontest 2) doğru arttığı görülmektedir. Ayrıca üçüncü ölçüm (sontest 2) sonunda katılımcıların motivasyon puanlarının birinci ölçüme (öntest) göre arttığı görülmektedir. Bu sonuca göre deney grubunda katılımcıların motivasyon puanlarının tüm uygulama sonunda arttığı söylenebilir. Bu değişimlerin hangi grupta anlamlı olarak farklılık gösterdiğini görmek için ölçümlerde elde edilen puanlar gruplara göre karşılaştırılmıştır. Bunun için basit etki analizi bağlamında ikinci bir analiz gerçekleştirilmiştir.

İkinci olarak motivasyon için farklı ölçüm zamanlarında elde edilen puanlar (öntest, sontest, sontest 2) gruplara göre karşılaştırılmıştır (gruplar arası karşılaştırma). Bu bağlamda motivasyon öntest, sontest ve sontest 2 puanlarının grup değişkenine göre farklılık gösterip göstermediğini test etmek amacıyla her bir ölçüm için bağımsız örneklem t-testi yapılmıştır. Analiz her bir ölçüm zamanı için ayrı ayrı yapıldığından Bonferroni düzeltmesi yapılmıştır. Bundan dolayı anlamlılık düzeyi yapılan analiz sayısına bölünerek .025/3 = .008 olarak ele alınmıştır. Analiz sonucuna göre deney ve kontrol grubu arasında motivasyon öntest puanları açısından istatistiksel açıdan anlamlı farklılık yok iken ($t_{(50)}$=-.205, p=.838); motivasyon sontest puanları açısından ($t_{(50)}$=-2.651,p<.008) ve sontest 2 puanları açısından deney grubu lehine anlamlı farklılık

olduğu görülmüştür ($t_{(50)}$=-3.434, p<.001). İlk başta motivasyon puanları her iki grupta aynı düzeyde iken, sontest ve sontest 2' de deney grubu lehine anlamlı bir farklılaşma olmuştur. Buna göre deney grubunda kullanılan öğretim yönteminin kontrol grubunda kullanılan yönteme göre öğrencilerin motivasyonlarına daha çok etki ettiği söylenebilir.

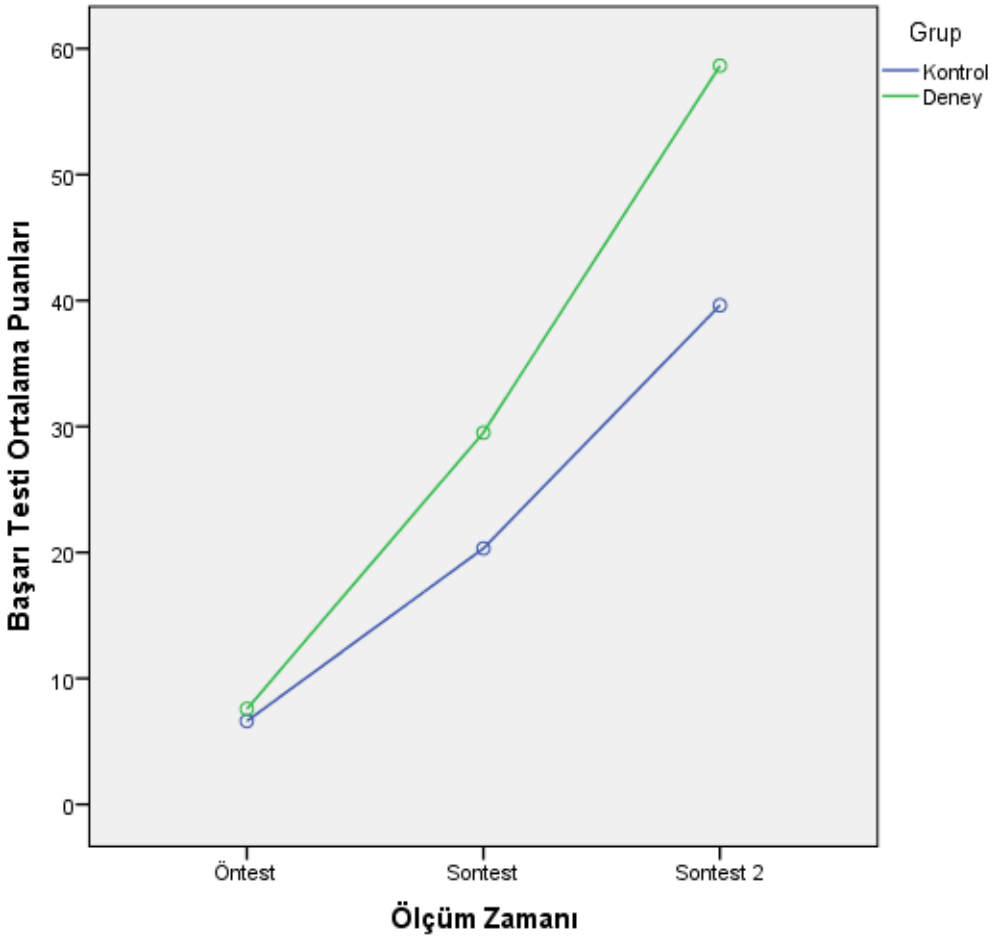
Ölçüm zamanı ve grup değişkeninin birlikte etkileşiminin daha iyi incelenebilmesi için bir diğer bağımlı değişken olan programlama başarısı için yapılan tek değişkenli varyans analizi sonuçları Tablo 13'te verilmiştir.

Tablo 13

Ölçme Zamanı ve Grup Değişkenine Göre Programlama Başarısı Ortalama Puanlarına İlişkin Tek Değişkenli Varyans Analizi Sonuçları

Varyansın Kaynağı	**KT**	**Sd**	**KO**	**F**	**p**	**η^2**
Gruplar arası						
Grup	3689.130	1	3689.130	11.021	.002	.181
Hata	16737.162	50	334.743			
Gruplar içi						
Ölçüm Zamanı	46290.716	1.232	37558.903	181.521	<.001	.784
Ölçüm Zamanı*Grup	2126.452	1.232	1725.340	8.339	.003	.143
Hata	12750.760	61.624	206.912			
Toplam	81594.22	115.088				

Tablo 13'e göre katılımcıların başarı puanları grup değişkenine göre anlamlı farklılık göstermektedir ($F_{(1,50)}$ =11.021, p<.025, η^2=.181). Buna göre katılımcıların başarı puanlarının grup değişkenine, yani öğretim yöntemi türüne göre farklılaştığı söylenebilir. Ayrıca etki büyüklüğü değeri incelendiğinde büyük bir etkiden söz edilebilir (Cohen, 1998). Ölçüm zamanı açısından incelendiğinde de başarı puanları arasında anlamlı farklılık olduğu görülmektedir ($F_{(1.232,\ 61.624)}$ = 181.521, p<.001,η^2=784). Buna göre katılımcıların başarı puanlarının öntest, sontest ve sontest 2 ölçümüne göre farklılaştığı söylenebilir. Etki değeri açısından da büyük bir etki söz konusudur (Cohen, 1998). Ölçüm zamanı ve grup değişkenin ortak etkisi incelendiğinde ise başarı puanları arasında anlamlı farklılık olduğu görülmektedir ($F_{(1.232,\ 61.624)}$ =8.339, p<.025,η^2=.143). Buna göre ölçüm zamanına göre kontrol ve deney grubunun başarı puanlarının değiştiği söylenebilir. Ortak etkinin daha ayrıntılı incelenebilmesi için ölçüm zamanına göre deney ve kontrol grubunun başarı puanları Şekil 12'de yer alan grafikte sunulmuştur.



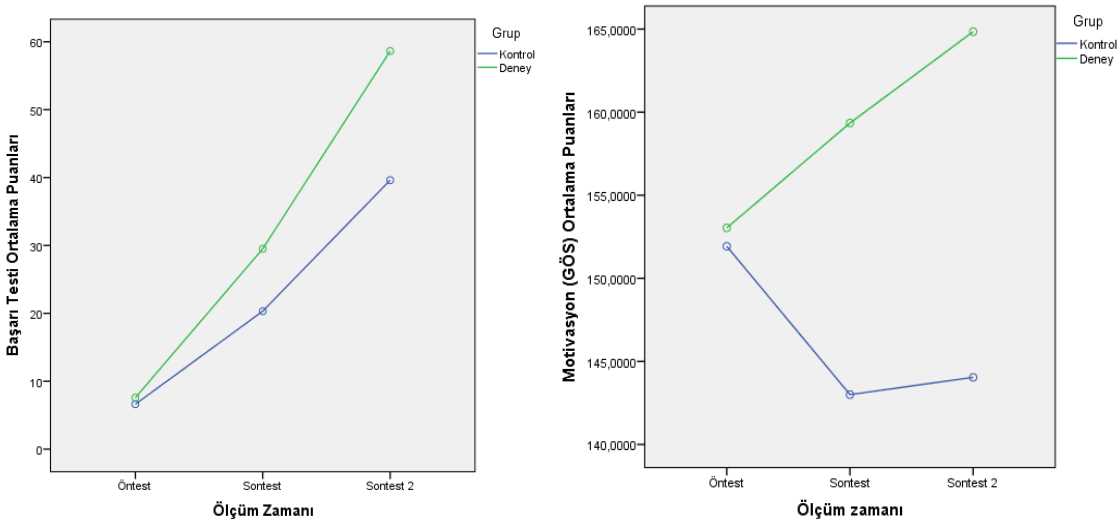
Şekil 12: Ölçüm zamanı ve gruplara göre başarı ortalama puanları grafiği

Şekil 12'de görüldüğü üzere hem deney hem de kontrol grubunda başarı puanı sürekli artış göstermiştir. Grup bağlamında incelendiğinde ise öntest puanlarının birbirine yakın, sontest ve sontest 2 puanlarının ise deney grubunda daha yüksek olduğu görülmektedir. Gözlenen bu durumların istatistiksel açıdan anlamlı olup olmadığının test edilmesi için basit temel etki analizi yapılmıştır.

İlk olarak grupların kendi içindeki başarı puanları ölçüm zamanına göre karşılaştırılmıştır (gruplar içi karşılaştırma). Bu bağlamda hem kontrol hem de deney grubunda; başarı öntest, sontest ve sontest 2 ortalama puanları arasında anlamlı farklılık bulunup bulunmadığını test etmek amacıyla her bir grupta ayrı ayrı olmak üzere bağımlı örneklemler t-testi yapılmıştır. Analiz her bir grupta ve her bir ölçme zamanı arasında ayrı ayrı yapıldığı için Bonferroni düzeltmesi yapılmıştır. Bundan dolayı anlamlılık düzeyi yapılan analiz sayısına bölünerek .025/6 = .004 olarak ele alınmıştır. Analiz sonucuna göre kontrol grubunda öntest ile sontest arasında sontest lehine ($t_{(25)}$=-7.427,

p<.001), sontest ile sontest 2 arasında sontest 2 lehine ($t_{(25)}$=-7.079, p<.001) ve öntest ile sontest 2 arasında sontest 2 lehine anlamlı farklılık vardır ($t_{(25)}$= -8.634, p<.001). Analiz sonucunda en yüksek puanın sontest 2' de alındığı, sontest ve öntest puanlarından anlamlı olarak yüksek olduğu görülmüştür. Buna göre kontrol grubundaki katılımcıların başarı puanlarının birinci ölçümden (öntest) üçüncü ölçüme (sontest 2) doğru anlamlı olarak arttığı söylenebilir. Deney grubunda da öntest ile sontest arasında sontest lehine ($t_{(25)}$=-12.054, p<.0001), sontest ile sontest 2 arasında sontest 2 lehine ($t_{(25)}$=-8.695, p<.0001) ve öntest ile sontest 2 arasında sontest 2 lehine anlamlı farklılık bulunmuştur ($t_{(25)}$=-11.756, p<.0001). Analiz sonucunda en yüksek puanın sontest 2' de alındığı, sontest ve öntest puanlarından anlamlı olarak yüksek olduğu görülmüştür. Buna göre deney grubundaki katılımcıların başarı puanlarının birinci ölçümden (öntest) üçüncü ölçüme (sontest 2) doğru anlamlı olarak arttığı söylenebilir. Hem deney grubunda hem de kontrol grubunda öntestten sontest 2 ölçümüne doğru anlamlı şekilde bir artış görülmüştür. Bu artışın hangi grupta anlamlı olarak daha yüksek olduğunu görmek için ölçümlerde elde edilen puanlar gruplara göre basit etki analizi ile karşılaştırılmıştır.

İkinci olarak programlama başarısı için farklı ölçümlerde elde edilen puanlar (öntest, sontest, sontest 2) gruplara göre karşılaştırılmıştır (gruplar arası karşılaştırma). Bu bağlamda başarı öntest, sontest ve sontest 2 puanlarının grup değişkenine göre farklılık gösterip göstermediğini test etmek amacıyla her bir ölçüm için bağımsız örneklem t-testi yapılmıştır. Analiz her bir ölçüm zamanı için ayrı ayrı yapıldığından Bonferroni düzeltmesi yapılmıştır. Bundan dolayı anlamlılık düzeyi yapılan analiz sayısına bölünerek .025/3 = .008 olarak ele alınmıştır. Analiz sonucuna göre deney ve kontrol grubu arasında başarı öntest puanları açısından istatistiksel açıdan anlamlı farklılık yok iken ($t_{(50)}$=-.589, p=.558); sontest puanları açısından ($t_{(50)}$=-3.666, p<.001) ve sontest 2 puanları açısından deney grubu lehine anlamlı farklılık vardır ($t_{(50)}$=-3.153, p<.008). Başarı puanları ilk başta her iki grupta aynı seviyede iken, sontest ve sontest 2 ölçümlerinde deney grubu lehine anlamlı bir farklılaşma olmuştur. Buna göre deney grubunda kullanılan öğretim yönteminin kontrol grubunda kullanılan yönteme göre öğrencilerin programlama başarılarına daha çok etki ettiği söylenebilir. Ölçüm zamanı ve gruplara göre başarı ve motivasyon ortalama puanlara ilişkin grafikler birlikte Şekil 13'de yer almaktadır.



Şekil 13: Ölçüm zamanı ve gruplara göre başarı ve motivasyon ortalama puanları grafiği

Yapılan varyans analizleri sonucunda ölçüm zamanı ve grup değişkeni etkileşiminin motivasyon ve başarı için anlamlı bir etki oluşturduğu görülmektedir. Ancak Şekil 13' te de görüldüğü üzere bu etkileşim katılımcıların motivasyon puanlarında farklı, başarı puanlarında farklı bir etki yaratmıştır.

Başarı puanları açısından karşılaştırıldığında hem deney hem de kontrol grubunda sürekli bir artış olduğu görülmektedir. Ancak gruplar arası puan fakları incelendiğinde grupların öntestlerde benzer başarı puan ortalamalarına sahip olduğu, sontestlerde deney grubunun kontrol grubundan 9.191 puan; sontest 2'de ise 19.025 puan yüksek olduğu görülmektedir (Tablo 10). Buradan da anlaşılacağı üzere gruplar arasında ki fark sürekli olarak deney grubu lehine anlamlı olarak artmıştır ($p<.001$). Buna göre hem deney grubu hem de kontrol grubunda uygulanan öğretim yönteminin başarı puanlarını artırdığı ancak gruplar arasında ki farkın deney grubu lehine olması nedeniyle deney grubunda uygulanan öğretim yöntemin başarıyı artırmada daha etkili olduğu söylenebilir. Ayrıca programlama için ortak öğretim yöntemin kullanıldığı ölçüm aralığında da (sontest-sontest 2 ölçüm aralığı) başarı puan artışın devam etmesinin nedeni kontrol ve deney grubunda kullanılan öğretim yöntemleri olabilir. Buna göre sontest 2 ölçümündeki puan farkı dikkate alındığında; deney grubu lehine anlamlı bir farkın olması, deney grubunda kullanılan yöntemin, programlama için ortak öğretim yönteminin kullanıldığı ölçüm aralığına (sontest-sontest 2 ölçüm aralığı) başarı açısından daha çok etki ettiği söylenebilir.

Motivasyon bağlamında ise deney grubunda sürekli bir artış yaşanırken, kontrol grubunda sontest ölçümünde 6.08 puanlık azalış, sontest 2 ölçümünde ise 1.04 puanlık artış görülmektedir. Öntest ile sontest 2 arasında ise 7.88 puanlık bir azalma olduğu görülmektedir (Tablo10). Buna göre kontrol grubunda katılımcıların motivasyon puanlarının sontest 2 sonunda, öntest puanlarına göre azaldığı ve birinci ölçümden üçüncü ölçüme doğru bir azalma olduğu söylenebilir. Ayrıca deney grubunda öntest ile sontest 2 arasında 11.81 puan bir artış olduğu görülmektedir (Tablo10). Buna göre deney grubunda katılımcıların motivasyon puanlarının sontest 2 sonunda öntest puanlarına göre arttığı ve birinci ölçümden üçüncü ölçüme doğru bir artış olduğu söylenebilir. Motivasyon puanları karşılaştırıldığında grupların öntestlerde benzer puana sahip olduğu, sontestlerde deney grubunun kontrol grubundan 16.65 puan, sontest 2' de ise 20.81 puan yüksek olduğu görülmektedir (Tablo10). Buradan da anlaşılacağı üzere gruplar arasında ki fark sürekli olarak deney grubu lehine anlamlı olarak artmıştır ($p<.001$). Ayrıca deney grubunda uygulanan yöntemin katılımcıların motivasyonunu sürekli arttırdığı, kontrol grubunda uygulanan yöntemin ise katılımcıların motivasyonu düşürdüğü, programlama öğretimi için ortak yöntemin kullanıldığı ölçüm aralığında ise (sontest- sontest 2 ölçüm aralığı) deney grubunda motivasyon bağlamında etkinin devam ettiği, kontrol grubunda ise durağanlığın olduğu söylenebilir.

Deney ve Kontrol Grubunda Yer Alan Katılımcıların Aldıkları Programlama Eğitimine İlişkin Görüşleri

Bu bölümde araştırmanın ikinci alt problemi olan ve katılımcıların aldıkları programlama eğitimleri hakkındaki görüşlerini irdeleyen araştırma sorusuna yanıt aranmıştır. Bu bağlamada hem ilk uygulamadan sonra (mantık öğretimi) hem de ikinci uygulamadan sonra (programlama öğretimi) yapılan odak grup görüşmesine ilişkin bulgular birlikte ele alınmıştır. Katılımcıların uygulama süresince aldıkları programlama eğitimine ilişkin görüşlerin belirlenmesi için görüşme soruları incelenmiş ve her bir soruya ilişkin tema ve alt temalar belirlenmiştir. Deney grubunda (DG) ve kontrol grubunda (KG) yer alan katılımcıların genel anlamda ders hakkındaki görüşlerini içeren, birinci görüşme sorusuna ait tema ve alt temalar Tablo 14' te belirtilmiştir.

Tablo 14

Katılımcıların Genel Anlamda Ders Hakkındaki Görüşlerini İlişkin Tema ve Alt Temalar

		Deney	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Olumlu	Eğlenceli Kolay Renkli	Kolay
	Olumsuz	Çok basit	Karmaşık ve zor Sıkıcı
II. Odak Grup Görüşmesi	Olumlu	Üst düzey Kolay Eğlenceli	Kolay Eğlenceli
	Olumsuz	Karmaşık ve zor Sıkıcı	Karmaşık ve zor

Tablo 14' te görüldüğü üzere deney ve kontrol grubunda yer alan katılımcıların ders hakkındaki görüşleri "olumlu" ve "olumsuz" olmak üzere iki temada toplandığı görülmektedir. Birinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların derse ilişkin görüşlerinin "olumlu" teması altında "eğlenceli ", "kolay " ve "renkli" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "kolay" alt temasında toplandığı görülmektedir. "Olumsuz" teması altında ise deney grubunda yer alan katılımcıların derse ilişkin görüşlerinin; "çok basit" alt temasında toplandığı, kontrol grubunda yer alan

katılımcıların görüşlerinin ise "karmaşık ve zor" ve "sıkıcı" alt temasında toplandığı görülmektedir.

Katılımcıların temel programlama yapılarının öğretildiği derse ilişkin olumlu görüşleri incelendiğinde; deney grubunda yer alan katılımcılar dersin eğlenceli, kolay ve renkli olduğunu belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise ders ile ilgili yalnızca kolay olduğuna ilişkin görüş belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların derse ilişkin olumlu görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Programlama lisede sıkıcıydı. Scratch ise zevkli geldi. Eğlenceliydi" (DGK-12) (Eğlenceli olması)

" Haftalar ilerledikçe programlamanın ne kadar kolay olduğunu anladım. (DGK-5) (Kolay olması)

" Bence ders çok kolay geçti. Zaten Scratch' te oyun yapmak basit. Scratch ile oyun falan tasarlarken programlamanın ne kadar kolay olduğunu gördüm"(DGK-19) (Kolay olması)

"Derse temel düzeyden başlamak iyi oldu, kolaylaştırdı" (KGK-13) (Kolay olması)

"Yani aslında umduğumdan basitti" (KGK-20) (Kolay olması)

"Scratch' te blokların renkleri var. O renkleri yerine yerleştirip eğer yapılarını oluşturuyorum mesela" (DGK-13) (Renkli olması)

Katılımcıların ders hakkındaki olumsuz görüşleri incelendiğinde; deney grubunda yer alan katılımcılar dersin çok basit olduğunu belirtirken, kontrol grubunda yer alan katılımcılar ise dersin zor, karmaşık ve sıkıcı olduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların derse ilişkin olumsuz görüşlerinin yer aldığı ifadeleri şu şekildedir:

"*Temel programlama için belki yeterli olabilir ama çok fazla basit olması ileri düzey programlama için yetmeyebilir"* (DGK-18) (Çok fazla basit olması)

"*Biraz karışıktı, adım adım yapmak gerekiyor ama benim kafamı çok karıştırdı"* (KGK-15) (Karmaşık ve zor olması)

"Problemleri çözemiyorum, çok zorladı. Çok karıştırdım ve mantığını anlamakta zorlandım " (KGK-25) (Karmaşık ve zor olması)

"Tahtadakileri sürekli yazmak, çizmek sıkıcıydı" (KGK- 24) (Sıkıcı olması)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların derse ilişkin görüşlerinin "olumlu" teması altında "üst düzey"," kolay" ve "eğlenceli" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise , "kolay" ve "eğlenceli" alt temalarında toplandığı görülmektedir. "Olumsuz" teması altında ise deney grubunda yer alan katılımcıların derse ilişkin görüşlerinin "karmaşık ve zor" ve "sıkıcı" alt temasında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "karmaşık ve zor" alt temasında toplandığı görülmektedir.

Katılımcıların C# programlama dillerinin öğretildiği derse ilişkin olumlu görüşleri incelendiğinde deney grubunda yer alan katılımcılar dersin üst düzey, kolay ve eğlenceli olduğunu belirtirken, kontrol grubunda yer alan katılımcılar ise dersin kolay ve eğlenceli olduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların derse ilişkin olumlu görüşlerinin yer aldığı ifadeleri şu şekildedir:

"C# daha üst düzey bir dil. Burada birşey yaparken daha özgürsünüz (DGK-18) (Üst düzey olması)

" İlk haftalarda kolaydı. Scratch zaten basitti. Sonra kodlara geçtik. Zor olabilir diye düşündüm. Ama kolay geldi. Tek fark yazıyoruz" (DGK-6) (Kolay olması)

"C# daha kolay geldi bana. Konsola yazmak programlamak daha kolay geldi." (KGK-4) (Kolay olması)

" İleri düzey birşey yapmak zevkliydi. Kod yazmak falan" (DGK-18) (Eğlenceli olması)

" Evet akış diyagramı temel kazandırdı ama şekiller çizmek çok sıkıcıydı. Böyle daha iyi. Konsol kullanmak iyi, bu daha çok zevkli şekillerde hep karıştırdım." (KGK-6) (Eğlenceli olması)

Katılımcıların ders hakkındaki olumsuz görüşleri incelendiğinde; deney grubunda yer alan katılımcılar dersin karmaşık, zor ve sıkıcı olduğunu ilişkin görüş bildirirken, kontrol grubunda yer alan katılımcılar ise dersin karmaşık ve zor olduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların derse ilişkin olumsuz görüşlerinin yer aldığı ifadeleri şu şekildedir:

" Scratch' te basit şekilde yapıyorduk. Bir anda geçiş sert oldu. Basitten zora çok hızlı geçtik gibi oldu." (DGK-17) (Karmaşık ve zor olması)

" Zaten algoritma zor gelmişti bana. Derken C#' a geçtik. Orada kafam iyice karıştı"(KGK-11) (Karmaşık ve zor olması)

"Beni çok zorladı. Kodlar falan. Karmaşık geldi. Problemleri iyice karıştırdım" (KGK-8) (Karmaşık ve zor olması)

"Genel iyiydi. Zaten program yapıyorduk bir nevi. Ama önceden problemler oyundu gibi yaptıydık, o eğlenceliydi. Şimdi bir iki problem yapınca sıkıyor. Yoruyor."(DGK-24) (Sıkıcı olması)

Deney grubunda ve kontrol grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerini içeren, ikinci görüşme sorusuna ait tema ve alt temalar Tablo 15' te yer almaktadır.

Tablo 15

Katılımcıların Ders Süresince Gerçekleştirilen Öğretme- Öğrenme Etkinliklerine İlişkin Görüşlerine Ait Tema ve Alt Temalar

Grup		**Deney**	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Üstünlükler	Programlama mantığını kazandırması Akılda kalıcılığı artırması Motivasyonu artırması Yaparak öğrenme olanağı sağlaması Ürün oluşturma olanağı sağlaması Öğrenmeyi kolaylaştırması Araştırarak öğrenme olanağı sağlaması Yaratıcılığı artırması	Programlama mantığını kazandırması Akılda kalıcılığı artırması
	Sınırlılıklar	Temel yapılar için yetersiz kalması	Temel yapılar için yetersiz kalması Öğrenenlerin pasif olması
II. Odak Grup Görüşmesi	Üstünlükler	Programlama yapabilme ortamı sağlaması	Uygulama yapma olanağı sağlaması Öğreneni aktif kılması
	Sınırlılıklar	Yalnızca kod yazmak zorunda kalınması	Yalnızca kod yazmak zorunda kalınması

Tablo 15' te görüldüğü üzere deney ve kontrol grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerinin "üstünlükler " ve "sınırlılıklar" olmak üzere iki temada toplandığı görülmektedir. Birinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerinin; "üstünlükler" teması altında "programlama mantığını kazandırması", "akılda kalıcılığı artırması", "motivasyonu artırması", "yaparak öğrenme olanağı sağlaması", "ürün oluşturma olanağı sağlaması", "öğrenmeyi kolaylaştırması", "araştırarak öğrenme olanağı sağlaması" ve "yaratıcılığı artırması" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "programlama mantığını kazandırması" ve "akılda kalıcılığı artırması" alt temalarında toplandığı görülmektedir. "Sınırlılıklar" teması altında ise deney grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerinin; "temel yapılar için yetersiz kalması" alt temasında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "temel yapılar için yetersiz kalması" ve "öğrenenlerin pasif olması" alt temasında toplandığı görülmektedir.

Katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin sağladığı üstünlüklere ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar Scratch ile oyun tasarım etkinliklerinin; programlama mantığını kazandırdığını, öğrenmeyi kolaylaştırdığını, akılda kalıcılığı, motivasyonu ve yaratıcılığı artırdığını, yaparak öğrenme olanağı, ürün oluşturma olanağı ve araştırarak öğrenme olanağı sağladığını belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme etkinliklerinin programlama mantığını kazandırdığını ve akılda kalıcılığını artırdığını belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların gerçekleştirilen öğretme-öğrenme etkinliklerinin sağladığı üstünlüklere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *".....sıfırdan başlayanlar için bence ideal. Çünkü böylece adım adım nasıl çalışacağını ve neyi nerede kullanacağımızı gördük. Algoritmayı anladık."* (DGK-24) (Programlama mantığını kazandırması)
>
> *"Böylelikle en azından temel kavramları aldık temelimiz oluştu. Bu iyi oldu."*(KGK-2) (Programlama mantığını kazandırması)

"Mesela diziler konusunda ekrandaki bloklar, ekran aklıma geliyor. Nasıl yapacağımı düşünüyorum. Problemi çözerken direk aklıma geliyor" (DGK-24) (Akılda kalıcılığı artırması)

"Şekiller algoritmanın aklıma gelmesini sağladı" (KGK- 4) (Akılda kalıcılığı artırması)

"Scratch yaptırıyordu problemi. Yani yapmamı istiyordu. Tasarım yapıyorum. Ses ekliyorum, bir şey yapıyorum yani, bu hoşum gitti aslında..." (DGK-4) (Motivasyonu artırması)

"Mesela siz anlattınız ama biz sonra uyguladık, oyun tasarladık. Çoğu zaman biz yaptık. Eğer bakıp geçseydik daha zor olurdu. Uygulamak iyiydi" (DGK-17) (Yaparak öğrenme olanağı sağlaması)

"Oyun tasarlamak iyiydi. Hep oynadığımız oyunları biz tasarladık..." (DGK-22) (Ürün oluşturma olanağı sağlaması)

"Algoritma süreci biraz karışık gelirdi. Scratch yardımıyla algoritmayı daha kolay algılamaya başladım" (DGK-2) (Öğrenmeyi kolaylaştırması)

"Oyun yapmak iyiydi en azından yazmadık. Mesela bilmediğimiz şeyleri araştırdık öğrendik" (DGK-1) (Araştırarak öğrenme olanağı sağlaması)

"Oyundan önce senaryo yazdık. Tamamen bizim hayalimizde olan. Sonra bunları kendi fikirlerimiz doğrultusunda oluşturduk. Bir şeyler yarattığımızı gösteriyordu" (DGK-13) (Yaratıcılığı artırması)

Katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin oluşturduğu sınırlılıklara ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar Scratch ile oyun tasarım etkinliklerinin bazı temel yapılar için yetersiz kaldığını belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme etkinliklerinin bazı temel yapılar için yetersiz kaldığını ve öğreneni pasif yaptığını belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların gerçekleştirilen öğretme-öğrenme etkinliklerinin oluşturduğu sınırlılıklara ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

" Herşeyi yapamadık. Dizilerde sıkıntı yaşadım. Bloklarını anlamadım. Bence o noktada yetersiz kaldı" (DGK-17) (Temel yapılar için yetersiz kalması)

"Döngüleri iç içe girince karıştı. Akış diyagramı orada yetersiz hatta kafa karıştırıcı" (KGK-4) (Temel yapılar için yetersiz kalması)

"Ben kendimi çok pasif hissetim. Sadece yazdım ders boyunca." (KGK-24) (Öğrenenlerin pasif kalması)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerinin; "üstünlükler" teması altında "programlama yapabilme ortamı sağlaması" alt temasında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "uygulama yapma olanağı sağlaması" ve "öğreneni aktif kılması" alt temalarında toplandığı görülmektedir. "Sınırlılıklar" teması altında ise hem deney grubunda hem de kontrol grubunda yer alan katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşlerinin "yalnızca kod yazmak zorunda kalma" alt temasında toplandığı görülmektedir.

Katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin sağladığı üstünlüklere ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar C# ile programlama öğretimi etkinliklerinin programlama yapabilme ortamı sağladığını belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise C# ile programlama öğretimi etkinliklerinin öğrenenler için uygulama yapma olanağı sağladığını ve öğreneni aktif kıldığını belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların C# programlama öğretimi süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin sağladığı üstünlüklere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Scratch te oyun yapıyorduk. Aslında oda programdı bir nevi. C# da bir farkı yok ama programlama yapma hissi veriyor." (DGK-4) (Programlama yapabilme ortamı sağlaması)

"C# da en güzel programlama yapmaktı. Zorlasa da program yapabilmek güzeldi" (DGK-16) (Programlama yapabilme ortamı sağlaması)

"Bence C#' ın en iyi tarafı uygulama yapabiliyor olmaktı" (KGK-1) (Uygulama yapma olanağı sağlaması)

"Belki akış diyagramı gerekliydi ama konsol bize uygulama yapmamızda yardımcı oldu. Şimdi algoritmanın uygulamasını yapıyoruz" (KGK-2) (Uygulama yapma olanağı sağlaması)

"Önceleri yazıp geçiyorduk. Ama şimdi siz yazıyorsunuz, biz de yapıyoruz. Yani aktif hale geldik en azından"(KGK-6) (Öğreneni aktif kılması)

Katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin oluşturduğu sınırlılıklara ilişkin görüşleri incelendiğinde; hem deney grubunda hem de kontrol grubunda yer alan katılımcılar C# ile programlama öğretimi etkinliklerinin yalnızca kod yazmak zorunda kalma durumunun sınırlılık oluşturduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların C# programlama öğretimi süresince gerçekleştirilen öğretme-öğrenme etkinliklerinin oluşturduğu sınırlılıklara ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *"Scratch' te güzeldi Oyun falan yapmıştık. Ses ekledik falan. C# sadece ekranda kod yazmak sıktı biraz"*(DGK-14) (Sadece kod yazmak zorunda kalınması)
>
> *"Aslında iyi ama siyah ekranda hep kod yazıyoruz. Farklı birşey yapmıyoruz. Problemleri sadece bu sefer burada yazıyoruz."* (KGK-26) (Sadece kod yazmak zorunda kalınması)

Deney grubunda ve kontrol grubunda yer alan katılımcıları ders süresince güdüleyen veya zorlayan durumlara ilişkin görüşlerini içeren üçüncü görüşme sorusuna ait tema ve alt temalar Tablo16'da yer almaktadır.

Tablo 16

Katılımcıları Ders Süresince Güdüleyen veya Zorlayan Durumlara İlişkin Görüşlere Ait Tema ve Alt Temalar

Grup		**Deney**	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Motivasyonu artıran etmenler	Eğlenceli ve zevkli olması Görsel olması Test etme ve dönüt olanağı sağlaması Uygulamalı olması Bir ürün ortaya koyma Kod gerektirmemesi Basitleştirmesi Zihinde canlandırması Birlikte yapma ve akran iletişimi sağlaması Dersin devamlılığını sağlaması Merakı artırması Canlandırma olanağı	Görsel olması Basitten zora doğru gitmesi Zihinde canlandırması
	Motivasyonu düşüren etmenler	Bazı temel yapıların mantığını anlayamama Temel yapıların iç içe girmesi Tasarımın sınırlı kalması	Bazı temel yapıların mantığını anlayamama Temel yapıların iç içe girmesi Test etme ve dönüt olanağı olmaması Süreç içinde aktif olamama Uygulamasız olması Temel yapıların nerede kullanacağını bilememe Farklı sonuçların çıkması Adım adım mantığını karıştırma Sürecin uzun olması
II. Odak Grup Görüşmesi	Motivasyonu artıran etmenler	İlk başta mantığı anlamış olma Program yapma	Süreç içinde aktif olma Test etme ve dönüt olanağı
	Motivasyonu düşüren etmenler	Konsol ekranında çalışma Kod hataları ile uğraşma	Kod hataları ile uğraşma Bazı temel kavramların mantığını anlayamama Problemin analizinde sıkıntı yaşama Kavramların iç içe girmesi

Tablo 16' da görüldüğü üzere deney ve kontrol grubunda yer alan katılımcıları ders süresince güdüleyen ve zorlayan durumlar "motivasyonu artıran etmenler" ve "motivasyonu düşüren etmenler" olarak iki temada toplanmaktadır. Birinci odak grup

görüşmesine ilişkin bulgular incelendiğinde; deney grubunda yer alan katılımcıları güdüleyen durumlara ilişkin görüşlerinin "motivasyonu artıran etmenler" teması altında "eğlenceli ve zevkli olması", "görsel olması", "test etme ve dönüt olanağı sağlaması", "uygulamalı olması", "bir ürün ortaya koyma", "kod gerektirmemesi", "basitleştirmesi", "zihinde canlandırması", "birlikte yapma ve akran iletişimi sağlaması", dersin devamlılığını sağlaması", "merakı artırması" ve "canlandırma olanağı" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "görsel olması", "basitten zora doğru gitmesi" ve "zihinde canlandırması" alt temalarında toplandığı görülmektedir. "Motivasyonu düşüren etmenler " teması altında ise deney grubunda yer alan katılımcıları zorlayan durumlara ilişkin görüşlerinin; "bazı temel yapıların mantığını anlayamama", "temel yapıların iç içe girmesi" ve "tasarımın sınırlı kalması" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "bazı temel yapıların mantığını anlayamama", "temel yapıların iç içe girmesi", "test etme ve dönüt olanağı olmaması", "süreç içinde aktif olamama", "uygulamasız olması", "temel yapıların nerede kullanacağını bilememe", "farklı sonuçların çıkması", "adım adım mantığını karıştırma" ve "sürecin uzun olması" alt temalarında toplandığı görülmektedir.

Katılımcıların ders süresince güdüleyen durumlara ilişkin görüşleri incelendiğinde, deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinde; sürecin eğlenceli, zevkli, görsel ve uygulamalı olmasını, kavramları algılamayı basitleştirmesini, test etme ve dönüt olanağı sağlamasını, kod gerektirmeyen bir yapısının olmasını, dersin devamlılığını sağlamasını, süreç içinde bir ürün ortaya çıkarmayı, tasarım sırasında zihinde canlandırma olanağı sağlamasını, birlikte yapma ve akran iletişimi sağlamasını, yapılan değişiklikleri hareketli olarak canlandırmasını motivasyonu artıran etmenler olarak belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme etkinliklerinde; sürecin görsel olmasını, basitten zora doğru gitmesini ve zihinde canlandırma olanağı sağlamasını motivasyonu artıran etmenler olarak belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların motivasyonlarını artıran etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Tabiki çok zevkliydi. Yani artık oyunlaştırma işine girmiştim eğlenceli olması için. Mesela siz ara veriyordunuz biz çıkmıyorduk. Oyunu yapmak için uğraşıyorduk." (DGK-4) (Eğlenceli ve zevkli olması)

" Ben zaten görsel öğrenen biriyim. Scratch bu konuda şüphesiz güzel bence." (DGK-13) (Görsel olması)

"Beğendiğim yönler ise şekil çiziyoruz, görebiliyoruz..." (KGK-4) (Görsel olması)

"Scratch ile problemi yapmak daha iyi anlamamızı sağladı. Yaptık, denedik, gördük, olmadı..."(DGK-12) (Test etme ve dönüt olanağı sağlaması)

"İlk önce siz anlattınız sonra bizden uygulama yapmamızı, bunları uygulamaya dökmemizi istemeniz gayet pratik için iyiydi." (DGK-14) (Uygulamalı olması)

"Aklımdan geçenleri tasarlamaya çalıştım. Ses ekledim. Bunlardan bir bütün oluşturdum….bir ürün yapmak mutlu etti hepimizi…" (DGK-13) (Bir ürün ortaya koyma)

"…ben Scratch'i görünce Flash programına benzettim. Scratch te kod yok daha kullanışlı sürükleyerek yapıyorsunuz" (DGK-11) (Kod gerektirmemesi)

"Aslında hocam tek tek öğrendik. Her yeni yapı problemini uzun yoldan değil de bize kolay yoldan çözmemizi öğretti" (DGK-19) (Basitleştirmesi)

"Scratch programında mesela dizilerde a=10 yazdığımızda, sağdaki köşede liste gözümün önüne geliyor, kafamda canlandırıyorum" (DGK-24) (Zihinde canlandırması)

"Oyunları birlikte yaptık, aramızda iş bölümü oluşturduk. O da iyi oldu" (DGK-19) (Birlikte yapma ve akran iletişimi)

"Evet iyi oldu. Çoğu zaman birbirimize danıştık. Ben yapamayınca DGK-4' den yardım aldım" (DGK-5) (Birlikte yapma ve akran iletişimi)

"Hocam yaptık denedik gördük. Yapamadık size sorduk, arkadaşıma sordum. Hiç olmadı evde tasarladık oyunu, orada uğraştık, yaptık."(DGK-12) (Dersin devamlılığını sağlaması)

"Bir şeyler yapıyorsunuz. Ekliyorsunuz. Basit algoritmadan birşey oluşuyor sonra daha fazla yapsam olacak diyorsunuz. Yenisi merak ediyorsunuz" (DGK-4) (Merakı artırması)

"Blokları ekliyorsunuz. Yapıyorsunuz işlemleri, aynısı çıkıyor. Görüyorsunuz yaptığınızı" (DGK-21) (Canlandırma olanağı)

"İlk başta problemlerin önce basit olması iyi. Problemleri yapabiliyoruz... yani basitten zora doğru gitmesi güzel oldu" (KGK-9) (Basitten zora doğru gitmesi)

"*Akış diyagramları çizerken neyin nereden bağlanacağını bilmek problemi çözmek açısından önemli. Göz önüne getirmeyi sağlıyordu*" (KGK-15) (Zihinde canlandırması)

Katılımcıların ders süresince zorlayan durumlara ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinde; bazı temel yapıların mantığını anlayamamalarını, temel yapıların iç içe girmesini ve tasarımın sınırlı kalmasını motivasyonu düşüren etmenler olarak belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise, akış diyagramları ile problem çözme etkinliklerinde bazı temel yapıların mantığını anlayamamalarını, temel yapıların iç içe girmesini, test etme ve dönüt olanağı olmamasını, süreç içinde aktif olamamalarını, sürecin uygulama içermemesini, temel yapıların nerede kullanılacağını karıştırmalarını, problemlerin farklı sonuçlarının çıkmasını, adım adım mantığını karıştırmalarını ve akış diyagramına ayrılan sürenin uzun olmasını motivasyonlarını düşüren etmenler olarak belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların motivasyonlarını düşüren etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

Döngülere kadar iyiydi. Dizilere geçince hele iyice kafam karıştı, zorladı" (DGK-20) (Bazı temel yapıların mantığını anlayamama)

"Döngüleri ve dizileri hiç anlamadım. Beni çok zorladı. Halende anlamış değilim" (KGK-17) (Bazı temel yapıların mantığını anlayamama)

"Ben döngülere kadar iyi anladım. Dizlerde kafam karıştı. Karmaşık yapılar belki diğerlerini de içine alınca karıştı, birlikte kullanınca." (DGK-7) (Temel yapıların iç içe girmesi)

"Sona doğru hepsi birbirine girdi. Hepsini birlikte kullanmaya başladık." (KGK-13) (Temel yapıların iç içe girmesi)

"Problemi çözüyorum ama doğrumu yanlış mı anlamıyorum." (KGK-7) (Temel yapıların iç içe girmesi)

"Biz önce senaryolarda herşeyi yapacağız diye yazdık. Oyun tasarlarken her şeyi yapamadık. Aklımızdan geçen her şeyi dönüştüremedik. Scratch yetersiz kaldı. Uğraşmadım sonra" (DGK-22) (Tasarımın sınırlı kalması)

"İlk başta iyiydi. İlerledikçe zorlaştı, problemi yazdım ama yaptıklarımın sonucu göremedim. Yani yapıyoruz ama ne yapıyoruz" (KGK-9) (Test etme ve dönüt olanağı olmaması)

"Ben sadece yazıyorum tahtayı, sadece problem çözüyoruz. Problemler iyi ama belki biz de çıkıp tahtada yapsak böylelikle daha aktif olacağız…problemler iyi ama pasif kalıyoruz" (KGK-5) (Süreç içinde aktif olamama)

"Siz problemi yazdıktan sonra, bizde kâğıda yazıyoruz. Sonra çözmeye çalışıyoruz. Ama yazıyoruz sonra. Siz anlatıyorsunuz onu da yazıyoruz. Biz sadece tahtadakini yazıyoruz ya da çiziyoruz. Etkin olsak biraz daha iyi olurdu" (KGK-21) (Süreç içinde aktif olamama)

"Bilgisayar kullanabilirdik. Böyle zor oluyor. Uygulama yok. Sürekli yazmak çizmek ben beğenmedim" (KGK-24) (Uygulamasız olması)

"Tam mantığını anladım derken, yeni bir problemde yine sıkıntı yaşıyorum. Döngüyü diziyi bilsem de neyi nerede nasıl kullanacağımı bilemiyorum" (KGK-23) (Temel yapıların nerede kullanacağını bilememe)

"Arkadaşla baktığımızda karşılaştırdığımızda onlar mı doğru yapıyor yoksa biz mi doğru yapıyor bilemedik. Kime göre doğru kime göre yanlış, o yüzden sıkıntı oldu biraz" (KGK-6) (Farklı sonuçların çıkması)

"Hangi sıra nasıl geliyor anlamadım. Hangi sırada hangi işlem olacak anlamıyorum. Ezberlemek zorunda kalıyorum" (KGK-16) (Adım adım mantığını karıştırma)

"Algoritma için bence sekiz hafta çok uzun. Bana algoritma için çok fazla geldi" (KGK-24) (Sürecin uzun olması)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıları güdüleyen durumlara ilişkin görüşlerinin "motivasyonu artıran etmenler" teması altında; "ilk başta mantığı anlamış olma" ve "program yapma" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "süreç içinde aktif olma" ve "test etme ve dönüt olanağı" alt temalarında toplandığı görülmektedir. "Motivasyonu düşüren etmenler" teması altında ise deney grubunda yer

alan katılımcıları zorlayan durumlara ilişkin görüşlerinin; "konsol ekranında çalışma" ve "kod hataları ile uğraşma" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "kod hataları ile uğraşma", "bazı temel kavramların mantığını anlayamama", "problemin analizinde sıkıntı yaşama" ve "kavramların iç içe girmesi" alt temalarında toplandığı görülmektedir.

Katılımcıların ders süresince güdüleyen durumlara ilişkin görüşleri incelendiğinde, deney grubunda yer alan katılımcılar C# ile programlama öğretimi etkinliklerinde; bir önceki öğretim sürecinde (Scratch) programlama mantığını öğrenmiş olmanın ve kod kullanarak programlama yapmanın motivasyonu artıran etmenler olduğunu belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise C# ile programlama öğretimi etkinliklerinde; süreç içinde aktif olmaları ile test etme ve dönüt olanağının olmasını motivasyonu artıran etmenler olarak belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların motivasyonlarını artıran etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Evet aslında önceden mantığını öğrendik, yazarken hemen aklınıza geliyor. Zaten önce mantığını yapınca C # da hemen aklınıza geliyor yapabiliyorsunuz." (DGK-24) (İlk başta mantığı anlamış olma)

"İlk başta yazmamıştım hep sunuya baktım. Sonra baktım ki yazıyorum artık. Hatta sizinkinden farklı yolla yapıyorum. Kendi mantığımla yapıyorum. Kod yazarak bir şeyler oluşturabilmek, programlayabilmek çok güzeldi" (DGK-3) (Program yapma)

" ...ilk haftalar bir şey yapmıyorduk. Bize daha sonra konsolda uygulama yapmak daha iyi geldi. " (KGK-25) (Süreç içinde aktif olma)

"Konsolda yazınca anında görebiliyorum, test etmek açısından bana faydası çok oldu. Konsol hataları size söylüyor" (KGK-7) (Test etme ve dönüt sağlama)

Katılımcıların ders süresince zorlayan durumlara ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar C# ile programlama öğretimi etkinliklerinde; sürekli konsol ekranında çalışmanın ve kod hataları ile uğraşmak zorunda kalmanın motivasyonu düşüren etmenler olduğunu belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise C# ile programlama öğretimi etkinliklerinde; kod hataları ile uğraşmanın, daha önce bazı temel kavramların mantığını anlayamamış olmanın, problemin analizinde yaşanan sıkıntıların ve kavramların iç içe girmesinin

motivasyonu düşüren etmenler olduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların motivasyonlarını düşüren etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Siyah ekran hiç hoşuma gitmedi. İçimi kararttı. Konsola sadece kod yazdık. Sıktı beni" (DGK-2) (Konsol ekranında çalışma)

"Scratch' in avantajı şuydu. Hata yaptırmıyordu. Sadece en son mantık hatası olabiliyordu. Konsolda ise sürekli kod hatası oluyor. Sürekli onu düzeltiyorum." (DGK-15) (Kod hataları ile uğraşma)

"Kodlar İngilizce. Zaten yazamıyorsun. Bir de baş harfi büyük mü küçük mü sıkıntısı var. Birşeyi yanlış yazınca hata veriyor"(KGK-15) (Kod hataları ile uğraşma)

"Beni diziler ve iç içe döngüler zorladı. Ben zaten önceden de mantığını anlamamıştım. "(KGK-16) (Bazı temel kavramların mantığını anlayamama)

"Tahtaya bakarak yazıyorum. Ama yeni bir problemle karşılaşınca yapamıyorum" (KGK-25) (Problemin analizinde sıkıntı yaşama)

"Ben yine diziyi anlamadım. Aslında hepsi birbirine girdiği için. Birden fazla şey kullanıyoruz çözmek için. İç içe giriyor yani" (KGK-2) (Kavramların iç içe girmesi)

Deney grubunda ve kontrol grubunda yer alan katılımcıların ders süresince öğrendiklerini kullanma durumlarına ilişkin görüşlerini içeren dördüncü görüşme sorusuna ait tema ve alt temalar Tablo17'de verilmiştir.

Tablo 17

Katılımcıların Ders Süresince Öğrendiklerini Kullanma Durumlarına İlişkin Görüşlerine Ait Tema ve Alt Temalar

		Deney	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Yeni bir programlama diline transfer etme	Temel mantığı sürdürme Test etme için kullanma	Temel mantığı sürdürme
	Başka bir derse transfer etme	Eğitim materyali olarak kullanma	
II. Odak Grup Görüşmesi	Geçmiş deneyimi transfer etme	Mantığı kavramaya yardımcı olması Eğlenceli olması Uygulamalı olması Kolaylaştırması Zihnimde canlandırması Mantıksal hatayı azaltması Ön yargıları yıkması	Mantığı kavramaya yardımcı oldu Kolaylaştırması
	Geçmiş deneyimi transfer etmeme		Mantığı kavrayamama Sonucu görememe Pasif kalma

Tablo 17' de görüldüğü üzere deney ve kontrol grubunda yer alan katılımcıların ders süresince öğrendiklerini ileride nasıl kullanacaklarını ilişkin görüşleri "yeni bir programlama diline transfer etme" ve " başka bir derse transfer etme" olmak üzere iki temada toplandığı görülmektedir. Birinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların görüşleri "yeni bir programlama diline transfer etme" teması altında "temel mantığı sürdürme" ve "test etmek için kullanma" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "temel mantığı sürdürme" alt temasında toplandığı görülmektedir. "Başka bir derse transfer etme" teması altında ise deney grubunda yer alan katılımcıların görüşlerinin; "eğitim materyali olarak kullanma" alt temasında toplandığı görülmektedir. Kontrol grubunda yer alan hiçbir katılımcı bu temaya ilişkin görüş belirtmemiştir.

Katılımcıların ders süresince öğrendiklerini ileride nasıl kullanacaklarına ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı sürecinde edindikleri bilgileri yeni bir programlama diline transfer etmek için, temel mantığı sürdürmek ve test etme amaçlı kullanacaklarını belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme sürecinde edindikleri bilgileri, yeni bir programlama diline transfer etmek için temel mantığı sürdürme amaçlı kullanacaklarını belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların ders süresince öğrendiklerini yeni bir programlama diline transfer etme amaçlı kullanımına ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

" *Bir kere zaten programını çözmek için önce mantığını kafamızda oluşturmak gerekiyor. Kodları tamamen bilsek bile hangi kodu nerede kullanacağımızı bilmezsek bir etkisi olmaz. Yani burada da mantığı öğrendik neyi nerde nasıl ne zaman kullanmamız gerektiğini öğrendik. Yani bir daha ki derslerde dili öğreneceğiz, yani birşey değişmeyecek. Kafamızda mantığını canlandıracağız"* (DGK-4) (Temel mantığı sürdürme)

"*Şimdi gördüklerimiz Türkçeydi. Sonra İngilizcesini görmüş gibi olacağız. Kod göreceğiz. Mantık var işlem sırası var aynı. Sonuçta temelini attık zaten*" (DGK-19) (Temel mantığı sürdürme)

"...ileriki programlama dersinde tekrar eğer gelecek, döngü gelecek. Bu sefer for olacak. Herşey aynı olacak. Gördüklerimizin İngilizcesi" (KGK-7) (Temel mantığı sürdürme)

"Algoritma kafamızda oluşunca kodun karşılığını bilebilirim belki. Paralel olabilir" (KGK-25) (Temel mantığı sürdürme)

"*Mesela ilerde yeni bir programlama öğrenme başlasam karşılaştığım problemi önce Scratch' te yapmaya çalışırım. Eğer yeniysem, denerim en azından"* (DGK-8) (Test etme için kullanma)

Katılımcıların ders süresince öğrendiklerini ileride nasıl kullanacaklarına ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecinde edindikleri bilgileri başka bir derse transfer etmek için, eğitim materyali tasarlamak amaçlı kullandıklarını belirtmişlerdir. Kontrol grubunda yer alan hiçbir katılımcı bu temaya ilişkin görüş belirtmemiştir. Deney grubunda yer alan

katılımcıların ders süresince öğrendiklerini başka bir derse transfer etme amaçlı kullanımına ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *"Biz materyal dersinde kullanmaya başladık. Bunun desteğini alarak bu derste materyal yaptık. Bir oyun yaptık"* (DGK-3) (Eğitim materyali olarak kullanma)
> *"Eğitimde materyal geliştirme dersinde kullandım. İleride de kullanırım. Bir materyal olarak oyun tasarladık bunun için kullandık. Görsel olması ve kolay olmasından dolayı seçtik"* (DGK- 8) (Eğitim materyali olarak kullanma)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney ve kontrol grubunda yer alan katılımcıların ders süresince (Scratch/akış diyagramı) öğrendiklerini C# programlama dersinde nasıl kullandıklarına ilişkin görüşleri "geçmiş deneyimi transfer etme" ve "geçmiş deneyimi transfer etmeme" olmak üzere iki temada toplandığı görülmektedir. Veriler incelendiğinde deney grubunda yer alan katılımcıların görüşleri "geçmiş deneyimi transfer etme" teması altında "mantığı kavramaya yardımcı olması", "eğlenceli olması", "uygulamalı olması", "kolaylaştırması", "zihinde canlandırması", "mantıksal hatayı azaltması", "ön yargıları yıkması" alt temalarında, kontrol grubunda yer alan katılımcıların görüşlerinin ise "mantığı kavramaya yardımcı olması" ve "kolaylaştırması" alt temalarında toplandığı görülmektedir. "Geçmiş deneyimi transfer etmeme" temasında ise kontrol grubunda yer alan katılımcıların görüşlerinin "mantığı kavrayamama", "sonucu görememe" ve "pasif kalma" alt boyutlarında toplandığı görülmektedir.

Katılımcıların ders süresince (Scratch/akış diyagramı) öğrendiklerini C# programlama dersinde nasıl kullandıklarına ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcılar geçmiş öğrenme deneyimlerini C# programlama dersine transfer etmelerine Scratch ile oyun tasarımı sürecinin yardımcı olduğunu belirtmişlerdir. Katılımcılar bununla ilgili olarak; oyun tasarım sürecinin programlama mantığını kavramaya yardımcı olduğunu, eğlenceli olduğunu, programlama dilini öğrenmeyi kolaylaştırdığını, zihinde canlandırmaya yardımcı olduğunu, mantıksal hatayı azalttığını, programlamayla ilgili ön yargıları yıktığını ve uygulama olmasının birer üstünlük olduğunu belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise geçmiş öğrenme deneyimlerini C# programlama dersine transfer etmelerine akış diyagramı ile problem çözme sürecinin, programlama mantığını ve programlama dilini öğrenmeyi kolaylaştırma bağlamında yardımcı olduğunu belirtmişlerdir. Deney

grubunda ve kontrol grubunda yer alan katılımcıların geçmiş öğrenme deneyimlerini yeni bir programlama diline transfer etme amaçlı kullanımına ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

" Kolay gelmesinin sebebi bence Scratch ile mantığını öğrenmiş olmamız. Zaten değişkenleri falan Scratch te oturttuk. Buraya gelince de zaten doğrudan kodları yazdık. " (DGK-24) (Mantığı kavramaya yardımcı olması)

"Scratch zaten bizim algoritma mantığını öğrenmemize faydalı oldu amacı da oydu. C# da zaten algoritmasını çıkarınca programlamada gerisi otomatik oldu "(DGK-2) (Mantığı kavramaya yardımcı olması)

"Önce algoritmasını görüp sonra koduna geçmemiz iyi oldu. Çünkü eğer orada yapabilirsek, diğer türlüde yapabiliriz" (KGK-2) (Mantığı kavramaya yardımcı olması)

" ...döngüyü, eğeri işte değişkeni çok rahat öğrendik. Geniş süreye yayıldı, rahat ve eğlenceliydi. Yani oyun yapmıştık hep aslında. C# a geçsek de mantık hep aklımızda kaldı." (DGK-26) (Eğlenceli olması)

"Devam etti çünkü Scratch ile doğrudan hep uygulama yaptık. Denedik gördük öğrendik, doğrusunu öğreniyorsun yani." (DGK-14) (Uygulamalı olması)

"Scratch programlamayı çok basitleştirdi. Zaten mantığını kavramında kod yazmak kolay. Kodları bilmek yeterli, gerisi tamamen kolay. İş algoritmasını yazmakta bitiyor" (DGK-19) (Kolaylaştırması)

"Ben doğrudan kod öğrenseydim zorlanırdım. Çünkü ben hiç programlama bilmiyorum. Akış diyagramları ile başlamak benim için iyi oldu. Eğer algoritmasını çıkartırsam kodunu zaten kolaylıkla yazabiliyorum" (KGK-9) (Kolaylaştırması)

"Kod yazarken fark ettim hızlanmışım. Bence bu Scratch sayesinde oldu. Çünkü sınavda da öyle oldu kodları yazmadan önce algoritmasını yazdım. Aklıma hep Scratch teki bloklar geliyordu, gözümün önüne geliyordu. Onları kodlara çevirdim işte" (DGK-7) (Zihinde canlandırması)

"Aslında artık sadece kod hatası yapıyoruz onu fark ettim. Önceden o yoktu. Mantıksal hata yapıyorduk. Ama şimdi artık demek ki mantıkta bir sıkıntım yok." (DGK-13) (Mantıksal hatayı azaltması)

"Mesela programlamanın aslında Scratch' te basit olduğunu gördük. O yüzden C# geçince bende bir önyargı oluşmadı" (DGK-11) (Ön yargıları yıkması)

Katılımcıların ders süresince (Scratch/akış diyagramı) öğrendiklerini C# programlama dersine transfer etmemelerine ilişkin görüşleri incelendiğinde, kontrol grubunda yer alan katılımcıların akış diyagramı ile problem çözme sürecinde programlama mantığını kavrayamadıkları, sonucu göremedikleri ve pasif kaldıkları için geçmiş deneyimlerini yeni bir programlama diline transfer edemediklerini belirtmişlerdir. Deney grubunda yer alan hiçbir katılımcı bu temayla ilgili görüş belirtmemiştir. Kontrol grubunda yer alan katılımcıların geçmiş öğrenme deneyimlerini yeni bir programlama diline transfer etmeme nedenlerine ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Kod yazmakta sıkıntı yok ezberlersin geçer. Ama algoritmasını yazmak öyle değil. Onu yapamadın mı kod da yazamıyorsun. Ben o noktada çok sıkıntı yaşıyorum. Bakarak yazabiliyorum ancak. Biraz ezberleme oluyor" (KGK-13) (Mantığı kavrayamama)

"Şimdi yazdığımda hatayı görüyorum. Ama önceki haftalar hiçbir zaman sonucunu doğru yanlış bilemedik. Belki de hep yanlış öğrendim. Dizileri halen ondan anlamıyorum" (KGK-21) (Sonucu görememe)

"Akış diyagramında siz yazıyordunuz biz geçiriyorduk. Şimdi bizde yazıyoruz. Ama hangi kodu yazacağını bilmek için neyi nerede kullanacağını bilmen gerekiyor. Keşke ilk haftalarda böyle aktif olabilseydik" (KGK-14) (Pasif kalma)

Deney grubunda ve kontrol grubunda yer alan katılımcıların ders süreci sonunda programlamaya karşı görüşlerini içeren beşinci görüşme sorusuna ait tema ve alt temalar Tablo 18' de verilmiştir

Tablo 18

Katılımcıların Ders Süresi Sonunda Programlamaya Karşı Görüşlerine İlişkin Tema ve Alt Temalar

Grup		**Deney**	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Olumlu değişime neden olan etmenler	Mantığını anlama Uygulama yapma Eğlenceli olması Basitleştirmesi Sonucunu görebilme Öğrenme isteğini artırması Birlikte yapma	Mantığını anlama
	Olumsuz değişime neden olan etmenler	Karamsar olma	Başarısız olma Uygulamama olmaması Karamsar olma Sıkıcı olması
II. Odak Grup Görüşmesi	Olumlu değişime neden olan etmenler	Geçmiş öğrenme deneyimleri Başaralı olma Öğrenme isteğimin artması	Başarılı olma
	Olumsuz değişime neden olan etmenler	Kod hataları Sıkıcı olması	Geçmiş öğrenme deneyimleri Başarısız olma Zor olması

Tablo 18'de görüldüğü gibi deney ve kontrol grubunda yer alan katılımcıların ders sonunda programlamaya ilişkin görüşlerinin "olumlu değişime neden olan etmenler" ve "olumsuz değişime neden olan etmenler" olmak üzere iki temada toplandığı görülmektedir. Birinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların ders sonunda programlamaya karşı görüşlerinin "olumlu değişime neden olan etmenler" teması altında; "mantığını anlama", "uygulama yapma", "eğlenceli olması", "basitleştirmesi", "sonucunu görebilme", "öğrenme isteğini artırması" ve “birlikte yapma" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "mantığını anlama" alt temasında toplandığı görülmektedir. "Olumsuz değişime neden olan etmenler" teması altında ise deney grubunda yer alan katılımcıların ders sonunda programlaya ilişkin görüşleri "karamsar olma" alt temasında, kontrol grubunda ise "başarısız olma",

"uygulamama olmaması", "karamsar olma" ve "sıkıcı olması" alt temasında toplandığı görülmektedir.

Katılımcıların ders süreci sonunda programlamaya karşı görüşlerinde olumlu yönde değişimine neden olan etmenlere ilişkin görüşleri incelenmiştir. Buna göre deney grubunda yer alan katılımcılar; Scratch ile oyun tasarımı sürecinin mantığını anlamaya yardımcı olması, uygulama yapabilme olanağı sağlaması, eğlenceli olması, programlamayı basitleştirmesi, sonucunu görebilme olanağı sağlaması, öğrenme isteğinin artırması ve birlikte yapma olanağı sunması açısından programlamaya karşı olan görüşlerini olumlu şekilde değiştirdiğini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme sürecinin mantığını anlamaya yardımcı olması açısından programlamaya karşı olan görüşlerini olumlu şekilde değiştirdiğini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların ders sonunda programlamaya karşı görüşlerinin olumlu yönde değişimine neden olan etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

> "*Aslında korkuyordum başta. Çünkü ben bu süreci akış diyagramı görerek geçirdim. Benim en büyük problemim, problemin ne istediğini anlamamak zorluyordu. Fakat burada oyun yapmak sürükleyici gibi geldi. Hem mantığını da daha iyi anladım. Ne istediğini anlayabiliyorum. Mantığını kavradım en azından. Korkuyordum şu an için ama şimdi iyi gibi"* (DGK-6) (Mantığını anlama)
>
> "*...dersten önce programlama deyince herhalde dedim çok zor bir ders. Tabi bazı kişilerde korkutuyor bizi böyle zor ders şöyle zor ders. En sonunda bir şeyler başarınca güzel oluyor. Anlamışım çünkü. Bir program yapabilmesi insanın iyi yani"* (KGK-12) (Mantığını anlama)
>
> "*Benimde aynı şekilde açıkçası üst sınıflar çok korkutuyordu yani onlardan dolayı bir önyargım vardı çok zor olduğunu söylüyorlardı genelde de hep alttan alıyorlarmış çok büyük bir önyargım var açıkçası derste uygulamalı olarak işleyince, Scratch gibi bir programda işleyince açıkçası benimde ön yargım kalktı"* (DGK-20) (Uygulama yapma)
>
> *"Ben zorlanmam ama sıkıcı geçer diye düşünüyordum. Ama beklediğim kadar zor olmadı. Kolaydı. Eğlenceliydi. Sıkıcı olmaması iyiydi. Lisede öyleydi. Akış diyagramları çizerdik. Nereye ne bağlardık anlamazdım. Çizerdik sadece*

dinlemezdim pek. Anladım ben şimdi anlatılanları. Dersin sonunda da oyun çıkarmak iyiydi." (DGK-8) (Eğlenceli olması)

"Ben lise de aynı 12 saat programlama yapardık ama derste genelde uyurduk. Hoca problemi verir giderdi. Ben lise bitince artık birşey olmaz herhalde dedim. Sonra burada ise dedim sıkıcıdır herhalde. Sonra oyun yaptık falan şaşırdım. Hiç beklediğim gibi olmadı. Artık ders de sıkılmıyorum. Hatta yurda gidince hep çözdüm." (DGK-15) (Eğlenceli olması)

"Dersten önce algoritmanın beni zorlayacağını düşünüyordum Scratch programı ile daha basit hale geldiğini düşünüyorum. Programlamaya karşı beklentim değişti" (DGK-2) (Basitleştirmesi)

"Bende ilk başta dediğim gibi programlama bana zor geliyor korkutuyor. Ama Scratch ile birşeyler yapmak dönüt almak bazı şeyleri aşmamı sağladı. Nerede yanlış yaptığımı görmek hep böyle olsa keşke. Lisede hatırlıyorum tek nokta yüzünden bir saat programın çalışmadığını bilirim." (DGK-5) (Sonucu görebilme)

"Programlamayı sevmezdim lisede de sevmezdim. Ama Scratch ile ne bileyim biraz daha yapabilmek geldi. Yani öğrenmeye de daha çok meyil aldım. Öğrenmeyi çok istedim daha fazlasını." (DGK-9) (Öğrenme isteğini artırması)

"*Çok sevdim sonra dersi. Eğlenceli geldi. Mesela yapamayınca birbirimize soruyorduk. Fikir alışverişindi bulunduk çoğu arkadaşımızla. Özgür bir ortam var gibiydi"* (DGK-26) (Birlikte yapma)

Katılımcıların ders süreci sonunda programlamaya karşı görüşlerinin olumsuz yönde değişimine neden olan etmenlere ilişkin görüşleri incelenmiştir. Buna göre deney grubunda yer alan katılımcılar programlaya karşı görüşlerinde karamsar olduklarını belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme sürecinde başarısız olmaları, uygulama olanaklarının olmaması, sıkıcı olması ve programlamaya karşı görüşlerinde karamsar olduklarından dolayı programlamaya karşı görüşlerinin olumsuz olduğunu belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların ders sonunda programlamaya karşı görüşlerinin olumsuz yönde değişimine neden olan etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

" ...ben programlamayı sevmiyordum. Şimdi biraz ısındım ama ben. Kafam yine de karışık. Zor geliyor halen " (DGK-12) (Karamsar olma)

"*Programlama zor, hep zor… Halen biraz korkuyorum. Bilmiyorum. Kafam çok karışık"* (KGK-25) (Karamsar olma)

"*Ben zaten bir ön yargı ile başladım. Ders başlayınca dedim belki yaparım, kolaymış. Sonra konular ilerledikçe yapamadım. Git gide zor konular geldi sonra. Tekrar başladı korkum. Yapamadım sınavda da"* (KGK-14) (Başarısız olma)

"Ben kod yapacağız bilgisayar kullanacağız sanıyordum. Umduğum gibi olmadı, yazdık sürekli" (KGK-16) (Uygulama olmaması)

"Benim için hep programlama sıkıcı olmuştur. Burada problemler yapmak iyiydi. Hem de temelden başlamak. Ama algoritma çok uzun sürdü. Bir müddet sonra yada aynı anda akış diyagramı çizmek yerine kod yazsak olurdu. Ben bazen sıkıldım" (KGK-24) (Sıkıcı olması)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların ders sonunda programlaya karşı görüşlerinin "olumlu değişime neden olan etmenler" teması altında; "geçmiş öğrenme deneyimleri", "başarabilme", "öğrenme isteklerinin artması" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "başarılı olma" alt temasında toplandığı görülmektedir. "Olumsuz değişime neden olan etmenler" teması altında ise deney grubunda yer alan katılımcıların ders sonunda programlamaya ilişkin görüşlerinin; "kod hataları" ve "sıkıcı olması" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "başarısız olma", "zor olması" ve "geçmiş öğrenme deneyimleri" alt temasında toplandığı görülmektedir.

Katılımcıların ders süreci sonunda programlamaya karşı görüşlerinin olumlu yönde değişimine neden olan etmenlere ilişkin görüşleri incelenmiştir. Buna göre deney grubunda yer alan katılımcılar; geçmiş öğrenme deneyimlerinin, başarılı olmalarının ve öğrenme isteklerinin artmasının programlamaya karşı olan görüşlerini olumlu şekilde değiştirdiğini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise başarılı olmalarının programlamaya karşı görüşlerini olumlu şekilde değiştirdiğini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların ders sonunda programlamaya karşı görüşlerinin olumlu yönde değişimine neden olan etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Ben ilk derse başlayacağım da sıkıcı olur diye düşünüyordum. Bir de zor olur diye düşünüyordum ama Scratch ile kolay ve eğlenceli oldu. Artık zaten iyice pratikleştim. Problemlerde hiç düşünmedim hemen mantığını kavradım kodlarını da yazabildim. Programlama zor değilmiş aslında" (DGK-8) (Geçmiş öğrenme deneyimleri)

"...ben zaten lisede sevmiyordum zaten programlama dersine karşı ön yargım vardı istemiyordum sonra baktım ki bir şeyler öğrendiğimin farkına vardım. C#' a geçince baktım ki aslında herşeyin algoritmasını yazabiliyorum önyargılarım aslında biraz geçti." (DGK-25) (Başarılı olma)

"Üst sınıflardan duyduklarımızdan dolayı çok korktum. Korkuyordum yapamam diyordum. Ama artık yapabiliyorum o kadarda zor değilmiş" (DGK-9) (Başarılı olma)

"Başta kaldım diyordum bu dersten. Sonra ısındım. Yapabildiğimin, yapabileceğimin farkına vardım" (KGK-2) (Başarılı olma)

"Benim için çok şey değişti tabiki. Çünkü sevmiyordum programlamayı. Ama bizim derslerimiz farklı geçti. Programlamayla ilgili olarak öğrenme isteğim arttı" (DGK-1) (Öğrenme isteğini artırma)

Katılımcıların ders süreci sonunda programlamaya karşı görüşlerinin olumsuz yönde değişimine neden olan etmenlere ilişkin görüşleri incelenmiştir. Buna göre deney grubunda yer alan katılımcılar kod hatalarının ve sürecin sıkıcı olmasının programlamaya karşı olan görüşlerini olumsuz etkilediğini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise geçmiş öğrenme deneyimlerinin, sürecin zor olmasının ve başarısız olmalarının programlamaya karşı olan görüşlerini olumsuz şekilde değiştirdiğini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların ders sonunda programlamaya karşı görüşlerini olumsuz yönde değişimine neden olan etmenlere ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Ne güzel oyun yapıyorduk iyiydi. Bu ikinci bölüm bana sıkıcı geldi. Yok kod yaz, virgülü unutma, parantezi unutma" (DGK-26) (Kod hataları)

"Siyah ekrana kadar her şey iyiydi. Sonra koptu. Sıktı beni" (DGK-2) (Sıkıcı olması)

"Benim için hep zordu programlama halen de zor olduğunu düşünüyorum. Çünkü başta anlamadım, sonra kodlar geldi iyice karıştı" (KGK-16) (Geçmiş öğrenme deneyimleri)

"Yapabilirsem severim diye düşünüyorum. Ama programlama yapamıyorum" (KGK-23) (Başarısız olma)

"Programlama zaten zor ve kafa karıştırıcı. Çok kafa yormak gerekiyor. Çok işlem gerektiriyor" (KGK-15) (Zor olması)

Deney grubunda ve kontrol grubunda yer alan katılımcıların gelecekte programlama ile ilgili bir işte çalışmaya ilişkin görüşlerini içeren altıncı görüşme sorusuna ait tema ve alt temalar Tablo 19'da verilmiştir.

Tablo 19

Katılımcıların Gelecekte Programlama İle İlgili Bir İşte Çalışmaya İlişkin Görüşlerine Ait Tema ve Alt Temalar

Grup		**Deney**	**Kontrol**
Görüşme	**Tema**	**Alt Temalar**	**Alt Temalar**
I. Odak Grup Görüşmesi	Tercih etme nedenleri	Mantığını kavradığını düşünme Öğretici olarak programlamayı seçme Zevkli olduğunu düşünme Kolay olduğunu düşünme	Mantığını kavradığını düşünme Hobi olarak seçme Maddi katkısı
	Tercih etmeme nedenleri	Bir karar vermek için erken olduğunu düşünme Zor olduğunu düşünme	Bir karar vermek için erken olduğunu düşünme Zor olduğunu düşünme Mantığını kavrayamadığını düşünme Sıkıcı olduğunu düşünme
II. Odak Grup Görüşmesi	Tercih etme nedenleri	Mantığını kavradığını düşünme Öğretici olarak programlamayı seçme Maddi katkısı Özgüven artışı	Mantığını kavradığını düşünme Hobi olarak seçme
	Tercih etmeme nedenleri	Bir karar vermek için erken olduğunu düşünme	Bir karar vermek için erken olduğunu düşünme Zor olduğunu düşünme Mantığını kavrayamadığını düşünme Özgüven eksikliği

Tablo 19'da görüldüğü üzere deney ve kontrol grubunda yer alan katılımcıların gelecekte programlama ile ilgili bir işte çalışmaya ilişkin görüşlerinin "tercih etme nedenleri" ve "tercih etmeme nedenleri" olmak üzere iki temada toplandığı görülmektedir. Birinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmaya ilişkin görüşlerinin "tercih etme nedenleri" teması altında; "mantığını kavradığını düşünme",

"öğretici olarak programlamayı seçme", "zevkli olduğunu düşünme" ve "kolay olduğunu düşünme" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "mantığını kavradığını düşünme", "hobi olarak seçme" ve "maddi katkısı" alt temalarında toplandığı görülmektedir. "Tercih etmeme nedenleri" teması altında ise deney grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmaya ilişkin görüşlerinin "bir karar vermek için erken olduğunu düşünme" ve "zor olduğunu düşünme" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "bir karar vermek için erken olduğunu düşünme", "zor olduğunu düşünme", mantığını kavrayamadığını düşünme" ve "sıkıcı olduğunu düşünme" alt temalarında toplandığı görülmektedir.

Katılımcıların ders süreci sonunda programlama ile ilgili bir işte çalışmayı tercih etmeyle ilgili görüşleri incelendiğinde deney grubunda yer alan katılımcılar; programlamanın mantığını kavradıklarını, programlamanın eğlenceli ve kolay olduğunu düşündükleri için programcılığı bir meslek olarak seçebileceklerini ve öğretmen olduklarında da programlama öğretmeyi tercih edeceklerini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise programlamanın mantığını kavradıklarını düşündükleri için ve maddi katkısından dolayı programcılığı bir meslek olarak seçebileceklerini ayrıca hobi olarak programlama yapabileceklerini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmayı tercih etme ile ilgili görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *"Ben de belki düşünebilirim. Aslında güzel iş. Ders sürecinin etkisi oldu mu dersek. Oldu tabi ki. Çünkü sevdim, mantığı oturtmada yardımcı oldu. Yapabiliyorum diyorum en azından. Ama ilerde ne olur bilmiyorum. Belki ders süreci farklı olsaydı soğuyabilirdim de"* (DGK-26) (Mantığını kavradığını düşünme)
>
> *" İleride belki yaparım. Önce programlama deyince nasıl yaparım bilmiyordum. Ama şimdi anladım"* (KGK-10) (Mantığını kavradığını düşünme)
>
> *"Ben öğretmen olmak istiyorum. Öğrendiğim birşeyi birilerine aktarmayı çok istiyorum. Mesela programlama öğretmek için klasik yöntem değil Scratch iyi kullanarak öğretmeği isterim hem görsel hem de daha etkili olabilir, bu yolla da öğretirim temel oturtmak kısmında kesinlikle faydalı olacağını düşünüyorum."* (DGK-9) (Öğretici olarak programlamayı seçme)

"Şimdi ben öğretmen olsam bende programlamayı böyle öğretirim. Çiziyorsunuz görüyorsun tasarlıyorsun oyun yapıyorsun, aşama aşama mantığını öğretirim. Böylelikle erken yaşta programlamayı adım adım problem çözmeyi öğrenirler" (DGK-24) (Öğretici olarak programlamayı seçme)

"Neden olmasın. Zaten önceden de biliyordum. Derslerin zevkli geçmesi beni ısındırdı programlamaya". (DGK-23) (Zevkli olduğunu düşünme)

"Derse başlamadan önce hep belkiler vardı. Ama gördüm ki zor birşey değil. Bu işin nasıl olacağını azçok anladım. Kendimi geliştirebileceğimin farkına vardım. Artık belki olabilirim programcı. Ya da programlama yaparım. O kafamda ki duraklamalar kalktı" (DGK-8) (Kolay olduğunu düşünme)

"...programlamayı meslek olarak değil de ancak hobi olarak yapabilirim Yani ben öğretmen olsam yanında hobi olarak yaparım, pek yanaşmam" (KGK-21) (Hobi olarak seçme)

"Mesela programlama ile ilgili çok güzel meslekler var. Eğer programlamayı tamamen çözebilsem tercih ederim. Öğretmenlik yerine. Maddi açıdan da iyi olur. Güncel bir alan" (KGK-4) (Maddi katkısı)

Katılımcıların ders süreci sonunda programlama ile ilgili bir işte çalışmayı tercih etmeme ile ilgili görüşleri incelendiğinde deney grubunda yer alan katılımcılar, programlamanın zor olduğunu ve bir karar vermek için erken olduğunu düşündükleri için programcılığı bir meslek olarak tercih etmeyeceklerini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise bir karar vermek için erken olduğunu düşündükleri, programlamanın zor olduğunu düşündükleri, programlamanın mantığını kavrayamadıklarını ve programlamanın sıkıcı olduğunu düşündükleri için programcılığı bir meslek olarak tercih etmeyeceklerini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmayı tercih etmeme ile ilgili görüşlerinin yer aldığı ifadeleri şu şekildedir:

"Benim için bu sorunun cevabı çok erken. Daha bir dil bilmiyorum. Şuan çok birşey bilmiyor gibiyim. Ama öğrenebilirim gibi geliyor. İlgim var ama." (DGK-22) (Bir karar vermek için erken olduğunu düşünme)

"Ben bilmiyorum...şu an tam yapabilir miyim yapamaz mıyım çelişki içindeyim. Kendime güvenim yok. Bir ön yargıyla başladım zaten. Öyle işte. Daha bir şey görmedim. Programcılık ile ilgili bilgim bilmiyorum. Düşünmedim pekte. Dersin

sonunda karar verebileceğim bir şey" (KGK-9) (Bir karar vermek için erken olduğunu düşünme)

*"Önceden de sevmiyordum. Şimdi biraz biraz ısındım ama yine de cesaretim yok. Dursun kenarda ama pek istemiyorum. Bana karmaşık ve zor geliyor. Çok kafa patlatmak lazım." (*DGK-25) (Zor olduğunu düşünme)

"Ben yapamam. Sıkıntılı bir süreç. Lisede hayalimdi ama. Gördüğüm kadarıyla zor gibi. Programlama sürecindeki başarıma bağlı" (KGK-24) (Zor olduğunu düşünme)

"Anlasam yaparım. Ama şuan mantığını çok anlamadım. Kod görmedim." (KGK-6) (Mantığını kavrayamadığını düşünme)

" Ben kesinlikle yapmam. Hiçbir şey anlamıyorum programlamadan. Sürekli problem çözmek. Sıkıntı orada. Sorun mantığı yapabilmekte. Ben onu anlayamıyorum henüz" (KGK-17) (Mantığını kavrayamadığını düşünme)

" Evet, bir şeyler yapmak belki mutlu olabilir insan ama güzel olabilir program yazmak bir de o işi yapmak bence sıkıcı gibi" (KGK-23) (Sıkıcı olduğunu düşünme)

İkinci odak grup görüşmesine ilişkin bulgular incelendiğinde deney grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmaya ilişkin görüşlerinin "tercih etme nedenleri" teması altında; "mantığını kavradığını düşünme", "öğretici olarak programlamayı seçme", "maddi katkısı" ve "özgüven artışı" alt temalarında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "mantığını kavradığını düşünme" ve "hobi olarak seçme" alt temalarında toplandığı görülmektedir. Tercih etmeme nedenleri" teması altında ise deney grubunda yer alan katılımcıların "bir karar vermek için erken olduğunu düşünme" alt temasında toplandığı, kontrol grubunda yer alan katılımcıların görüşlerinin ise "bir karar vermek için erken olduğunu düşünme", "zor olduğunu düşünme", "mantığını kavrayamadığını düşünme" ve "özgüven eksikliği" alt temalarında toplandığı görülmektedir.

Katılımcıların ders süreci sonunda programlama ile ilgili bir işte çalışmayı tercih etme ile ilgili görüşleri incelendiğinde deney grubunda yer alan katılımcılar; programlamanın mantığını kavradıklarını ve programlamayla ilgili özgüvenlerinin arttığını düşündükleri için ve maddi katkısından dolayı programcılığı bir meslek olarak seçebileceklerini ve öğretmen olduklarında da programlama öğretmeyi tercih

edeceklerini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise programlamanın mantığını kavradıklarını düşündükleri için programcılığı bir meslek olarak seçebileceklerini ayrıca hobi olarak programlama yapabileceklerini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmayı tercih etme ile ilgili görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *"Benim için ikinci planda programlama. Eğer öğretmenlik olmazsa programlamaya yönelirim diye düşünüyorum. Programlamaya başlamadan önce kafamda programcılık hiç yoktu ama bu dersten sonra aslında yapabileceğimi anladım"* (DGK-14) (Mantığını kavradığını düşünme)
>
> *"Ben programlamadan uzak dururum diyordum. Sadece dersten geçecek kadar öğrensem yeter diyordum. Ama yapabileceğimi anladım"* (DGK-16) (Mantığını kavradığını düşünme)
>
> *"Geleceği parlak bir meslek programcılık. Şu anki bilgim ne kadar yeter bilmiyorum ama ben ileride de öğreneceklerimle yapabilirim herhalde. Aslında birçok açıdan öğrendim. "* (KGK-4)(Mantığını kavradığını düşünme)
>
> *"Ben kesinlikle öğretmen olarak devam ederim ama programlama da anlatırım. Ders benim için çok verimli geçti. En azından nasıl programlama öğreteceğimi öğrendim".* (DGK-1) (Öğretici olarak programlamayı seçme)
>
> *"Programlamayı tamamen hayatımdan çıkartmam ama öğretmen olduğumda da programlamayı en iyi şekilde öğretmeyi amaçlıyorum. Mesela bende Scratch kullanarak bu düzende devam etmeyi düşünüyorum. Yani önce mantığını çok basitleştirerek öğretmeyi sonra da C# öğretmeyi".* (DGK-9) (Öğretici olarak programlamayı seçme)
>
> *"Programcılık iyi bir meslek zevkli bir meslek öğretmenlikten daha çok getirisi olur kodları yazarken bir şeyler yapmak beni heyecanlandırıyor"* (DGK-15) (Maddi katkısı)
>
> *"Ben zaten düşünüyorum. Dersten sonra da bu düşüncem daha da arttı. Bu konuda kendime olan güvenim arttı"* (DGK-3) (Özgüven artışı)
>
> *"Ben programlamayı yaparsam hobi olarak yaparım. Hayatımı kolaylaştırmak için kullanırım. O konuda programlamaya devam ederim."* (KGK-12) (Hobi olarak seçme)

Katılımcıların ders süreci sonunda programlama ile ilgili bir işte çalışmayı tercih etmeme ile ilgili görüşleri incelendiğinde deney grubunda yer alan katılımcılar; bir karar vermek için erken olduğunu düşündükleri için programcılığı bir meslek olarak tercih etmeyeceklerini belirtmişlerdir. Kontrol grubunda yer alan katılımcılar ise bir karar vermek için erken olduğunu düşündükleri, programlamanın zor olduğunu, programlamanın mantığını kavrayamadıklarını ve programlamayla ilgili özgüvenlerinin azaldığını düşündükleri için programcılığı bir meslek olarak tercih etmeyeceklerini belirtmişlerdir. Deney grubunda ve kontrol grubunda yer alan katılımcıların programlama ile ilgili bir işte çalışmayı tercih etmemek ile ilgili görüşlerinin yer aldığı ifadeleri şu şekildedir:

> *"Ben aslında düşündüm de programlama yapabilir miyim meslek olarak seçebilir miyim ömrüm boyunca sıkar mı diye düşünüyorum. Fakat önce öğretmen olmayı düşünüyorum. Yine de benim için daha erken bu kararı vermek için"* (DGK-26) (Bir karar vermek için erken olduğunu düşünme)
>
> *"Programlama hakkında bilgim olsun isterim ama bilemiyorum ilerde gidişatta ne olur. Pek emin değilim"* (KGK-11) (Bir karar vermek için erken olduğunu düşünme)
>
> *"Ben bitsin okul bir daha elimi sürmem. Yapmak istemiyorum. Çok zorluyor, çok yorucu"* (KGK-24) (Zor olduğunu düşünme)
>
> *"Programlama bir tutku yani onu o kadar çok seversen ancak öyle yaparsın hani ben şimdi programlama yaparım diyemem. Programlama belli bir altyapı belli bir birikim belli bir çalışma gerektirir."* (KGK-21) (Mantığını kavrayamadığını düşünme)
>
> *"Ben program seviyorum ama yapmam yani bazı şeyler mantığıma otursa da bazı şeyler mantığıma oturmuyor yani. Üzerine çok düşünmek yorucu. Bu konuda kendime güvenmiyorum yani yapamam herhalde"* (KGK-7) (Özgüven eksikliği)

Nitel bulgular incelendiğinde katılımcı görüşleri doğrultusunda deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı etkinliklerini eğlenceli ve kolay, kontrol grubunda yer alan katılımcıların ise akış diyagramları ile problem çözme sürecini zor ve sıkıcı olarak nitelendirdikleri görülmektedir. Bununla birlikte; deney grubunda yer alan

katılımcılar ders süresince aktif olmalarını, ürün oluşturma imkanı sağlamasını, süreci basitleştirmesini ve akılda kalıcılığı artırmasını Scratch ile oyun tasarımı etkinliklerinin üstünlükleri olarak belirtmişken, kontrol grubunda yer alan katılımcılar ise ders süresince pasif olmalarını, problemleri test etme imkanlarının olmamasını ve sonuçların farklı çıkmasını akış diyagramlarını ile problem çözme etkinliklerinin sınırlılıkları olarak belirtmişlerdir. Katılımcıları ders süresince güdüleyen yada zorlayan etmenlere ilişkin görüşleri incelendiğinde ise deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinin sağladığı üstünlüklerden dolayı programlamaya ilişkin motivasyonlarının artığını ve programlama mantığını kavramalarına yardımcı olduğunu bildirmişlerdir. Ayrıca Scratch ile oyun tasarımı etkinlikleri sürecinde programlama mantığını kavramalarının, C# programlama dilini öğrenmelerine katkı sağladığını vurgulamışlardır. Ancak kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme etkinliklerinin oluşturduğu sınırlılıklardan dolayı programlamaya ilişkin motivasyonlarının düştüğünü ve programlama mantığını kavramada sıkıntı yaşadıklarını belirtmişlerdir. Ayrıca akış diyagramları ile problem çözme etkinlikleri sürecinde programlama mantığını kavramakta sıkıntı yaşamalarının C# programlama dilini öğrenmelerine katkı sağlamada yetersiz kaldığını vurgulamışlardır. Diğer önemli bir bulgu ise; Scratch ile oyun tasarımı etkinlikleri geçmiş öğrenme deneyimi teması altında deney grubunda yer alan katılımcıların programlamaya karşı düşüncelerinde olumlu yönde değişime neden olurken, akış diyagramları ile problem çözme etkinlikleri ise yine geçmiş öğrenme teması altında kontrol grubunda yer alan katılımcıların programlamaya karşı düşüncelerinde olumsuz yönde değişme neden olmuştur. Buna göre nitel bulgular doğrultusunda; Scratch ile oyun tasarımı etkinliklerinin sağladığı üstünlüklerinden dolayı katılımcıların motivasyonunu artırdığı, programla mantığını kavramada yardımcı olduğu ve bundan dolayı katılımcılara C# programlama dilini öğrenmede kolaylık sağladığı söylenebilir. Bunun aksine akış diyagramları programlama öğretiminin oluşturduğu sınırlılıklardan dolayı katılımcıların motivasyonunu düşürdüğü, programla mantığını kavramada ve C# programlama dilini öğrenmede katılımcılara katkı sağlamada yetersiz kaldığı söylenebilir.

DÖRDÜNCÜ BÖLÜM
SONUÇ VE ÖNERİLER

Bu bölümde araştırmada elde edilen bulgular çerçevesinde ulaşılan sonuçlar ile bu sonuçlara dayalı olarak uygulamaya ve yapılacak araştırmalara yönelik önerilere yer verilmiştir.

Sonuçlar

Günümüzde yazılım geliştirebilen ve programlama becerisine sahip insan gücüne oldukça gereksinim duyulmaktadır. Ancak programlama eğitiminde yaşanan sorunlar dolayısıyla bilgisayar bilimleri alanında eğitim gören öğrencilerin sayısında önemli azalmalar görülmektedir (Heersink ve Moskal, 2010). Özellikle başlangıç düzeyindeki programcılar programlamaya karşı olumsuz tutum geliştirmekte, programlama dersinden başarısız olmakta ve birçoğu da dersi ya da ilgili bölümü bırakmaktadır (Kinnunen ve Malmi, 2008). Yaşanan bu olumsuzluğun azaltılması için başlangıç düzeyindeki programlama derslerinde temel programlama yapılarının ve programlama mantığının etkili şekilde öğretilmesi gerekmektedir. Programlama mantığının öğretiminde algoritma problemlerinden yararlanılmaktadır. Ancak birçok öğrenci karmaşık ve soyut olan bu algoritmik yapıyı kavramada ve problemi algoritmaya dönüştürmede sorun yaşamaktadır (Winslow, 1996). Bu nedenle temel programlama mantığı ve algoritma öğretimini daha somut ve anlaşılır hale getirmek için eğitimciler farklı görselleştiriciler kullanmaktadırlar (Baldwin ve Kuljis, 2000). Bu görselleştiricilerin bazıları yalnızca programlama sürecini görselleştirerek çıktı sağlarken, bazıları da oyun ve hikâye gibi çokluortam tasarımına olanak sağlamaktadır (Cooper ve diğerleri, 2006). Özelikle oyun tasarımı süreci; öğrenenleri tasarım sırasında daha etkin hale getirmekte, zevkli ve eğlenceli bir öğrenme ortamı oluşturmakta ve öğrenenlerin programlamayı daha etkili öğrenmesini sağlamaktadır (Ke, 2014; Li, 2012; Mishra ve Girod, 2006; Resnick, 2007). Tasarım odaklı görselleştiriciler ele alındığında Scratch algoritma ve programlama öğretiminde en çok kullanılan araçlardan biridir (Resnick ve diğerleri, 2009). Scratch basit arayüzü, sürükle bırak yapısı, hikâye ve oyun tasarımına olanak tanıması ve her yaş için uygun olması özelliği ile programlama öğretiminde yaşanan zorlukları ve olumsuzlukları ortadan kaldırmak için etkili bir araç olarak kullanılabilir. Yapılan çalışmalar incelendiğinde programlama öğretiminde

Scratch kullanımına ilişkin birçok çalışma yer almaktadır. Bu çalışmaların genellikle lise ve alt kademelerde yapıldığı, Scratch ile oyun tasarımını etkinliklerinin programlama başarısına etkisinin incelendiği, bazı çalışmalarda ise başarının yanında motivasyona etkisi de incelendiği görülmektedir (Gülmez, 2009). Bununla birlikte alan yazında Scratch ile oyun tasarımı etkinliklerinde elde edilen deneyim ve bilgilerin yeni bir programlama dili öğretiminde devam edip etmediği ya da transfer edilip edilmediği ile ilgili yeterli çalışma bulunmadığı görülmektedir (Ozoran ve diğerleri, 2012; Rivzi ve diğerleri, 2011; Wescott, 2008). Bu bağlamda bu araştırmada Scratch ile programlama öğretiminin öğrencilerin motivasyon ve programlama başarısına etkisini incelenmiştir. Araştırmada ulaşılan sonuçlar aşağıda yer almaktadır.

Başarı Değişkenine İlişkin Sonuçlar

- Çok değişkenli varyans analizi sonucuna göre katılımcıların ölçme araçlarından aldığı puanlar, grup değişkenine, ölçüm zamanına, ölçüm zamanı ve grup etkileşimine (ölçüm zamanı*grup) göre anlamlı bir farklılık göstermektedir.
- Bu doğrultuda her bir ölçme aracı için ayrı ayrı tek değişkenli varyans analizi yapılmıştır. Yapılan varyans analizi sonucuna göre katılımcıların başarı puanları grup değişkenine, ölçüm zamanına, ölçüm zamanı ve grup etkileşimine (ölçüm zamanı*grup) göre anlamlı bir farklılık göstermektedir. Ölçüm zamanı ve grup değişkeninin ortak etkisinin daha iyi incelenebilmesi için basit temel etki analizi yapılmıştır. Buna göre:
    - Kontrol grubunda yer alan katılımcıların başarı puanları ölçme zamanı açısından karşılaştırıldığında; öntest ile sontest arasında sontest lehine, sontest ile sontest 2 arasında sontest 2 lehine, öntest ile sontest 2 arasında sontest 2 lehine anlamlı farklılık bulunmuştur.
    - Kontrol grubunda yer alan katılımcıların başarı puanlarının öntestten sontest 2 ölçümüne doğru anlamlı olarak arttığı görülmüştür.
    - Deney grubunda yer alan katılımcıların başarı puanları ölçme zamanı açısından karşılaştırıldığında; öntest ile sontest arasında sontest lehine, sontest ile sontest 2 arasında sontest 2 lehine, öntest ile sontest 2 arasında sontest 2 lehine anlamlı farklılık bulunmuştur.

- Deney grubunda yer alan katılımcıların başarı puanlarının öntestten sontest 2 ölçümüne doğru anlamlı olarak arttığı görülmüştür.
- Katılımcıların öntest, sontest ve sontest 2'den aldıkları başarı puanları grup değişkenine göre karşılaştırıldığında; öntest puanları açısından farklılık bulunmazken, sontest puanları açısından ve sontest 2 puanları açısından deney grubu lehine anlamlı farklılık bulunmuştur.
- Deney grubunun sontest ve sontest 2 puanları kontrol grubundan anlamlı olarak yüksektir.

Motivasyon Değişkenine İlişkin Sonuçlar

- Yapılan tek değişkenli varyans analizi sonucuna göre katılımcıların motivasyon puanları, ölçüm zamanına göre anlamlı bir farklılık göstermezken, grup değişkenine, ölçüm zamanı ve grup etkileşimine (ölçüm zamanı*grup) göre anlamlı bir farklılık göstermektedir. Ölçüm zamanı ve grup değişkeninin ortak etkisinin daha iyi incelenebilmesi için basit temel etki analizi yapılmıştır. Buna göre:
  - Kontrol grubunda yer alan katılımcıların motivasyon puanları ölçme zamanı açısından karşılaştırıldığında; öntest ile sontest arasında, sontest ile sontest 2 arasında, öntest ile sontest 2 arasında anlamlı farklılık bulunamamıştır.
  - Ortalama puanlar incelendiğinde kontrol grubunda katılımcıların motivasyon puanlarının birinci ölçümden ikinci ölçüme doğru azaldığı, ikinci ölçümden üçüncü ölçüme doğru arttığı görülmüştür.
  - Üçüncü ölçüm sonunda katılımcıların motivasyon puanlarının birinci ölçüme göre azalmış olduğu görülmüştür.
  - Deney grubunda yer alan katılımcıların motivasyon puanları ölçme zamanı açısından karşılaştırıldığında; öntest ile sontest arasında, sontest ile sontest 2 arasında, öntest ile sontest 2 arasında anlamlı farklılık bulunamamıştır.
  - Ortalama puanlar incelendiğinde deney grubunda katılımcıların motivasyon puanlarının birinci ölçümden ikinci ölçüme doğru ve ikinci ölçümden üçüncü ölçüme doğru arttığı görülmüştür.

- Üçüncü ölçüm sonunda katılımcıların motivasyon puanlarının birinci ölçüme göre arttığı görülmüştür.
- Katılımcıların öntest, sontest ve sontest 2'den aldıkları motivasyon puanları grup değişkenine göre karşılaştırıldığında; öntest puanları açısından farklılık yok iken, sontest puanları ve sontest 2 puanları açısından deney grubu lehine anlamlı farklılık bulunmuştur.
- Deney grubunun sontest ve sontest 2 puanları kontrol grubundan anlamlı olarak yüksektir.

Odak Grup Görüşmelerinden Elde Edilen Bulgular

Katılımcıların aldıkları programlama eğitimleri hakkındaki görüşleri her bir görüşme sorusu bağlamında incelenmiştir. Katılımcıların,

- Genel olarak ders hakkındaki görüşleri birinci (Scratch ile oyun tasarımı ve akış diyagramı ile problem çözme uygulamalarından sonra) ve ikinci odak grup görüşmesi (C# programlama öğretiminden sonra) için ayrı ayrı olmak üzere şu şekildedir:
  - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecine yönelik olumlu görüşlerinin; eğlenceli, kolay ve renkli alt temalarında, olumsuz görüşlerinin ise çok basit alt temasında toplandığı görülmüştür.
  - Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramı ile problem çözme sürecine yönelik olumlu görüşlerinin; kolay alt temasında, olumsuz görüşlerinin ise karmaşık, zor ve sıkıcı alt temalarında toplandığı görülmüştür.
  - İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretimine yönelik olumlu görüşlerinin; üst düzey, kolay ve eğlenceli alt temalarında, olumsuz görüşlerinin ise karmaşık, zor ve sıkıcı alt temasında toplandığı görülmüştür.
  - İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretimine yönelik olumlu görüşlerinin; kolay ve eğlenceli alt temalarında, olumsuz görüşlerinin ise karmaşık ve zor temasında toplandığı görülmüştür.

- Öğrencilerin ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşleri birinci ve ikinci odak grup görüşmesi için ayrı ayrı olmak üzere şu şekildedir:
    - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı etkinliklerinin sağladığı üstünlüklere ilişkin görüşlerinin; programlama mantığını kazandırması, akılda kalıcılığı artırması, motivasyonu artırması, yaparak öğrenme olanağı sağlaması, ürün oluşturma olanağı sağlaması, öğrenmeyi kolaylaştırması, araştırarak öğrenme olanağı sağlaması ve yaratıcılığı artırması alt temalarında; oluşturduğu sınırlılıklara ilişkin görüşlerinin ise temel yapılar için yetersiz kalması alt temasında toplandığı görülmüştür.
    - Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramı ile problem çözme etkinliklerinin sağladığı üstünlüklere ilişkin görüşlerinin; programlama mantığını kazandırması, akılda kalıcılığı artırması alt temalarında; oluşturduğu sınırlılıklara ilişkin görüşlerinin ise temel yapılar için yetersiz kalması ve öğrenenlerin pasif olması alt temalarında toplandığı görülmektedir.
    - İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretiminin sağladığı üstünlüklere ilişkin görüşlerinin; programlama yapabilme ortamı sağlaması alt temasında; oluşturduğu sınırlılıklara ilişkin görüşlerinin ise yalnızca kod yazmak zorunda kalmak alt temasında toplandığı görülmüştür.
    - İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretiminin sağladığı üstünlüklere ilişkin görüşlerinin; uygulama yapma olanağı sağlaması ve öğreneni aktif kılması alt temalarında; oluşturduğu sınırlılıklara ilişkin görüşlerinin ise yalnızca kod yazmak zorunda kalma alt temasında toplandığı görülmüştür.
- Öğrencileri ders süresince güdüleyen veya zorlayan durumlara ilişkin görüşleri birinci ve ikinci odak grup görüşmesi için ayrı ayrı olmak üzere şu şekildedir:
    - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecinde motivasyonlarını

artıran etmenlere ilişkin görüşlerinin eğlenceli ve zevkli olması, görsel olması, test etme ve dönüt olanağı sağlaması, uygulamalı olması, bir ürün ortaya koyma, kod gerektirmemesi, basitleştirmesi, zihinde canlandırması, birlikte yapma ve akran iletişimi sağlaması, dersin devamlılığını sağlaması, merakı artırması ve canlandırma olanağı sağlaması alt temalarında; motivasyonu düşüren etmenlere ilişkin görüşlerinin ise bazı temel yapıların mantığını anlayamama, temel yapıların iç içe girmesi ve tasarımın sınırlı kalması alt temalarında toplandığı görülmüştür.

- Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramları ile problem çözme sürecinde motivasyonlarını artıran etmenlere ilişkin görüşlerinin görsel olması, basitten zora doğru gitmesi ve zihinde canlandırması alt temalarında; motivasyonu düşüren etmenlere ilişkin görüşlerinin ise bazı temel yapıların mantığını anlayamama, temel yapıların iç içe girmesi, test etme ve dönüt olanağı olmaması, süreç içinde aktif olamama, uygulamanın olmaması, temel yapıların nerede kullanacağını bilememe, farklı sonuçların çıkması, adım adım mantığını karıştırma ve sürecin uzun olması alt temalarında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretimi sürecinde motivasyonlarını artıran etmenlere ilişkin görüşlerinin ilk başta mantığını anlamış olmak ve program yapma alt temalarında; motivasyonlarını düşüren etmenlere ilişkin görüşlerinin ise konsol ekranında çalışma ve kod hataları ile uğraşma alt temalarında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretimi sürecinde motivasyonlarını artıran etmenlere ilişkin görüşlerinin; süreç içinde aktif olma ve test etme ve dönüt olanağı alt temalarında; motivasyonlarını düşüren etmenlere ilişkin görüşlerinin ise kod hataları ile uğraşmak, bazı temel kavramların mantığını anlayamama, problemin analizinde sıkıntı yaşama ve kavramların iç içe girmesi alt temalarında toplandığı görülmüştür.

- Öğrencilerin ders süresince öğrendiklerini kullanma durumlarına ilişkin görüşleri birinci ve ikinci odak grup görüşmesi için ayrı ayrı olmak üzere şu şekildedir:
    - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecinde öğrendikleri bilgileri yeni bir programlama diline transfer etmeye ilişkin görüşlerinin; temel mantığı sürdürme ve test etme için kullanma alt temalarında; başka derse transfer etmeye ilişkin görüşlerinin ise eğitim materyali olarak kullanma alt temasında toplandığı görülmüştür.
    - Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramı ile problem çözme sürecinde öğrendikleri bilgileri yeni bir programlama diline transfer etmeye ilişkin görüşlerinin; temel mantığı sürdürme alt temasında toplandığı görülmüştür.
    - İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretimi sürecine geçmiş öğrenme deneyimlerini transfer etmeye ilişkin görüşlerinin; mantığı kavramaya yardımcı olması, eğlenceli olması, uygulamalı olması, kolaylaştırması, zihninde canlandırması, mantıksal hatayı azaltması ve ön yargıları yıkması alt temalarında toplandığı görülmüştür.
    - İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretimi sürecine geçmiş öğrenme deneyimlerini transfer etmeye ilişkin görüşlerinin; mantığı kavramaya yardımcı olması ve kolaylaştırması alt temalarında toplandığı; geçmiş deneyimi transfer etmemeye ilişkin görüşlerinin ise mantığı kavrayamama, sonucu görememe ve pasif kalma alt temalarında toplandığı görülmüştür.
- Öğrencilerin ders süreci sonunda programlamaya karşı düşünceleri birinci ve ikinci odak grup görüşmesi için ayrı ayrı olmak üzere şu şekildedir:
    - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı süreci sonunda programlamaya ilişkin düşüncelerinde olumlu yönde değişime neden

etmenlere ilişkin görüşlerinin; mantığını anlama, uygulama yapma, eğlenceli olması, basitleştirmesi, sonucu görebilme, öğrenme isteğini artırması ve birlikte yapma alt temalarında, olumsuz değişime neden olan etmenlere ilişkin görüşlerinin ise karamsar olma alt temasında toplandığı görülmüştür.

- Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramları ile problem çözme süreci sonunda programlamaya ilişkin düşüncelerinde olumlu yönde değişime neden etmenlere ilişkin görüşlerinin; mantığını anlama alt temasında, olumsuz değişime neden olan etmenlere ilişkin görüşlerinin ise başarısız olma, uygulamama olmaması, karamsar olma ve sıkıcı olma alt temalarında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretimi süreci sonunda programlamaya ilişkin düşüncelerinde olumlu yönde değişime neden etmenlere ilişkin görüşlerinin; geçmiş öğrenme deneyimleri, başarılı olma ve öğrenme isteğinin artması alt temalarında, olumsuz değişime neden olan etmenlere ilişkin görüşlerinin ise kod hataları ve sıkıcı olması alt temalarında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretimi süreci sonunda programlamaya ilişkin düşüncelerinde olumlu yönde değişime neden etmenlere ilişkin görüşlerinin; başarılı olma alt temasında, olumsuz değişime neden olan etmenlere ilişkin görüşlerinin ise geçmiş öğrenme deneyimleri, başarısız olma ve zor olma alt temalarında toplandığı görülmüştür.

- Öğrencilerin programlama ile ilgili bir işte çalışmaya ilişkin düşünceleri birinci ve ikinci odak grup görüşmesi için ayrı ayrı olmak üzere şu şekildedir:
  - Birinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı süreci sonunda programcılığı tercih etmeye ilişkin görüşlerinin; mantığını kavradığını düşünme, öğretici olarak programlamayı seçme, zevkli olduğunu

düşünme ve kolay olduğunu düşünme alt temalarında; programcılığı tercih etmemeye ilişkin görüşlerinin ise karar vermek için erken olduğunu düşünme ve zor olduğunu düşünme alt temalarında toplandığı görülmüştür.

- Birinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların akış diyagramları ile problem çözme süreci sonunda programcılığı tercih etmeye ilişkin görüşlerinin; mantığını kavradığını düşünme, hobi olarak seçme ve maddi katkısı alt temalarında; programcılığı tercih etmemeye ilişkin görüşlerinin ise bir karar vermek için erken olduğunu düşünme, zor olduğunu düşünme, mantığını kavrayamadığını düşünme ve zor olma alt temalarında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde deney grubunda yer alan katılımcıların C# programlama öğretimi süreci sonunda programcılığı tercih etmeye ilişkin görüşlerinin; mantığını kavradığını düşünme, öğretici olarak programlamayı seçme, maddi katkısı ve özgüven artışı alt temalarında; programcılığı tercih etmemeye ilişkin görüşlerinin ise bir karar vermek için erken olduğunu düşünme alt temasında toplandığı görülmüştür.
- İkinci odak grup görüşmesi bulguları incelediğinde kontrol grubunda yer alan katılımcıların C# programlama öğretimi süreci sonunda programcılığı tercih etmeye ilişkin görüşlerinin; mantığını kavradığını düşünme ve hobi olarak seçme alt temalarında; programcılığı tercih etmemeye ilişkin görüşlerinin ise bir karar vermek için erken olduğunu düşünme, zor olduğunu düşünme, mantığını kavrayamadığını düşünme ve özgüven eksikliği alt temalarında toplandığı görülmüştür.

Tartışma

Araştırma sonuçları incelendiğinde katılımcıların programlama başarılarının hem deney grubunda hem de kontrol grubunda artış gösterdiği ancak bu farkın deney grubu lehine anlamlı olduğu görülmüştür. Bu bulgu deney grubunda uygulanan Scratch ile oyun tasarımı etkinliklerinin, kontrol grubunda uygulanan akış diyagramları ile problem çözme etkinliklerine göre katılımcıların programlama dersine ilişkin

başarılarını artırmada daha etkili olduğunu göstermektedir. Bunun yanı sıra üçüncü ölçüm olan sontest 2 sonuçlarında da farkın deney grubu lehine devam ettiği görülmüştür. Buna göre deney grubunda uygulanan Scratch ile oyun tasarımı etkinliği, C# programlama öğretiminin yapıldığı haftalarda başarıyı kontrol grubuna göre daha çok artırmıştır . Ayrıca sontest 2 ölçümü sonucunda da başarı açısından farkın deney grubu lehine devam etmesi; katılımcıların Scratch ile oyun tasarımı sürecinde edindikleri becerileri, C# ile programlama öğrenme sürecine transfer ettikleri şeklinde yorumlanabilir. Bu bulgu Rizvi ve arkadaşları (2011) tarafından üniversite birinci sınıf öğrencileri ile yaptıkları çalışma sonunda elde edilen; programlama için temel ders olan CS0 dersinde Scratch ile oyun tasarımı etkinliklerinin, programlama eğitiminin yapıldığı CS1 dersindeki programlama başarısına etkili olduğuna ilişkin bulgu ile paralellik göstermektedir. Bununla birlikte Wescott (2008)'in çalışmasında elde edilen; C++ eğitimi ile paralel olarak yapılan Scratch ile oyun tasarımı etkinliklerinin, programlama başarısına etkisi olduğuna ilişkin bulgu ve Çağıltay (2007)'nin yaptığı çalışmada elde edilen; Scratch ile oyun tasarımı yapan öğrenci grubunun yapmayan gruba göre programlama başarılarının daha yüksek olduğuna ilişkin bulgular araştırma bulgularını destekler niteliktedir. Ancak alanyazında yer alan bu çalışmalarda; deney grubunda yer alan katılımcılara programlama dili öğretiminden önce ya da programlama dili öğretimi ile birlikte programlama mantığı öğretimine ilişkin bir uygulama söz konusu iken, kontrol grubunda yer alan katılımcılara programlama mantığı öğretimi söz konusu değildir. Bu durum iç geçerliliği etkileyen bir sınırlılık olarak ele alınabilir. Ancak bu araştırmada yapılan benzer çalışmalardan (Çağıltay, 2007; Rivzi ve diğerleri, 2011; Wescott, 2008) farklı olarak; C# programlama dili öğretiminden önce hem deney hem de kontrol grubunda farklı programlama mantığı öğretimi uygulanmış ve böylelikle iç geçerliliği tehdit eden bu durum ortadan kaldırılmaya çalışılmıştır. Ayrıca oyun tasarımı açısından ele alındığında araştırma bulguları; Alice, Flip gibi görselleştiriciler kullanılarak yapılan oyun tasarımı etkinliklerinin programlama başarısını artırdığına ilişkin çalışmalar ile de paralellik göstermektedir (Bishorp-Clark ve diğerleri, 2007; Cooper ve diğerleri, 2003; Howland ve Good, 2015). Araştırma sonunda yapılan odak grup görüşmelerinden elde edilen öğrenci görüşleri de Scratch ile oyun tasarımı sürecinin katılımcıların programlama başarısını artırdığına ilişkin bulguları ve programlama mantığının C# ile programlama sürecine transfer edildiğine ilişkin

bulguları desteklemektedir. Deney grubunda yer alan katılımcılar görüşlerinde Scratch ile oyun tasarımı sürecinde edindikleri bilgileri C# öğretimi sürecinde kullandıklarını ve Scratch'in sağladığı üstünlüklerle programlama mantığını ve programlamayı daha iyi öğrendiklerini belirtmişlerdir. Bununla ilgili olarak deney grubunda yer alan DGK-7 kodlu katılımcı; "*Kod yazarken fark ettim hızlanmışım. Bence bu Scratch sayesinde oldu. Çünkü sınavda da öyle oldu kodları yazmadan önce algoritmasını yazdım. Aklıma hep Scratch teki bloklar geliyordu, gözümün önüne geliyordu. Onları kodlara çevirdim işte"* şeklinde, DGK-24 kodlu katılımcı;" *Kolay gelmesinin sebebi bence Scratch ile mantığını öğrenmiş olmamız. Zaten değişkenleri falan Scratch te oturttuk. Buraya gelince de zaten doğrudan kodları yazdık. Zaten siz de bir daha döngü nedir anlatmadınız"* şeklinde ve DGK-2 kodlu katılımcı *"Algoritma süreci biraz karışık gelirdi. Scratch yardımıyla algoritmayı daha kolay algılamaya başladım"* şeklinde görüş bildirmişlerdir. Buradan da anlaşılacağı üzere; Scratch ile oyun tasarımı etkinliklerinin katılımcıların programlama mantığını anlamalarında yardımcı olduğu ve bununda C# programlama dilini öğrenmelerine katkı sağladığı görülmektedir.

Araştırma sonunda elde edilen diğer bir sonuç ise katılımcıların motivasyonlarının hem sontest hem de sontest 2 ölçümünde deney grubu lehine farklılık göstermesidir. Bir başka deyişle deney grubunda uygulanan Scratch ile oyun tasarımı etkinliklerinin, kontrol grubunda uygulanan akış diyagramları ile problem çözme etkinliklerine göre katılımcıların motivasyonlarını artırmada daha etkili olduğu ortaya çıkmıştır. Bununla birlikte deney grubunda uygulanan Scratch ile oyun tasarımı etkinliklerinin oluşturduğu motivasyon artışı etkisinin C# ile programlama öğretimi sürecinde de devam ettiği görülmektedir. Diğer bir önemli sonuç ise deney grubunda uygulama sonunda katılımcıların motivasyonunun artmış olması, kontrol grubunda azalmış olmasıdır. Araştırmada elde edilen bu bulgu alanyazında yer alan çalışmalar ile paralellik göstermektedir (Nikou ve Economides, 2014; Osman ve diğerleri, 2012). Ancak bu araştırmada diğer çalışmalardan farklı olarak; özellikle programlama mantığı öğretiminde birçok eğitimcinin tercih ettiği akış diyagramları ile problem çözme etkinliklerinin katılımcıların motivasyonunu düşürdüğü sonucuna ulaşılmıştır. Araştırma sonunda yapılan odak grup görüşmelerinden elde edilen öğrenci görüşleri de motivasyona ilişkin bulguları desteklemektedir. Deney grubunda yer alan katılımcılar görüşlerinde Scratch ile oyun tasarımı etkinliklerinin; eğlenceli, uygulamalı ve görsel

olması gibi sağladığı üstünlükler ile motivasyonlarının arttırdığını belirtmişlerdir. Bununla ilgili olarak deney grubunda yer alan DGK-4 kodlu katılımcı; "*Tabiki çok zevkliydi. Yani artık oyunlaştırma işine girmiştim eğlenceli olması için. Mesela siz ara veriyordunuz biz çıkmıyorduk. Oyunu yapmak için uğraşıyorduk."* şeklinde ve DGK-13 kodlu katılımcı; *"Aklımdan geçenleri tasarlamaya çalıştım. Ses ekledim. Bunlardan bir bütün oluşturdum." ve* yine aynı katılımcı *"bir ürün ortaya koymak mutlu etti hepimizi…"*şeklinde görüş bildirmişlerdir. Kontrol grubunda yer alan katılımcılar ise görüşlerinde akış diyagramı ile problem çözme etkinliklerinin; sıkıcı olma, süreç içinde pasif olma ve farklı sonuçların çıkması gibi oluşturduğu sınırlılıklar ile motivasyonlarını düşürdüğünü belirtmişlerdir. Bununla ilgili olarak kontrol grubunda yer alan KGK21 kodlu katılımcı "*Biz sadece tahtadakini yazıyoruz ya da çiziyoruz. Etkin olsak biraz daha iyi olurdu"* şeklinde ve KGK-24 kodlu katılımcı *"Tahtadakileri sürekli yazmak, çizmek sıkıcıydı"* şeklinde görüş bildirmişlerdir. Buradan da anlaşılacağı üzere; Scratch ile oyun tasarımı etkinliklerinin katılımcıların motivasyonunu artırdığı ve bu etkinin C# öğretimi sürecinde de devam ettiği, bunun aksine akış diyagramları ile problem çözme etkinliklerinin katılımcıların motivasyonunu düşürdüğü görülmüştür.

Tüm bu bulgular incelendiğinde Scratch ile oyun tasarımı etkinliklerinin programlama mantığını kazandırmada akış diyagramları ile problem çözme etkinliklerine göre daha etkili olduğu, öğrenenlerin programlamaya ilişkin motivasyonunu artırdığı ve bu artışın C# programlama dili öğretimi sürecinde de devam ettiği söylenebilir. Ayrıca programlama mantığının öğretildiği süreçte motivasyonun yüksek olması, öğrenenlerin programlama mantığını kavramalarına yardımcı olmuş olabilir ve programlama dili öğretimi sürecinde de öğrenenlerin motivasyon ve programlama başarısını artıran bir etmen olmuş olabilir. Bunun nedeni ise Scratch'in oyun tasarlama olanağı sunarak programlama mantığının öğretildiği algoritma sürecini daha anlaşılır ve eğlenceli hale getirmesi olabilir. Ayrıca oyun tasarımı sırasında öğrenenlerin etkin hale gelmesi, ürünle birlikte etkileşimde bulunması onların daha çok öğrenmeye çalışmalarına neden olmuş olabilir. Papert (1980) bu durumu; en iyi öğrenme tasarım sırasında ve öğrenenler kendi oyunlarını inşa ederken gerçekleşmektedir şeklinde açıklamaktadır. Özellikle öğrenenler oyunlarını inşa ederken ya da tasarlarken kendi öğrenme süreçlerini kontrol edebilmekte ve teoride yer alan bilgileri yaparak ve uygulayarak daha iyi öğrenebilmektedir (Mishra ve Girod,

2006). Ayrıca bir görselleştirici olan Scratch soyut olan algoritmik yapıyı daha somut ve anlaşılır hale getirerek katılımcıların programlama mantığını daha iyi öğrenmelerine olanak tanımış olabilir. Bununla birlikte; Scratch'in kolay kullanılabilir bir araç olması, oyun tasarım sürecinin ilgi çekici ve eğlenceli yapısı, zor ve sıkıcı olarak görülen programlama dersini daha zevkli ve eğlenceli hale getirerek öğrenenlerin motivasyonlarını arttırmış, dolayısıyla katılımcıların programlamayı daha iyi öğrenmelerini sağlamış olabilir (Gee, 2005; Moreno, 2012; Rajaravivarma, 2005). Bunun aksine mevcut ders programında yer alan ve düz anlatım yönteminin kullanıldığı akış diyagramları ile problem çözme etkinliklerinin ise öğrenenlerin programlamaya ilişkin motivasyonlarını düşürdüğü ve bu etkinin C# programlama dili öğretimi sürecinde de durağan şekilde devam ettiği söylenebilir. Dolayısıyla motivasyonun düşük olması katılımcıların programlama mantığını kavramalarını güçleştirmiş ve programlama başarılarını artırmada daha az etkili olmuş olabilir. Ayrıca akış diyagramları ile problem çözme etkinliklerinin programlama mantığı öğrenme sürecini somutlaştırmada yetersiz kalması, öğrenenlerin dersten sıkılmalarına ve motivasyonlarının düşmesine, dolayısıyla programlama mantığını kavramalarında sıkıntı yaşamalarına neden olmuş olabilir. Programlama mantığını kavramada yaşanan sıkıntı da öğrenenlerin C# programlama dillerine geçince başarılarını artırmada etkisiz kalmış olabilir. Buna göre Scratch ile oyun tasarımı etkinliklerinin öğrenenlerin programlamaya ilişkin motivasyonlarını artırdığı, böylelikle programlama mantığını kavramlarını kolaylaştırdığı ve sonuç olarak da programlama başarılarını artırmada etkili olduğu, ancak akış diyagramları ile problem çözme etkinliklerinin öğrenenlerin programlamaya ilişkin motivasyonlarını düşürdüğü ve programlama mantığını kavramada sıkıntı oluşturduğu söylenebilir.

Araştırmanın nitel bulguları incelendiğinde deney grubunda yer alan katılımcıların Scratch ile programlama öğretimine ilişkin görüşlerinin genellikle olumlu olduğu, dersi eğlenceli ve kolay olarak gördükleri, C# programlama öğretimini daha üst düzey bir ders olarak görmelerinin yanı sıra Scratch ile programlama öğretimine göre daha zor ve sıkıcı olarak gördükleri söylenebilir. Katılımcıların Scratch ile oyun tasarımı etkinliklerinin eğlenceli olmasına ilişkin görüşleri; Alimisi ve Winters (2010), Chui (2014), Malan ve Leitner (2007), Maloney ve arkadaşları (2008), Ozoron ve arkadaşları (2012), Salant ve arkadaşları (2013) tarafından yapılan çalışmalarda söz

edilen Scratch'in eğlenceli olmasıyla ilgili görüşler ile benzerlik göstermektedir. Ayrıca katılımcıların Scratch ile oyun tasarımı etkinliklerinin kolay olmasına ilişkin görüşleri; Koorsse ve arkadaşları (2015), Malan ve Leitner (2007), Wyfffles ve arkadaşları (2014) tarafından yapılan çalışmalarda ortaya çıkan Scratch'in kolay olmasıyla ilgili görüşler ile benzerlik göstermektedir. Buna göre deney grubunda yer alan katılımcılara göre Scratch ile oyun tasarımı etkinliklerinin eğlenceli ve kolay olduğu söylenebilir. Bunun nedeni Scratch'in kolay kullanılabilir bir araç olması ve oyun tasarım sürecinin eğlenceli ve ilgi çekici olması olabilir. Bununla birlikte deney grubunda yer alan katılımcılara göre C# programlama dili öğretimimin Scratch ile oyun tasarım sürecine göre daha zor ve karmaşık bir süreç olduğu ve daha üst düzey bir ders olduğu söylenebilir. Bunun nedeni C# programlama dilinde kod kullanımının Scratch'te blok yapı kullanımına göre daha zor olması ve programlama yapabilme bağlamında Scratch'ten daha üst düzey bir program olması olabilir. Kontrol grubunda yer alan katılımcıların akış diyagramıyla programlama öğretimine ilişkin görüşlerinin genellikle olumsuz olduğu, dersi karmaşık, zor ve sıkıcı olarak gördükleri, C# programlama öğretimini ise akış diyagramıyla programlama öğretimine göre daha eğlenceli gördükleri söylenebilir. Ortaya çıkan bu duruma göre deney grubunda yer alan katılımcıların aldıkları programlama eğitimine ilişkin genellikle olumlu görüş bildirdiği, kontrol grubunda yer alan katılımcıların ise olumsuz görüş bildirdiği söylenebilir.

Katılımcıların ders süresince gerçekleştirilen öğretme-öğrenme etkinliklerine ilişkin görüşleri incelendiğinde; Scratch ile oyun tasarımı etkinliklerinin deney grubunda yer alan katılımcıların, programlama mantığını kazanmalarına yardımcı olduğu, öğrenme süreçlerini kolaylaştırdığı, yaparak öğrenme ve ürün oluşturma olanağı sağladığı, motivasyonu ve yaratıcılıklarını artırma açısından üstünlük sağladığı görülmektedir. Bu bulgular Briggs (2013) ve Çağıltay (2007) tarafından yapılan çalışmalarda ortaya çıkan araştırma yapmaya teşvik etme, Salant ve arkadaşları (2013) tarafında yapılan çalışmada ulaşılan öğrenmeyi teşvik etme, Ozoron ve arkadaşları (2012) tarafından yapılan çalışmada elde edilen yaratıcılığı artırma görüşleri ile benzerlik göstermektedir. Ayrıca deney grubunda yer alan katılımcıların Scratch'in bazı temel yapılar için yetersiz kalmasını bir sınırlılık olarak dile getirmişlerdir. Bu bulgu Genç ve Karakuş (2011), Salant ve arkadaşları (2013) tarafından yapılan çalışmalarda ortaya çıkan programlama için yetersiz olması görüşleri ile benzerlik göstermektedir.

Kontrol grubunda yer alan katılımcıların görüşlerine göre ise akış diyagramıyla programlama öğretimi etkinliklerinin programlama mantığını kazandırma açısından üstünlük sağladığı, ancak öğrenenlerin süreç içinde pasif olmalarının katılımcılar açısından sınırlılık oluşturduğu görülmüştür. Katılımcı görüşleri doğrultusunda; deney grubunda uygulanan Scratch ile oyun tasarımı etkinliklerinin, akış diyagramlarına göre katılımcılar açısından daha çok üstünlük sağladığı söylenebilir. Ayrıca katılımcı görüşlerine göre; C# programlama öğretimi sürecinde kod yazmanın deney grubunda yer alan katılımcılar açısından programlama yapma olanağı sağladığı, ancak sürekli kod yazmanın oyun tasarımına göre sıkıcı olduğu sonucuna ulaşılmıştır. Kontrol grubunda yer alan katılımcılar açısından ise C# programlama etkinliklerinin uygulama yapma olanağı sağlayarak öğreneni daha aktif yaptığı söylenebilir. Buna göre katılımcı görüşleri doğrultusunda Scratch ile programlama öğretiminin deney grubu için programlama mantığını kazandırma, öğrenme sürecini kolaylaştırma, motivasyonlarını ve yaratıcılıklarını artırma açısından genel olarak üstünlük sağladığı, akış diyagramlarının ise kontrol grubu açısından süreç içinde pasif olma gibi nedenlerden dolayı daha çok sınırlılık oluşturduğu söylenebilir. C# programlama öğretiminin ise deney grubu açısından programlama yapma olanağı sağlaması yönüyle üstünlük sağladığı, kontrol grubu açısından ise aktif katılım sağlama açısından üstünlük sağladığı söylenebilir.

Katılımcıları ders süresince güdüleyen veya zorlayan durumlara ilişkin görüşler incelendiğinde; deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı etkinlikleri kapsamında motivasyonlarını artıran birçok etmenin olduğu, kontrol grubunda yer alan katılımcıların ise akış diyagramı ile problem çözme etkinlikleri kapsamında motivasyonlarını düşüren birçok etmenin olduğu görülmüştür. Nitekim deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinin eğlenceli ve zevkli olması, görsel olması, test etme ve dönüt olanağı sağlaması, uygulamalı olması, bir ürün ortaya koyma olanağı sağlaması, kod gerektirmemesi, basitleştirmesi, zihinde canlandırması, birlikte yapma ve akran iletişimi sağlaması, dersin devamlılığını sağlaması, merakı artırması ve canlandırma olanağı sunması açısından motivasyonu artıran etmenler olarak görüş bildirmişlerdir. Bu bulgular Alimsi ve Winters (2010) tarafından yapılan çalışmada ortaya çıkan test etme ve dönüt olanağı sağlaması görüşüyle, Malan ve Leitner (2007) tarafından yapılan çalışmada ortaya çıkan kod

gerektirmemesi görüşüyle, Briggs (2013) tarafından yapılan çalışmada ulaşılan akranlar ile iletişim görüşüyle benzerlik göstermektedir. Ayrıca Alimisi ve Winters (2010), Chui (2014), Malan ve Leitner (2007), Maloney ve arkadaşları (2008), Ozoron ve arkadaşları (2012), Salant ve arkadaşları (2013) tarafından yapılan çalışmalarda sözü edilen eğlenceli olmasıyla ilgili görüşlerle benzerlik göstermektedir. Buna göre katılımcıların görüşleri doğrultusunda; deney grubunda uygulanan Scratch ile oyun tasarımı etkinliklerinin, akış diyagramlarına göre katılımcıların motivasyonunu daha çok artırdığı söylenebilir. Ayrıca öğrenci görüşleri doğrultusunda deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecinde kazanmış oldukları programlama mantığının C# programlama öğretimi sürecinde motivasyonu artıran bir etmen olduğu, kontrol grubunda yer alan katılımcıların ise akış diyagramı ile problem çözme sürecinde programlama mantığını kavramada yaşanan sıkıntının C# programlama öğretimi sürecinde motivasyonu düşüren bir etmen olduğu söylenebilir. C# programlama öğretimi sürecinde deney grubu açısından sürekli konsol ile çalışmanın ve kod hataları ile karşılaşmanın motivasyonu düşüren bir etmen olduğu, kontrol grubu açısından ise uygulama yapmanın ve test etme olanağı olmasının motivasyonu artıran bir etmen olduğu söylenebilir. Bu doğrultuda deney grubunda yer alan katılımcılarının ders süresince yapılan etkinliklerin motivasyonlarını artığına ilişkin görüşlerinin daha çok olduğu, kontrol grubunda yer alan katılımcılarının ise ders süresince yapılan etkinliklerin motivasyonlarını düşürdüğüne ilişkin görüşlerinin daha çok söylenebilir.

Katılımcıların ders süresince öğrendikleri bilgileri nasıl kullanacaklarına ilişkin görüşleri incelendiğinde hem deney hem de kontrol grubunda yer alan katılımcıların temel programlama mantığının aynı olduğunu ve tek farkın kodlar olduğunu düşündükleri görülmüştür. Aynı zamanda deney grubunda yer alan katılımcıların Scratch ile oyun tasarımı sürecinde edindikleri bilgileri eğitim materyali tasarlamak amaçlı başka derslere de transfer ettikleri görülmüştür. Ayrıca deney grubundaki katılımcıların geçmiş öğrenme deneyimlerini, Scratch ile oyun tasarım sürecinin sağladığı üstünlüklerden dolayı C# programlama dili dersine daha çok transfer ettikleri söylenebilir. Kontrol grubundaki katılımcıların ise geçmiş öğrenme deneyimlerini, akış diyagramlarıyla problem çözme sürecinde oluşan sınırlılıklarından dolayı transfer etmedikleri söylenebilir. Ortaya çıkan bu sonuçlar doğrultusunda; Scratch ile oyun tasarımı etkinliklerinin temel programlama mantığının kazandırılmasında ve yeni bir

programlama diline transfer için etkili olduğu söylenebilir. Bu bulgu Westcott (2008) tarafından yapılan çalışmada ulaşılan öğrencilerin Scratch ile öğrendikleri programlama mantığını programlama dillerine aktarabildikleri sonucuyla benzerlik göstermektedir.

Katılımcıların ders sonunda programlamaya karşı görüşleri incelendiğinde; deney grubundaki katılımcıların Scratch ile oyun tasarımı sürecinin sağladığı üstünlüklerden dolayı ders sonunda programlamaya karşı görüşlerinin olumlu olarak değiştiği ve bu etkinin C# programlama sürecine geçildiğinde de geçmiş öğrenme deneyimi olarak devam ettiği söylenebilir. Ancak C# programlama sürecine geçince bazı katılımcılarda kod hataları ve konsol ekranı yüzünden programlamaya karşı görüşlerinde olumsuz bir değişim olduğu söylenebilir. Kontrol grubundaki katılımcıların ise; akış diyagramları ile problem çözme sürecinin oluşturduğu sınırlılıklardan ve yaşanan başarısızlıktan dolayı programlaya karşı olan görüşlerinin olumsuz olarak değiştiği ve bu etkinin C# programlama sürecine geçildiğinde de geçmiş öğrenme deneyimi olarak devam ettiği söylenebilir. Katılımcıların gelecekte programlama ile ilgili bir işte çalışmaya ilişkin görüşleri incelendiğinde; deney grubunda yer alan katılımcıların daha çok programcılığı bir meslek olarak seçme düşüncesinde oldukları görülmektedir. Bu kararı almalarında katılımcıların programlama mantığını kavramış olmalarının, programlamayı zevkli ve kolay olarak düşünmelerinin etkili olduğu katılımcı görüşleri doğrultusunda söylenebilir. Kontrol grubunda yer alan katılımcıların ise daha çok programcılığı bir meslek olarak seçme düşüncesinde olmadıkları görülmüştür. Bu kararı almalarında, katılımcıların programlama mantığını kavrayamamış olmalarının, programlamayı sıkıcı ve zor olarak düşünmelerinin etkili olduğu katılımcı görüşleri doğrultusunda söylenebilir. Buna göre deney grubunda gerçekleştirilen Scratch ile oyun tasarımı sürecinin deney grubundaki katılımcıların bu kararı almalarına olumlu yönde etki ettiği, akış diyagramları ile problem çözme sürecinin kontrol grubundaki katılımcıların bu kararı almalarına olumsuz yönde etki ettiği söylenebilir.

Hem nicel hem de nitel veriler incelendiğinde Scratch ile oyun tasarımı etkinliklerinin eğlenceli olması, öğrenmeyi kolaylaştırması, süreci somutlaştırması ve öğrenenlere süreç içinde yaparak öğrenme fırsatı sağlaması gibi üstünlüklerinin öğrenenlerin motivasyonunu artırdığı ve bu etkinin programlama mantığını öğrenme sürecine dolayısıyla da programlama dilleri öğrenme sürecine olumlu etki ettiği

görülmektedir. Buna göre öğrenenlerin Scratch ile oyun tasarımı sürecinde edindikleri deneyimleri sayesinde programlama mantığını öğrendikleri ve bu deneyimi yeni bir programlama diline transfer edebildikleri söylenebilir. Bunun aksine mevcut ders programında yer alan akış diyagramları ile problem çözme etkinliklerinin ise zor ve sıkıcı olması, süreci somutlaştırmada yetersiz kalması ve öğrenenlerin süreç içinde pasif kalması gibi sınırlılıklarının öğrenenlerin motivasyonlarını düşürdüğü ve bu etkinin programlama mantığını öğrenme sürecine dolayısıyla da programlama dilleri öğrenme sürecine olumsuz etki ettiği görülmektedir. Buna göre mevcut ders programına yer alan akış diyagramları ile problem çözme etkinliklerinin programlama mantığı öğretiminde yetersiz kaldığı, bu nedenle öğrenenlerin süreçte edindikleri deneyimi yeni bir programlama diline transfer etmekte sıkıntı yaşadıkları söylenebilir. Bununla birlikte nitel bulguların nicel bulgular sonucunda ortaya çıkan; deney grubundaki motivasyon artışını ve kontrol grubundaki motivasyon düşüşünü destekler nitelikte olduğu da görülmektedir. Sonuç olarak; Scratch ile oyun tasarımı etkinliklerinin öğrenenlerin motivasyonlarını artırdığı, bundan dolayı hem programlama mantığı hem de programlama dilleri öğretimin yapıldığı süreçte öğrenenlerin başarılarını, akış diyagramları ile problem çözme etkinliklerine göre daha fazla artırdığı söylenebilir.

Öneriler

Araştırma sonuçları ele alındığında uygulamaya ve yapılacak araştırmalara yönelik aşağıdaki öneriler geliştirilebilir.

Uygulamaya Yönelik Öneriler

- Programlama mantığını kazandırmak için programlamaya giriş derslerinde akış diyagramları yerine Scratch, Alice gibi benzer görselleştiriciler kullanılarak oyun tasarımı etkinlikleri gerçekleştirilebilir.
- Hem programlama mantığı hem de programlama dili öğretiminde, öğrenenlerin başarı ve motivasyonlarını artırmak için öğrenen merkezli öğrenme ortamları tasarlanabilir.
- Programlama dili derslerinde; öğrenme sürecini daha eğlenceli hale getirmek için kodlar ile problem çözümü etkinlikleri yerine yine kod kullanarak oyun tasarımı etkinlikleri gerçekleştirilebilir.

- Hem programlama mantığı hem de programlama dili öğretiminde, öğrenenlerin başarı ve motivasyonlarını artırmak için öğrenenlerin birlikte öğrendiği ve öğrenenleri araştırmaya teşvik eden öğrenme ortamları tasarlanabilir.
- Üniversitelerde öğretmen adayları için programlama öğretiminin nasıl yapılacağına ilişkin bir rehber olması açısından; Scratch ya da benzeri görselleştiriciler kullanılarak oyun tasarımı etkinlikleri içeren eğitimler verilebilir.
- Üniversitelerde bilişim teknolojileri öğretmenliği alanında eğitim veren bölümlerde temel programlama derslerinin içeriği yeniden düzenlenebilir.
- Programlama mantığının kazandırmak ve programlamayı sevdirmek için yalnızca lisans düzeyinde değil, daha erken yaşlarda da (ilkokul, ortaokul ve lise) farklı bilişim dersleri altında Scratch ya da benzeri görselleştiriciler kullanılarak oyun tasarımı etkinlikleri yapılabilir.
- Beşinci ve altıncı sınıflarda yer alan Bilişim Teknolojileri ve Yazılım dersi içeriği MEB tarafından güncellenerek Scratch ile programlama öğretimi içerik olarak ele alınabilir.
- Mesleki ve Teknik Eğitim okullarında Mesleki Eğitim ve Öğretim Sistemini Güçlendirme Projesi (MEGEP) kapsamında geliştirilen ders modülleri MEB tarafından güncellenerek Scratch ile programlama öğretimi içerik olarak ele alınabilir.
- Tüm eğitim kademelerinde Scratch ile oyun tasarımı etkinlikleri yalnızca programlama becerisini kazandırmak için değil, aynı zamanda matematik ve fen bilimleri gibi derslerde de içerik öğretimi için kullanılabilir.
- Tüm eğitim kademelerinde Scratch ile oyun tasarımı etkinlikleri yalnızca programlama becerisini kazandırmak için değil, aynı zamanda yaratıcılık ve problem çözme gibi becerilerin de kazandırılması için kullanılabilir.
- Scratch'in diziler gibi bazı temel yapılar için yetersizliğini ortadan kaldırmak için uygulama ve oyunlar yeniden düzenlenebilir ve sayısı arttırılabilir.
- Scratch'in oyun tasarımında bloklar açısından yetersiz ve sınırlı kalmasını ortadan kaldırmak için farklı görselleştiriciler kullanılabilir.

Yapılacak Araştırmalara Yönelik Öneriler

- Scratch ile oyun tasarımı etkinliklerinin programlama başarısına ve motivasyona etkisinin araştırılması için;
    - BÖTE bölümü ile sınırlı kalmayarak farklı bilişim teknolojileri alanlarında,
    - üniversite öğrencileri ile sınırlı kalmayarak farklı eğitim kademelerinde (lise -ortaokul -ilkokul) benzer deneysel araştırmalar gerçekleştirilebilir.
- Oyun tasarımı etkinliklerinde Scratch yerine farklı görselleştiricilerin kullanıldığı ve akış diyagramları ile karşılaştırıldığı çalışmalar yapılabilir.
- Scratch'in programlama başarı ve motivasyonlarına etkisini incelemek amacıyla farklı görselleştiriciler ile karşılaştırıldığı çalışmalar gerçekleştirilebilir.
- Cinsiyet gibi yeni bağımsız değişkenler eklenerek benzer çalışmaların çıkarımları güçlendirilebilir.
- Katılımcı sayısı arttırılarak istatistiksel güç bazında üstünlük sağlanabilir ve bulgular farklı bağlamlarda doğrulanabilir.
- Scratch ve benzeri araçlar ile oyun tasarımı etkinliklerinin öğrenenlerin yaratıcılık, problem çözme ve eleştirel düşünme gibi 21. yüzyıl becerilerine etkisini inceleyen çalışmalar gerçekleştirilebilir.
- Odak grup görüşmeleri yanında gözlem ve doküman incelemesi gibi nitel yöntem ve tekniklerle desenlenen çalışmalar gerçekleştirilebilir.
- Çalışmada kullanılan Scratch uygulamaları yeniden düzenlenerek benzer ampirik çalışmalar gerçekleştirilebilir.
- Çalışmada gerçekleştirilen C# programlama dili öğretimi yerine Java gibi farklı programlama dilleri kullanılabilir.
- Kalıcılığın etkisinin incelenmesi için öğrenenlerin diğer programlama derslerindeki başarı ve motivasyonları incelenebilir.

EKLER

EK A- MEHMET AKİF ERSOY ÜNİVERSİTESİ EĞİTİM FAKÜLTESİ UYGULAMA İZNİ

Evrak Tarih ve Sayısı: 08/10/2014-34426

T.C.
MEHMET AKİF ERSOY ÜNİVERSİTESİ
Eğitim Fakültesi Dekanlığı

Sayı : 92934064-900-
Konu : Öğr.Gör.Osman EROL

BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ BÖLÜM BAŞKANLIĞINA

İlgi : 01/10/2014 tarihli, 34077 sayılı ve "Öğr.Gör.Osman EROL" konulu yazı

Bölümünüz öğretim üyelerinden Öğr.Gör.Osman EROL'un "Scratch ile Programlama Öğretiminin Motivasyon ve Başarıya Etkisi" başlıklı tez çalışmasını Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü 3. yarıyıl dersi olan Programlama Dilleri I dersinde 08.09.2014-26.12.2014 tarihleri arsında gerçekleştirmesi Dekanlığımızca uygun görülmüştür.
Bilgilerinizi ve gereğini rica ederim.

Yrd. Doç. Dr. Harun ŞAHİN
Dekan a.
Dekan Yardımcısı

Evrakı Doğrulamak İçin : https://ebys.mehmetakif.edu.tr/enVision/Dogrula/L5N8KN

İstiklal Yerleşkesi 15030 / BURDUR
Telefon:+90 248 213 40 00 Faks+90 248 213 41 60
e-Posta egitim@mehmetakif.edu.tr , Elektronik Ağ:http://egitim.mehmetakif.edu.tr

Ayrıntılı bilgi için irtibat: İlknur Patır
Evrak Pin Kodu: 31361

Kep Adresi : maku@hs01.kep.tr

.

EK B- GÖS ÖLÇEĞİ KULANIM İZNİ

Öğr. Gör. Osman EROL
Bilgisayar ve Öğretim Teknolojileri Eğitimi
Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi

11.06.2014 14:15, Ozcan Erkan Akgun yazmış:

> Tekrar Merhaba Osman,
>
> Ölçek ve gerekli olabilecek bazı bilgiler ekte. Bilimsel ve etik ilkelere bağlı kalmak ve çalışmamıza atıfta bulunmak koşullarıyla ölçeğimizi kullanabilirsin.
>
> İyi çalışmalar dilerim.
>
> Özcan Akgün
>
> > 2014-06-01 19:56 GMT+03:00 Osman EROL <oerol@mehmetakif.edu.tr>:
> >
> > Merhaba Hocam;
> >
> > "Güdülenme ve Öğrenme Stratejileri Ölçeğinin Türkçe Formunun Geçerlik ve Güvenirlik Çalışması" başlıklı çalışmanızda Türkçe uyarlamasını yapmış olduğunuz " Güdülenme ve Öğrenme Stratejileri Ölçeğini" Doç.D A.Aşkım KURT danışmanlığında gerçekleştirdiğim doktora tezim kapsamında izniniz olursa kullanmak istiyorum. Saygılarımla

EK C- GÜDÜLENME VE ÖĞRENME STRATEJİLERİ ÖLÇEĞİ

MOTİVASYON ÖLÇEĞİ

Bu çalışmanın amacı sizin **Programlama dilleri derslerine ilişkin motivasyon düzeyiniz**i belirlemektir. Soruların doğru bir cevabı yoktur. Verdiğiniz samimi cevaplar çalışmanın geçerliliği açısından önemlidir. Çalışmanın verileri tamamen **gizli kalacak ve bilimsel bir çalışma** için kullanılacaktır. Teşekkür ederim.

Öğr. Gör. Osman EROL

- **Öğrenci numaranızın son üç hanesi**: ( ) * Lütfen eksiksiz ve doğru giriniz.
- **Cinsiyetiniz:** Erkek ( ) Kadın ( )
- **Hangi lise mezunusunuz?**
  ( ) Genel Lise/Anadolu Lisesi / And. Öğretmen Lisesi/ Fen Lisesi
  ( ) Meslek Lisesi / Teknik Lisesi / Çok Programlı Lise

Soruları yanıtlamak için aşağıdaki ölçütleri kullanın. Soruda geçen ifade sizin için kesinlikle doğru ise (7)'yi; sizinle ilgili **kesinlikle yanlışsa (1)**'i işaretleyin. Eğer ifadenin size göre doğruluğu bunlardan farklı ise sizin için en uygun düzeyi gösteren (1)'le (7) arasındaki rakamı işaretleyin.

	Benim için kesinlikle Yanlış	1	2	3	4	5	6	7	**Benim için kesinlikle Doğru**

1.	**Bunun gibi bir derste beni gerçekten çalışmaya zorlayacağına inandığım ders materyallerini tercih ederim, bu sayede yeni şeyler öğrenebilirim.**	(1)	(2)	(3)	(4)	(5)	(6)	(7)
2.	Ancak uygun bir şekilde çalışırsam bu dersin konularını öğrenebilirim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
3.	Sınavdayken diğer öğrencilerden daha yetersiz olduğumu düşünürüm.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
4.	Bu derste öğrendiklerimi diğer derslerde de kullanabilirim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
5.	Bu dersten çok iyi bir not alacağıma inanıyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
6.	Bu derste okumam için verilecek en zor konuları bile anlayacağımdan eminim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
7.	Benim için en tatmin edici şey sınıfta iyi bir not almaktır.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
8.	Sınavda soruları çözerken, sınav kağıdının diğer bölümlerindeki yanıtlayamayacağım soruları düşünürüm.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
9.	Eğer bu dersi öğrenemiyorsam bu benim kendi hatamdır.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
10.	Bu derste verilen kaynakları (kaynak materyalleri) öğrenmek benim için önemlidir.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
11.	Bu derste benim için en önemli şey, genel not ortalamamı yükseltmektir, yani bu dersteki asıl amacım iyi bir not almaktır.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
12.	Bu derste anlatılan temel kavramları anlayabileceğim konusunda kendime güveniyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
13.	Eğer yapabilirsem, bu sınıftaki diğer öğrencilerin hepsinden daha yüksek not almak isterim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
14.	Sınavdayken başarısızlığı ve bunun doğuracağı sonuçları düşünürüm.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
15.	Bu derste öğretmenin anlatacağı en zor konuyu bile anlayacağıma güveniyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
16.	Bunun gibi bir derste, zor olsalar bile, bende merak uyandıran ders materyallerini tercih ederim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
17.	Bu dersle ilgili konulara oldukça ilgi duyuyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
18.	Yeterince çalışırsam dersi anlayabilirim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
19.	Sınavdayken kendimi rahatsız ve morali bozuk hissederim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
20.	Bu dersteki ödevleri ve sınavları mükemmel yapabileceğim konusunda kendime güveniyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
21.	Bu derste başarılı olmayı bekliyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
22.	Bu derste benim için en tatmin edici şey içeriği mümkün olduğunca çok anlayabilmektir.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
23.	Bence bu derste kullanılan materyaller dersi öğrenmem için faydalıdır.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
24.	Eğer olanak tanınırsa, iyi not almamı sağlamayacak olsa bile en iyi şekilde öğrenmemi sağlayacak ödevleri seçerim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
25.	Dersi yeterince anlayamıyorsam, bu yeterince çalışmadığım içindir.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
26.	Bu dersin konularını seviyorum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
27.	Bu dersin konularını öğrenmek benim için çok önemlidir.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
28.	Sınavdayken kalbimin hızla çarptığını hissederim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
29.	Eminim ki bu derste öğretilen tüm becerileri ustalıkla yapabilirim.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
30.	Sınıfta başarılı olmak isterim; çünkü yeteneğimi aileme, arkadaşlarıma üstlerime ve diğerlerine göstermek benim için önemlidir.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**
31.	Dersin zorluğunu, öğretmeni ve becerilerimi dikkate aldığımda, bence bu derste başarılı olurum.	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**

EK D- BAŞARI TESTİ

Adı Soyadı…:
No...:

Açıklama: Bu sınav Programlama I dersi konularına yönelik başarıyı ölçmeyi amaçlamaktadır. Test 20 adet çoktan seçmeli ve 4 adet problemden oluşmaktadır. Süreniz 80 dakikadır.

Öğr. Gör. Osman EROL

```
string a= "MehmetAkifErsoy";
int x=1250;
int b=a.Length;
float c;
c = (float)x / b;
Console.WriteLine (c);
```

1. Yukarıdaki program çalıştığı zaman ekranda hangi değer yazar?

A) 0 B) 83 C) 83,33334 D) 83,5 E) 84

```
Başla
OKU A
Eğer  .....1.....  ise

      .....2.....
Değil İse
      .....3.....
Bitir
```

2. Yukarıda algoritması verilen programda boş bırakılan yerler ile ilgili olarak aşağıdakilerden hangisi yanlıştır?

A) 1. bölüm bir döngü başlangıcını ifade eder
B) 1. bölüme A< 5 VE A > 0 ifadesi yazılabilir
C) 2. bölüm koşul sağlanma durumunda yapılacak işlemleri ifade eder
D) Yukarıdaki program bir karar verme yapısını ifade eder
E) 3. bölüme tekrardan bir Eğer yapısı getirilebilir

1. Başla
2. Oku B, i=0,
3. Döngü
   - i=B olana kadar tekrarla
   - i= i+1
   - Yaz B
4. Döngü Sonu
5. Bitir

3. Yukarıda algoritması verilen programda B değişkenine girilen değer **250**' dir. Buna göre programla ilgili olarak aşağıdaki ifadelerden hangisi doğrudur?

A) 250 nin karesi yazar
B) 250 sayısını ikiyüz elli defa yazar
C) 1 den 250 ye kadar tüm sayıları yazar
D) 250 den geriye sayıları yazar
E) Sonsuz döngü oluşur

```
int [] dizi = { 18,-33,-8, 401, 712,280,141 };
Array.Sort(dizi);
Console.WriteLine(dizi[          ]);
```

4. Yukarıdaki kod çalıştırıldığında ekranda **401** değeri yazmaktadır. Buna göre boş bırakılan bölüme aşağıdakilerden hangisi gelmelidir?

A) 0 B) 1 C) dizi.Length
D) dizi.Length-1 E)dizi.Length-2

1. Başla
2. i=0
3. Dizi A (10)
4. Döngü
   - i=10 olana kadar tekrarla
   - i= i+1
   - Oku A (i)
5. Döngü Sonu
6. Döngü
   - i =0 olana kadar tekrarla
   - Yaz A (i)
   - i = i -2
7. Döngü Sonu

5. Yukarıda algoritması verilen program çalıştırıldığında "A" dizisi için klavyeden okunan değerler sırasıyla - **3, 10, 54, 15, 113, -83, 78, -8, 514, 41** sayılarıdır. Buna göre bu programın ekran çıktısı aşağıdakilerden hangisidir?

A) -83 -8 -3 10 15 41 54 78 113 514
B) -3 54 113 78 514
C) 514 113 78 54 41 15 10 -3 -8 -83
D) 41 -8 -83 15 10
E) -3 10 54 15 113 -83 78 -8 514 41

```
Başla
Yaz ("Adınızı giriniz" )
Oku (A)
Yaz ("Soyadınızı giriniz")
Oku (B)
Yaz ("Unvanınızı giriniz")
Oku (C)
D= "Hoş geldiniz"
Yaz (C,B,A,D)
Bitir
```

6. Yukarıdaki algoritması verilen programa klavyeden sırasıyla "**Ayşe**", "**Yener**", "**Miss**" ifadeleri girilmiştir. Programın ekran çıktısı aşağıdakilerden hangisidir?

A) Hoş geldiniz Ayşe Yener
B) Hoş geldiniz Miss Ayşe Yener
C) Miss Yener Ayşe Hoş geldiniz
D) Hoş geldiniz Miss Yener
E) Miss Ayşe Yener Hoş geldiniz

1 .Basla
2. Oku A // SCRATCH kelimesi girilmiştir.
3. T = Uzunluk (A)
4. Dizi B(T)
5. Döngü
   - i= T olana kadar tekrarla
   - i=i+1
   - B(i) = A nın (i.) harfi
6. Döngü Sonu
7. Döngü
   - i=0 olana kadar tekrarla
   - i= i-1
   - Yaz B(i)
8. Döngü Sonu
9. Bitir

7. Yukarıda algoritması verilen program için aşağıdaki ifadelerden hangisi doğrudur?

A) Ekrana SCRATCH yazar
B) Ekrana HCTARCS yazar
C) Ekrana 7654321 yazar
D) Ekrana 1234567 yazar
E) Ekrana 7 yazar

1

```
int i;
int toplam = 0;
int x = Convert.ToInt32 (Console.Readline ());
for (i = 0; i <= x; i+=1)
{
toplam += i;
}
Console.WriteLine(toplam);
```

8. Yukarıdaki C# programında klavyeden 10 değeri girilirse ekrana hangi değer yazar?

A)0
B)10
C) Sonsuz değerde toplama yapar
D) 55
E)110

```
double  x=6;
double y=4;
double  z=5;
İf  ((x < z) || ( x =y))
{
x= Math.Pow (x,2);
Console.WriteLine (x);
}
else if  (( y<z ) && (x < z))
{
y= Math.Pow (y,3);
Console.WriteLine  (y);
}
else
{
z= Math.Pow (z,4);
Console.WriteLine (z);
```

9. Yukarıdaki program çalıştırıldığında ekrana hangi değer yazar?

A) 5 B) 6 C) 36 D 64 E)625

- Başla
- i, J, T
- Döngü
  - i=50 olana kadar tekrarla
  - i=i+1
  - Yaz "A"
  - J=0
  - Döngü
    - J= 3 olana kadar tekrarla
    - J=J+1
    - Yaz "B"
    - T=0
    - Döngü
      - T= 5 olana kadar tekrarla
      - T=T+1
      - Yaz "C"
    - Döngü Sonu
  - Döngü Sonu
- Döngü sonu
- Bitir

10. Yukarıda algoritması verilen program için aşağıdakilerden hangisi doğrudur?

A) 50 kez A, 150 kez B ve 750 kez C yazar
B) Sonsuza kadar ekrana ABC yazar
C) 50 kez A, 3 kez B ve 5 kez C yazar
D) 5 kez ABC, 15 kez BC ve 50 kez A yazar
E) 750 kez ABC yazar

```
int x=50,y=50,z=100;
x- = 5;
y- = 7;
x- = y ;
z- = x;
```

11. Yukarıdaki program parçası çalıştığı zaman **z'** nin yeni değeri kaç olur?

A) -55 B) 2 C) 55 D) 57 E) 98

1 Adım: Başla
2 Adım: S= 0, A=1
3 Adım: Eğer S< 10 ise
4 Adım: A = A+A
5 Adım: Git 3. Adım
6 Adım: Değilse
7 Adım: Yaz A
8 Adım: Bitir

12. Yukarıdaki algoritması verilen program çalıştığı zaman mantık hatası oluşmakta ve sürekli tekrar eden sonsuz bir döngü oluşturmaktadır. Programı çalışır hale getirmek için aşağıdakilerden hangisinin yapılması gerekmektedir?

A) 7. Adım ile 6. Adım' ı yer değiştirmek
B) 1. Adımdan sonra A değişkenini 1 arttırmak
C) 3. Adımdan sonra S değişkenini 1 arttırmak
D) 5. Adım da "Git 3. Adım" ifadesini "Git 1. Adım" şeklide değiştirmek
E) 2. Adımda "S=0" yerine "S=1" kullanmak

```
int sayi =0;
int[] A= new int [12] { -4, 4, -33, 8, 23,54, 64, 35,-50, 121, 314, -8 };
for (int i = 0; i <A.Length; i++)
{
   sayi = A[i];
   if (sayi%2 == 0 ||  sayi < 0 )
 {
   Console.WriteLine(sayi);
 }
}
```

13. Yukarıdaki program çalıştırıldığında ekrana aşağıdaki sayılardan hangisi yazdırılmaz?

A) -33 B) -4 C) 8 D) 23 E) 64

1. Adım: Başla
2. Adım: Yaz ("Yarı çap değerini giriniz?")
3. Adım: Oku R
4. Adım: P=3,14
5. Adım: Cevre = 2 * P * R
6. Adım: Yaz ("çevresi=" Cevre)
7. Adım: Bitir

14. Klavyeden yarı çapı verilen çemberin çevresini hesaplayan programın algoritması yukarıdaki gibidir. Buna göre bu programla ilgili olarak aşağıdaki ifadelerden hangisi doğrudur?

I. R bir değişkendir
II . 3. adım çıktı sağlar
III. 6. adımdan sonra tekrar 3. adıma gitmektedir
IV P' ye atanan değer sabittir

A) Yalnız I B) I - IV C) I- II- III
D) I - II E) I - II –IV

```
int x;
Console.WriteLine ("Bir Sayı Giriniz");
x=Convert.ToInt32(Console.ReadLine());

if ( (x < 0)    .....1.....  ( x .....2..... 0) )
   {
   Console.WriteLine ("Sayı negatiftir");
   }
       .....3.....
{
  Console.WriteLine ("Sayı pozitiftir");
}
```

15. Yukarıdaki programda girilen sayıya göre eğer sayı **0' dan küçük veya 0' a eşit ise** "Sayı negatiftir" **değil ise** "Sayı pozitiftir " yazmaktadır. Buna göre numaralandırılmış boşluklu bölümler ile ilgili olarak hangisi ya da hangileri doğrudur?

I. 3. bölüme "else" ifadesi gelmelidir
II. 1. bölüme "&&" işareti gelmelidir
III. 2. bölüme "==" işareti gelmelidir.
IV. 3. bölüme "(x != 0) ifadesi yazılmalıdır.

A) Yalnız I B) I ve II C) II ve III
D) I ve III E) I, II, III ve IV

1. Başla
2. Oku A
3. i=0, F=1
4. Döngü
   - i=A olana kadar tekrarla
   - i=i+1
   - F= F* i
5. Döngü Sonu
6. Yaz F
7. Bitir

16. Yukarıda algoritması verilen programda A değişkenine klavyeden **6** değeri girilirse programın ekran çıktısı ne olur?

A) 6 B)6 5 4 3 2 1 C) 720
D)1 2 3 4 56 E) Sonsuza kadar 6 yazar

1. Başla
2. Oku A
3. S=0
4. Eğer S< A ise
5. S= S+1
6 . Yaz A
7. Git 4. Adım
8. Değilse
9. Bitir

17. Yukarıda algoritması verilen programda A değişkenine **5** değeri girilirse programın ekran çıktısı aşağıdakilerden hangisi olur?

A) 55555

B) 5
55
555
5555
555555

C) 54321

D) 55555
5555
555
55
5

E) 12345

```
int i=2;
int top =0;
while (i < 12)
{
top=top+i;
i = i + 3;
Console.WriteLine(top);
}
```

18. Yukarıdaki kod çalıştırıldığında ekrana aşağıdaki sayılardan hangisi **yazdırılmaz**?

A) 26 B) 18 C)15 D) 7 E) 2

1. Tutar = Ekmek * 1 + Yumurta * 0,25 + Margarin * 2
2. Başla
3. Yaz (“ Borcunuz =”, Toplam)
4. Toplam = Tutar + Vergi
5. Bitir
6. Ekmek = 1 TL, Yumurta = 0,25 TL , Margarin= 2 TL
7. Yaz (“Kaç ekmek istiyorsunuz”)
Oku (Ekmek)
Yaz (“ Kaç tane yumurta istiyorsunuz”)
Oku (Yumurta)
Yaz (“ Kaç paket margarin istiyorsunuz”)
Oku (Margarin)
8. Vergi = Tutar * 18 / 100

19. Yukarıda algoritması karışık şekilde verilen program bir marketin satış programıdır. Kullanıcılar marketten ekmek, yumurta ve margarin ürünlerinden kaçar adet istediklerini klavyeden girmekte ve program ürünlerin birim fiyatları üzerinden ne kadar ödeme yapılması gerektiğini hesaplamaktadır. Bu bilgilere göre bu programın algoritmasının doğru sıralanışı aşağıdakilerden hangisidir? (Toplam miktara % 18 KDV eklenmektedir.)

A) 2 - 6- 7 - 1 - 8 - 4 - 3 - 5
B) 2 - 4 - 6- 3 - 7 - 8 - 1 - 5
C) 2 - 3 - 8 - 7 - 4 - 6 - 1 - 5
D) 2 - 1 - 7 - 6 - 3 - 8 - 4 - 5
E) 2 - 8 - 3 - 6 - 1- 7 - 4 – 5

```
string yaz1="";
string yaz2= "";
string kelime="Seni";
string kelime2= "Seviyorum";
for (int i=0; i< kelime.Length; i+=1)
{
   yaz1 = kelime;
   Console.WriteLine(yaz1);
for (int j=0; j< kelime2.Length; j+=3)
{
    yaz2 = kelime2;
   Console.WriteLine(yaz2);
}
}
```

20. Yukarıda yer alan programın ekran çıktısı ile ilgili olarak aşağıdakilerden hangisi doğrudur?

A) Önce "Seni" yazar Sonra 1 defa "Seviyorum" yazar VE bu durum 4 kez tekrarlanır

B) Önce "Seni" yazar Sonra 9 defa "Seviyorum" yazar VE bu durum 4 kez tekrarlanır

C) Önce 4 defa "Seni" yazar Sonra 9 defa "Seviyorum" yazar.

D) Önce "Seni" yazar Sonra 1 defa "Seviyorum" yazar VE bu durum 9 kez tekrarlanır.

E) Önce "Seni" yazar Sonra 3 defa "Seviyorum" yazar VE bu durum 4 kez tekrarlanır.

Soru					
1.	(A)	(B)	(C)	(D)	(E)
2.	(A)	(B)	(C)	(D)	(E)
3.	(A)	(B)	(C)	(D)	(E)
4.	(A)	(B)	(C)	(D)	(E)
5.	(A)	(B)	(C)	(D)	(E)
6.	(A)	(B)	(C)	(D)	(E)
7.	(A)	(B)	(C)	(D)	(E)
8.	(A)	(B)	(C)	(D)	(E)
9.	(A)	(B)	(C)	(D)	(E)
10.	(A)	(B)	(C)	(D)	(E)
11.	(A)	(B)	(C)	(D)	(E)
12.	(A)	(B)	(C)	(D)	(E)
13.	(A)	(B)	(C)	(D)	(E)
14.	(A)	(B)	(C)	(D)	(E)
15.	(A)	(B)	(C)	(D)	(E)
16.	(A)	(B)	(C)	(D)	(E)
17.	(A)	(B)	(C)	(D)	(E)
18.	(A)	(B)	(C)	(D)	(E)
19.	(A)	(B)	(C)	(D)	(E)
20.	(A)	(B)	(C)	(D)	(E)

Çoktan Seçmeli Testler bitmiştir. Problemler bölümüne geçebilirsiniz.

PROBLEMLER

Açıklama: Lütfen her bir soruyu dikkatlice okuyunuz Her bir problemin algoritması ve C# kodları ayrı ayı yazılacaktır.

21 Problem 1: Bir X firması çalışanlarının maaşını hesaplayan bir programa ihtiyaç duymaktadır. Bir yazılımcı olarak sizden "Bir çalışanın çalışma saati ve saatlik ücreti klavyeden girildikten sonra aşağıdaki kurallara göre; çalışanın net maaşını hesaplayarak ekrana yazan programı" hazırlamanız istenmektedir.

Kurallar:

- Brüt maaş çalışma saati ile saatlik ücretin çarpımı ile bulunur.
- Brüt maaştan %10 SSK kesintisi kesilmektedir
- Brüt maaştan %15 gelir vergisi kesilmektedir.
- Net maaş; brüt maaştan kesintilerden sonra geriye kalan maaştır.

Yukarıdaki kuralları dikkate aldığınızda:

a)Programın algoritmasını yazınız
b) Programın C# kodlarını yazınız.

22 Problem 2:Bir matematik öğretmeni kenar değerlerine göre bir üçgenin çizilip çizilmeyeceğini belirleyen bir eğitim yazılımına ihtiyaç duymaktadır. Buna göre bir yazılımcı olarak sizden; klavyeden 3 adet kenar uzunluğu girildikten sonra aşağıdaki üçgen çizilme kurallarına göre;

a. Üçgenin çizilip çizilemeyeceğini belirleyen ve üçgen çizilemez ise bunu ekrana yazan;
b. Eğer üçgen çizilirse üçgenin çeşidini(ikizkenar, çeşitkenar, eşkenar) ekrana yazan programı hazırlamanız istenmektedir.
   Lütfen aşağıdaki kurallara dikkat ediniz.

Üçgenin çizilip çizilemeyeceği ile ilgili kuralar:

a. Bir kenarı diğer iki kenarının toplamından küçük olmak zorundadır. (Örneğin a> b+c gibi)
b. Bir kenar diğer iki kenarın farkından büyük olmak zorundadır. (Örneğin a<b-c gibi)

Yukarıdaki kuralları dikkate aldığınızda:

a)Programın algoritmasını yazınız
b) Programın C# kodlarını yazınız.

23.Problem 3: Bir lise öğrencisi bir kimya problemi ile ilgili bir yazılım oluşturmak istemektedir. Probleme göre; 4,5 kg (4500 gram) ağırlığındaki buz kütlesi 30 saniyede 5 gram erimektedir. Buna göre 1 saat sonunda buz kütlesinden kaç gram kaldığını hesaplayıp ekrana yazan programı tasarlamasına yardımcı olunuz.

Yukarıdaki kuralları dikkate aldığınızda:

a)Programın algoritmasını yazınız
b) Programın C# kodlarını yazınız.

24. Problem 4: Bir futbol antrenörü futbolcuların performanslarına göre sahaya süreceği ilk 11' i oluşturmaktadır. Performans puanı en büyük ilk 11 kişi maçta ilk 11 de yer almaktadır. Buna göre bir yazılımcı olarak sizden; 18 kişilik takımın performansının klavyeden sırayla girilerek, futbolcuların performans puanını büyükten küçüğe sıralayan ve ilk 11 kişinin performans puanlarını ekrana sırasıyla yazan programı tasarlamanız istenmektedir.

Yukarıdaki kuralları dikkate aldığınızda:

a)Programın algoritmasını yazınız
b) Programın C# kodlarını yazınız.

EK E- ODAK GRUP GÖRÜŞME FORMU

ODAK GRUP GÖRÜŞME SORULARI

Araştırmacı: Osman EROL
Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi
Bilgisayar ve Öğretim Teknolojileri Eğitimi

Görüşme Başlama Saati :
Görüşme Bitiş Saati :
Görüşme Tarihi :
Görüşme Yapılan Yer :

Merhabalar,

Öncelikle görüşme yapmayı kabul ettiğiniz için teşekkür ederim. Görüşme sorularına geçmeden önce adınızı, soyadınızı paylaşarak kendinizi tanıtır mısınız?

Görüşme Soruları

1. Genel anlamda ders hakkındaki (Temel Programlama yapıları/ C#) düşünceleriniz nelerdir?
    - Beğendiğiniz yönler nelerdir? Lütfen açıklayınız.
    - Beğenmediğiniz yönler nelerdir? Lütfen açıklayınız.
2. Ders süresince gerçekleştirilen (Temel Programlama yapıları/ C#) öğrenme / öğretme etkinliklerini ilişkin görüşleriniz nelerdir?
3. Ders süresince (Temel Programlama yapıları/ C#) sizi zorlayan / güdüleyen (motive eden) durumlar nelerdi? (Anlamakta zorlandığınız/ en iyi anladığınız noktalar nelerdi?)
    - Neden? Lütfen açıklayınız.
4. Bu ders süresince (Temel Programlama yapıları/ C#) öğrendiklerinizi nasıl kullanmayı düşünüyorsunuz?
5. Ders süreci sonunda programlama ile ilgili düşüncelerinizde nasıl bir değişiklik oldu?
    - Neden? Lütfen açıklayınız.
6. İleride programlama ile ilgili bir işte çalışmayı düşünüyor musunuz?
    - Programlamaya devam etmeyi düşünüyor musunuz? Neden? Lütfen açıklayınız.
    - Bu kararı almada ders sürecinin nasıl bir etkisi oldu? Neden? Lütfen açıklayınız.

Teşekkür ederim.

EK F- ODAK GRUP GÖRÜŞME KATILIMCI BLGİLENDİRME VE İZİN FORMU

“SCRATCH İle Programlama Öğretiminin
Başarı Ve Motivasyona Etkisi “ Tez Çalışması
Katılımcı Bilgilendirme Formu

Değerli katılımcı,
Hazırlanan bu metnin amacı, sizi, katılmakta olduğunuz yarı yapılandırılmış görüşmenin amacı hakkında ve katılımınız durumunda sizden istenen izinler hakkında bilgilendirmektir.

SCRATCH İle Programlama Öğretiminin Başarı Ve Motivasyona Etkisi adlı tez çalışması Anadolu Üniversitesi Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Eğitimi (BÖTE) Bölümü öğretim üyesi Doç.Dr. Adile Aşkım KURT ve Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Eğitimi (BÖTE) Bölümü öğretim elemanı Öğr.Gör. Osman EROL tarafından yürütülmektedir. Bu çalışmanın amacı SCRATCH ile programlama öğretiminin öğrencilerin motivasyon ve programlama başarılarına etkisini ölçmektir. Katılmakta olduğunuz yarı yapılandırılmış görüşmenin ses kaydı altına alınacağı konusunda sizi bilgilendirmek isteriz. Araştırmacılar, görüşme boyunca alınacak kayıtların sadece bilgilendirilmiş olduğunuz çalışma kapsamında bilimsel amaçlarla kullanılacağını ve çalışma ekibi dışında üçüncü kişilerle bu bilgilerin paylaşılmayacağını temin etmektedirler. Ayrıca araştırmadan çıkarılacak bilimsel ürünler ve raporların hiçbirinde kimliğinizi işaret edecek bilgilerin yer almayacağı da araştırmacılar tarafından garanti edilmektedir. Yapılacak olan kayıtların doğası gereği kimliğinizi belirtecek verilerin alınması olasıdır. Bu nedenle bu çalışmaya katılım isteğinize bağlıdır. Katılımcıların görüşmeye doğrudan katılmama ya da istedikleri noktada terk etme özgürlükleri saklı tutulmaktadır.

Saygılarımızla,
Doç.Dr. Adile Aşkım KURT
Öğr. Gör. Osman EROL

“SCRATCH İle Programlama Öğretiminin
Başarı Ve Motivasyona Etkisi “ Tez Çalışması
Katılımcı İzin Formu

SCRATCH İle Programlama Öğretiminin Başarı Ve Motivasyona Etkisi adlı tez çalışması çerçevesinde düzenlenen bu yarı yapılandırılmış görüşmeye kendi isteğim ile katıldığımı bildiririm. Bu görüşmenin ses kayıt cihazı ile kayıt altına alınmasına ve toplanan verinin sadece bilimsel amaçlar için kullanılmasına izin veriyorum.

İmza Tarih

Adı Soyadı :
Telefon :
e-Posta :
Adres :

EK G- DENEY GRUBU DERS İÇERİĞİ

TEMEL PROGRAMLAMA YAPILARI VE SCRATCH

HAFTA 1

Kazanımlar

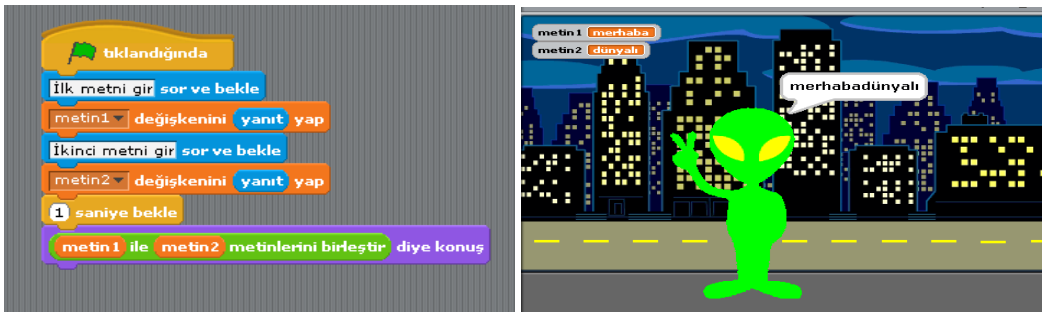
- Temel programlama yapıları ile ilgili olarak değişken tanımlayabilme,
- Temel programlama yapıları ile ilgili olarak değişkenlere değer atama,
- Temel programlama yapıları ile ilgili olarak girdi ve çıktı eylemlerini gerçekleştirebilme

İçerik

1. **Temel Kavramlar**
   - 1.1. Veri - Data- Bilgi
   - 1.2. Bilgisayar Çalışma Mantığı
   - 1.3. İkili sayı sistemi (Binary Digit System)
2. **Problem Çözme ve Algoritma**
   - 2.1. Sözde Kod yazımı
   - 2.2. Örnek hayat problemleri ile sözde kod yazma
3. **Scratch**
   - 3.1. Temel Kavramlar
   - 3.2. Scratch Arayüzü
     - 3.2.1. Kodlama Paneli
     - 3.2.2. Sahne ve Karakterler
     - 3.2.3. Araç Çubuğu
   - 3.3. Kontrol Komutları 1
     - 3.3.1. Tıklandığında
     - 3.3.2. Tuşuna basıldığında
     - 3.3.3. Karakter tıklandığında
     - 3.3.4. sn bekle
4. **Değişken Kavramı**
   - 4.1. Scratch Değişken Kavramı
     - 4.1.1. Değişken Oluşturma
     - 4.1.2. Değer Atama
     - 4.1.3. Değişken Değerini Değiştirme (Artırma/azaltma)

Etkinlikler

1. Çay pişirme işleminin basamaklarını yazınız.
2. Bir kavşakta trafik ışığı ile karşılaşan sürücünün ışığın durumuna göre (kırmızı, sarı, yeşil) hareketini basamaklar halinde yazınız.
3. Klavyeden girilen iki farklı metinsel ifadeyi birlikte ekrana yazan programı Scratch ile hazırlayınız. (Merhaba Dünyalı).



4. Adınız sorusunu klavyeden cevaplayarak "Adım …….dır" şeklinde ekrana yazdıran programı Scratch ile tasarlayınız.

Serbest Tasarım Etkinlikleri

1. Tasarım hakkında bilgiler
   1.1. Oyun tasarımı hakkında bilgilendirme
   1.2. Oyun senaryosu (Oyun Hikayesi) ve oyun nesneleri tasarımı hakkında bilgilendirme

HAFTA 2 (Değişken Kavramı)

Kazanımlar

- Temel programlama yapıları ile ilgili olarak değişken tanımlayabilme,
- Temel programlama yapıları ile ilgili olarak değişkenlere değer atama,
- Temel programlama yapıları ile ilgili olarak girdi ve çıktı eylemlerini gerçekleştirebilme

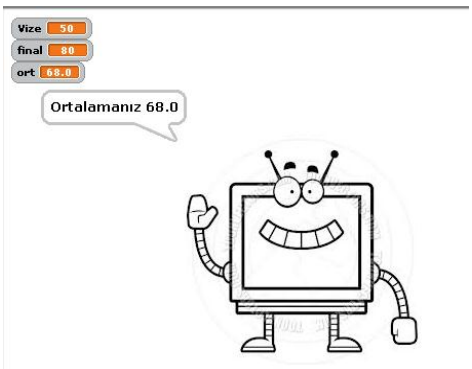
İçerik

1. Operatörler 1
   1.1.1. Aritmetik İşlemler
   1.1.2. rastgele sayı üret
   1.1.3. mod
   1.1.4. yuvarla
   1.1.5. karakök vb.
2. Scratch- Görünüm Komutları
   2.1. diye konuş / sn boyunca diye konuş
   2.2. kostümüne geç / sonraki kostüm/ şuanda görünen kostüm
   2.3. diye düşün / sn boyunca diye düşün
   2.4. Efektini yap/ efektini değiştir/ Grafik efektlerini temizle
   2.5. boyu değiştir / boyu yap/ ebat
   2.6. göster/ gizle/ üste çık/ 1 katman alta geç
3. Scratch- Algılama komutları 1
   3.1. Sor ve bekle
   3.2. yanıt
   3.3. farenin x koordinatı/ farenin y koordinatı
   3.4. ile arasındaki mesafe
4. Scratch- Hareket Komutları
   4.1. adım git
   4.2. derece dön
   4.3. yönüne dön
   4.4. doğru dön
   4.5. x: y: konumuna git
   4.6. ile aynı konuma git
   4.7. saniyede x: y: konumuna git
   4.8. x'i değiştir
   4.9. x' i yap
   4.10. y' yi değiştir
   4.11. y' yi yap
   4.12. kenara geldiğinde geri dön
   4.13. x koordinatı/ y koordinatı / yön

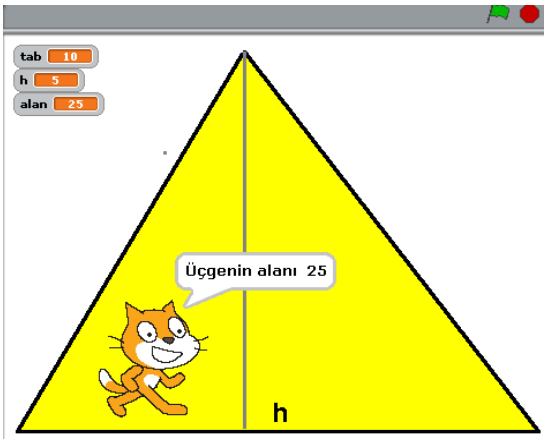
Etkinlikler

1. Klavyeden girilen 3 sayının toplamını bulup ekrana yazdıran programı Scratch ile hazırlayınız.

2. Klavyeden girilen vize notunun %40' ını ve final notunun % 60' ını alarak toplam notu hesaplayarak ekrana yazan programı Scratch ile hazırlayınız.



3. Kenar değerleri (taban/ yükseklik) verilen üçgenin alanını hesaplayan programı Scratch ile hazırlayınız.



Diğer Etkinlikler

- diye konuş / sn boyunca diye konuş
- kostümüne geç / sonraki kostüm/ şuanda görünen kostüm
- diye düşün / sn boyunca diye düşün
- Efektini yap/ efektini değiştir/ Grafik efektlerini temizle
- boyu değiştir / boyu yap/ ebat

- farenin x koordinatı/ farenin y koordinatı
- adım git
- derece dön
- yönüne dön
- doğru dön
- x: y: konumuna git
- ile aynı konuma git
- saniyede x: y: konumuna git
- x'i değiştir
- x' i yap
- y' yi değiştir
- y' yi yap
- kenara geldiğinde geri dön
- x koordinatı/ y koordinatı / yön

Serbest Tasarım Etkinlikleri

1. Oyun senaryolarının hazırlanması

HAFTA 3 (Karar Verme ve Kontrol İşlemleri- Eğer)

Kazanımlar

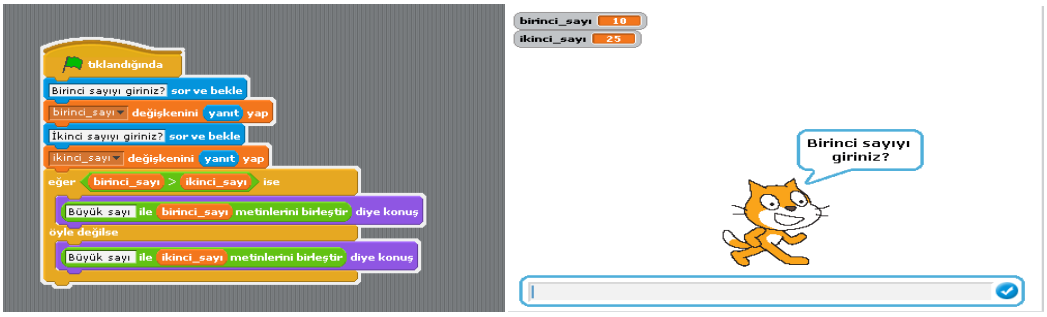
- Temel programlama yapıları ile ilgili olarak karar verme ve kontrol işlemlerini gerçekleştirebilme.

İçerik

1. Karar verme ve kontrol işlemleri
2. Scratch - Operatörler 2
   2.1. Mantıksal İşlemler
   2.2. metinlerini birleştir
   2.3. metninin sıradaki harfi
   2.4. metninin uzunluğu
3. Scratch- Kontrol Komutları 2
   3.1. eğer ise
   3.2. eğer ise öyle değilse
4. Scratch- Algılama Komutları 2
   4.1. değiyor mu?
   4.2. rengine değiyor mu?
   4.3. renk renge değiyor mu?
   4.4. sayaç
   4.5. sayacı sıfırla
5. İç içe karar verme
   5.1. Etkinlikler
6. Sayaç Mantığı

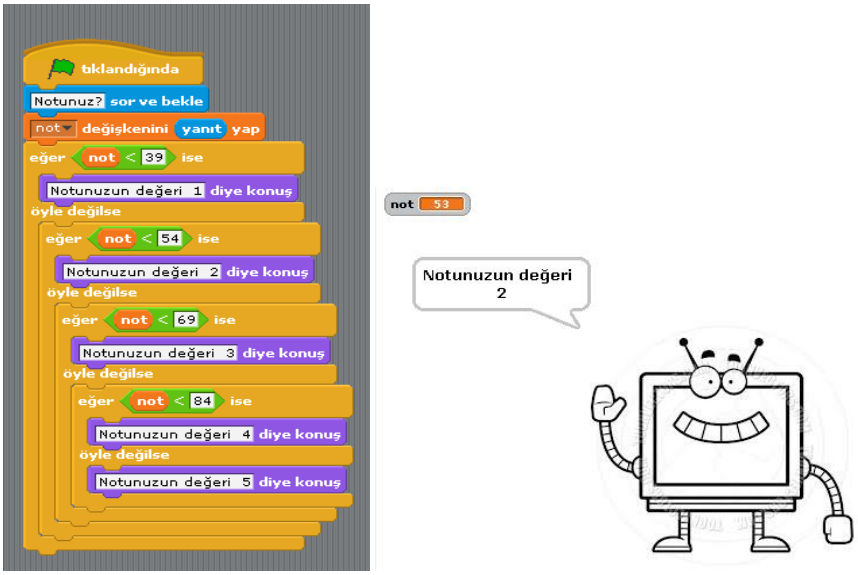
Etkinlikler

1. Ekrandan girilen iki sayıyı büyüklük ve küçüklük durumuna göre karşılaştıran ve ekrana "Büyük sayı = ..." şeklinde yazan programı Scratch ile tasarlayınız.

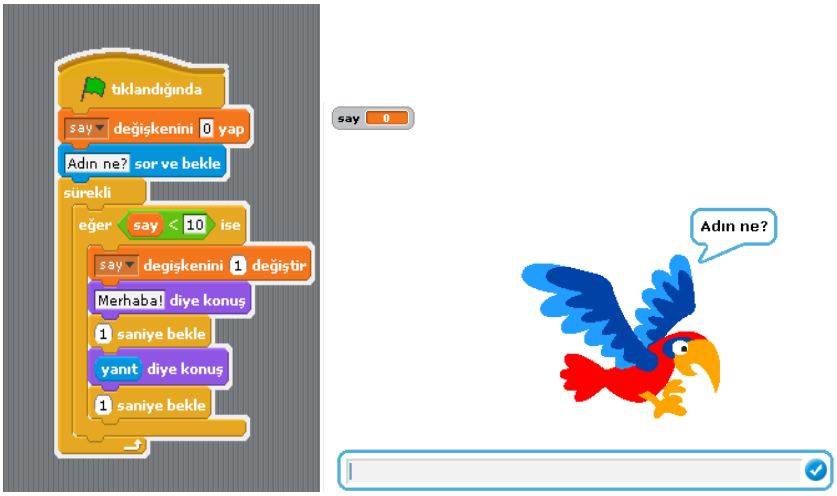


2. Klavyeden girilen bir sayının negatif mi, pozitif mi yoksa "sıfır" mı olduğunu test eden ve sonucu ekrana yazan programı Scratch ile tasarlayınız.

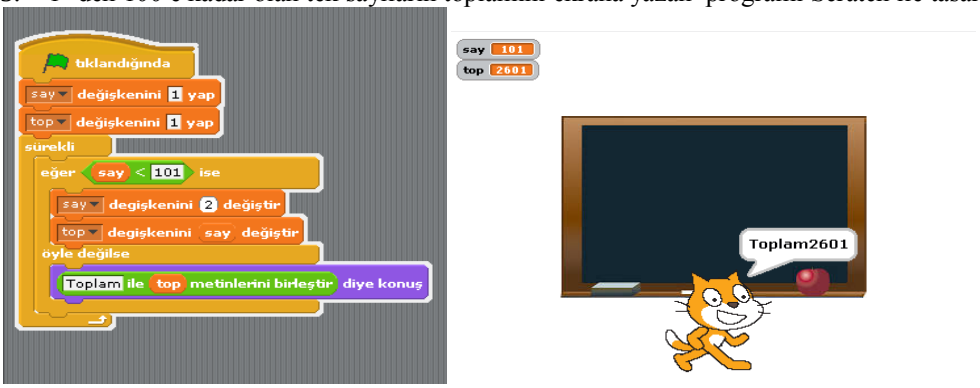
3. Klavyeden girilen 100 lük sistemdeki bir notun 5' lik sistemdeki karşılığını ekrana yazan programı Scratch ile tasarlayınız.
   * 0- 39 arası 1
   40 - 54 arası 2
   55- 69 arası 3
   70- 84 arası 4
   85- 100 arası 5



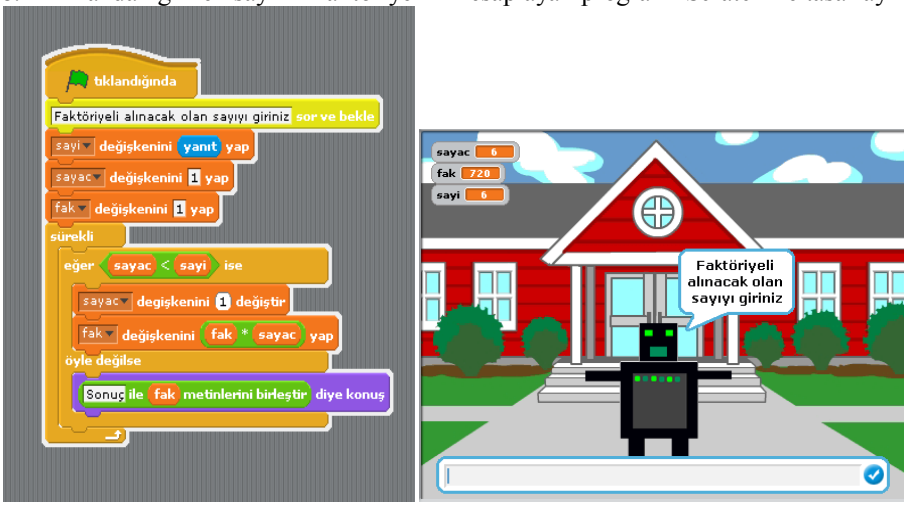
4. Klavyeden girilen bir kelimeyi ekrana 10 kez ekrana yazan programı Scratch ile tasarlayınız. (Sayaç Mantığı)



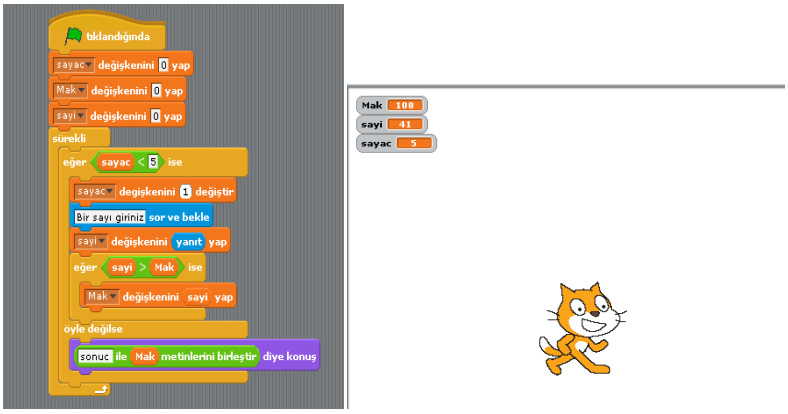
5. 1' den 100 e kadar olan tek sayıların toplamını ekrana yazan programı Scratch ile tasarlayınız.



6. Ekrandan girilen sayının faktöriyelini hesaplayan programı Scratch ile tasarlayınız.



7. Klavyeden girilen 5 sayının en büyüğünü bulan programı Scratch ile tasarlayınız.



8. Tahmin etme oyunu: 1 ile 100 arasında rastgele üretilen bir sayıyı, klavyeden girilen sayıya göre tahmin eden ve "Yukarı" yada "Aşağı" şeklinde yönlendiren tahmin oyununu Scratch ile tasarlayınız.

9. Diğer Etkinlikler
   - değiyor mu?
   - rengine değiyor mu?
   - renk renge değiyor mu?
   - sayaç
   - sayacı sıfırla

   komutları ile ilgili uygulama Etkinlikleri

Serbest Tasarım Etkinlikleri

1. Oyun senaryolarının hazırlanması

HAFTA 4. Döngüler
Kazanımlar

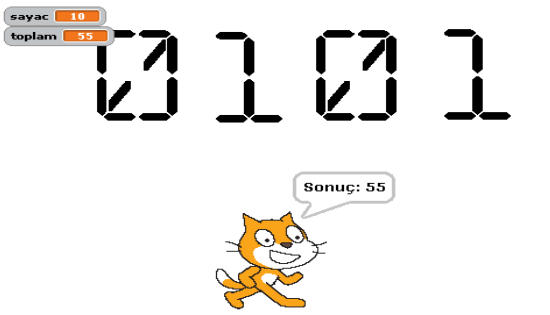
- Temel programlama yapıları ile ilgili olarak döngü işlemlerini gerçekleştirebilme.

İçerik

1. Döngü kavramı
2. Sayaç Mantığı
3. Scratch - Kontrol komutları 3
   - 3.1. sürekli
   - 3.2. kez tekrarla
   - 3.3. eğer ise sürekli
   - 3.4. olana kadar bekle
   - 3.5. olana kadar tekrarla
4. Scratch- Kalem
   - 4.1. Kalemle çizilenleri temizle
   - 4.2. Kalemi bastır
   - 4.3. Kalemi kaldır
   - 4.4. Kalem rengini yap
   - 4.5. Kalem rengini değiştir
   - 4.6. Kalem gölgesini değiştir
   - 4.7. Kalem gölgesini yap
   - 4.8. Kalem boyutunu değiştir
   - 4.9. Kalem boyutunu yap
   - 4.10. damgala

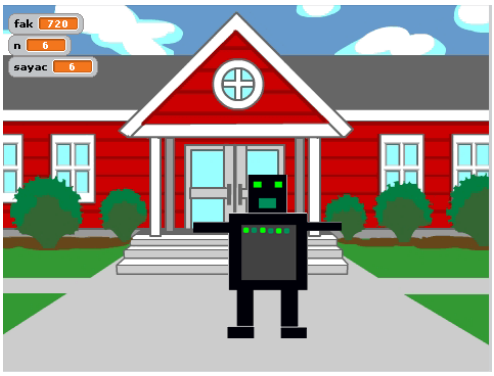
Etkinlikler

1. 1' den 10 a kadar sayıların toplamını ekrana yazan programı tasarlayınız (Döngü ile).

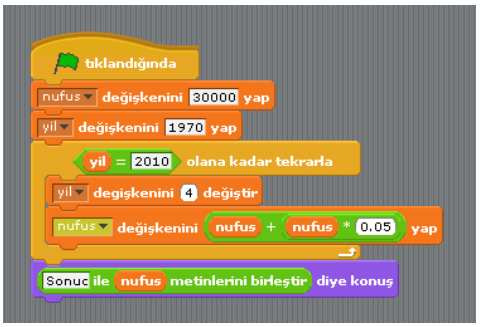


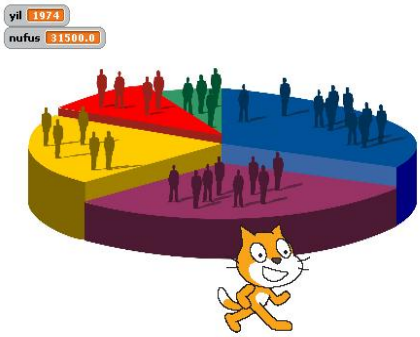
2. Bir iş yerinde çalışan ve klavyeden girilen 10 kişinin maaşının ortalamasını hesaplayan programı Scratch ile tasarlayınız.

3. Klavyeden girilen bir sayının faktöriyelini hesaplayan programı tasarlayınız.(Döngü ile)

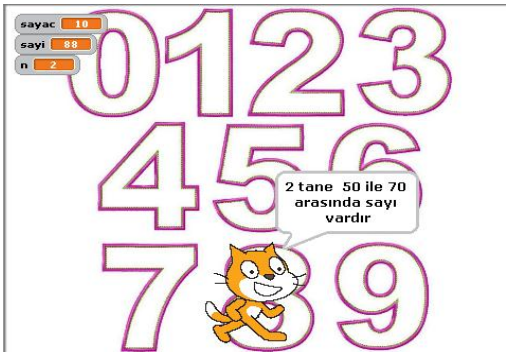


4. Bir ilin 1970 yılında nüfusu 30000 dir. Her 4 yılda nüfus artışı %5 ise bu ilin 2010 daki nüfusunu bulan programı Scratch ile tasarlayınız.





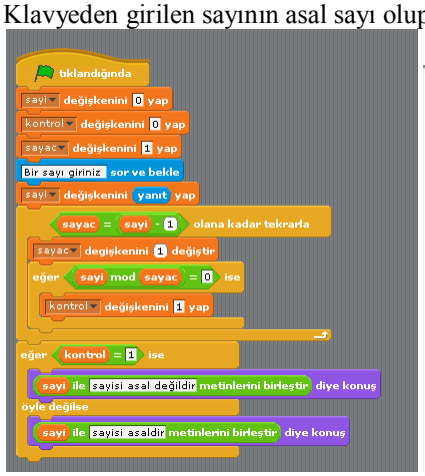
5. Ekrandan girilen 10 tane sayı içinde 50 ile 70 arasında kaç tane sayı oluğunu bulan ve bu sayıların toplamını ve aritmetik ortalamasını bularak ekrana yazan programı Scratch ile tasarlayınız.

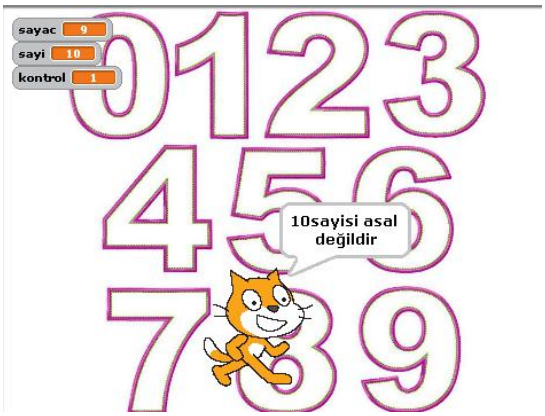


6. Bir kömür kuyusu 960 m derinliğindedir. En dipte yer alan vagona 2586 kg kömür yüklenmektedir. Kepçe yukarı çekilirken her 40 metrede 58 kg kömür dökülmektedir. Kepçe yüzeye çıkınca ne kadar kömür kalacağını hesaplayan programı Scratch ile tasarlayınız.

7. Bir kuruluşta çalışan 10 kişiye ikramiye verilecektir. Hizmet yılı 5 ve üzeri olanlara 500 TL, hizmet yılı 5 yılın altında olanlara 250 tl ikramiye verilecektir. Personele ödenecek toplam ikramiye bedelini bulan ve ekrana yazan programı tasarlayınız.

8. Klavyeden girilen sayının asal sayı olup olmadığını test eden programı Scratch ile tasarlayınız.





9. Diğer Etkinlikler
   - sürekli
   - kez tekrarla
   - eğer ise sürekli
   - olana kadar bekle
   - olana kadar tekrarla
   - Kalemle çizilenleri temizle
   - Kalemi bastır
   - Kalemi kaldır
   - Kalem rengini yap
   - Kalem rengini değiştir
   - Kalem gölgesini değiştir

- Kalem gölgesini yap
- Kalem boyutunu değiştir
- Kalem boyutunu yap
- damgala

Serbest Tasarım Etkinlikleri

1. Oyun nesnelerinin tasarımı
    1.1. Karakterlerin tasarlanması
    1.2. Oyun sahnelerinin tasarlanması

Hafta 5. İç içe Döngüler
Kazanımlar

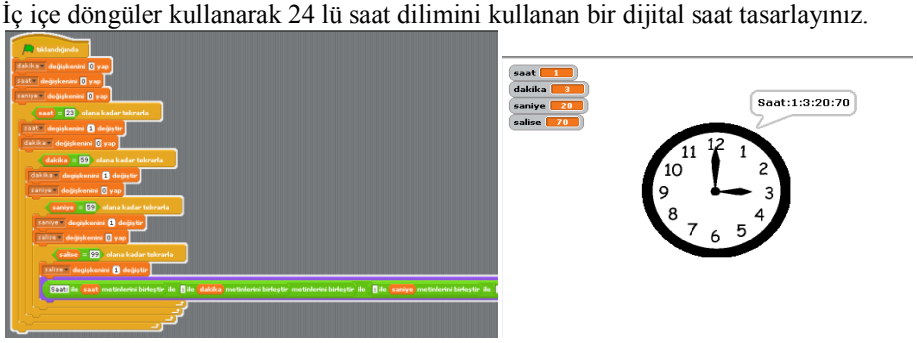
- Temel programlama yapıları ile ilgili olarak döngü işlemlerini gerçekleştirebilme.

İçerik

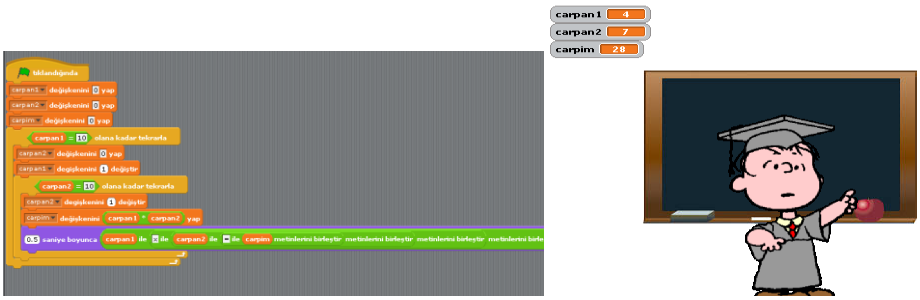
1. İç içe döngüler
2. Etkinlikler
3. Scratch - Ses Komutları
   - 3.1. sesini çal
   - 3.2. sesini bitene kadar çal
   - 3.3. tüm sesleri durdur
   - 3.4. sesini vuruş çal
   - 3.5. vuruşunu bekle
   - 3.6. notasını vuruş çal
   - 3.7. enstrümanı yap
   - 3.8. sesi değiştir
   - 3.9. ses yüksekliğini % yap
   - 3.10. ses seviyesi
   - 3.11. tempoyu değiştir
   - 3.12. tempo

Etkinlikler

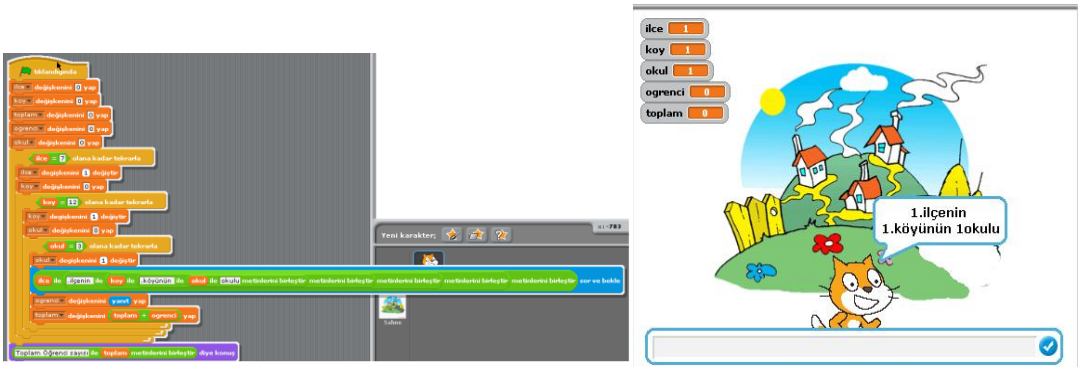
1. İç içe döngüler kullanarak 24 lü saat dilimini kullanan bir dijital saat tasarlayınız.



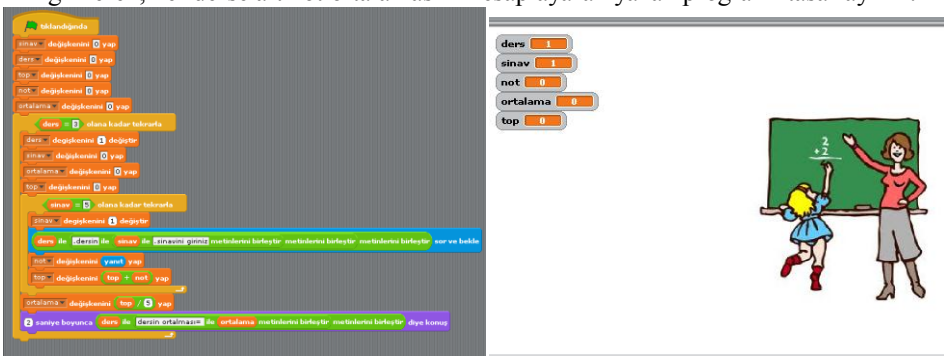
2. 10' lara kadar çarpım tablosunu hesaplayarak ekrana yazdıran programı tasarlayınız.



3. Bir ilde 7 tane ilçe, her ilçede 12 tane köy, her köyde 3 tane okul bulunmaktadır. Klavyede her okulda bulunan öğrenci sayısı sırası ile girilmektedir. Tüm ilçe, köy, ve okuldaki toplam öğrenci sayılarını bulan programı Scratch ile tasarlayınız.



4. Bir öğrencinin Matematik, Fen ve Sosyal bilgiler derslerinden 5' er yazılı sınav notunu klavyeden girilerek, her derse ait not ortalamasını hesaplayarak yazan programı tasarlayınız.



5. Diğer Etkinlikler
   - sesini çal
   - sesini bitene kadar çal
   - tüm sesleri durdur
   - sesini vuruş çal
   - vuruşunu bekle
   - notasını vuruş çal
   - enstrümanı yap
   - sesi değiştir
   - ses yüksekliğini % yap
   - ses seviyesi
   - tempoyu değiştir
   - tempo

komutları ile ilgili uygulama Etkinlikleri

Serbest Tasarım Etkinlikleri

1. Oyun nesnelerinin tasarımı
   1.1. Karakterlerin tasarlanması
   1.2. Oyun sahnelerinin tasarlanması

HAFTA 6. Diziler

Kazanımlar

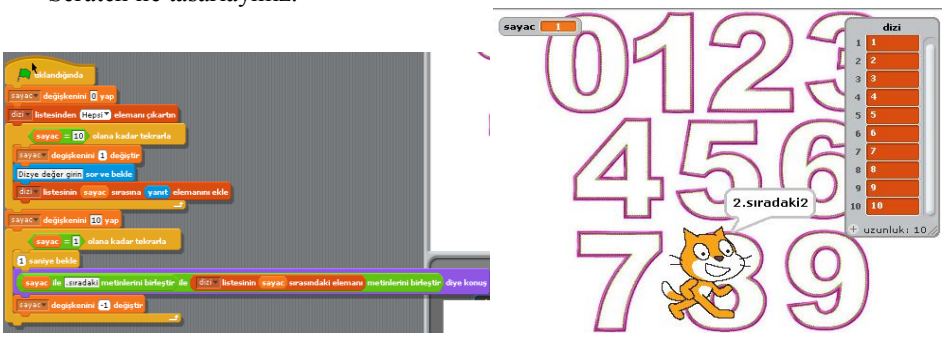
- Temel programlama yapıları ile ilgili olarak dizi işlemlerini gerçekleştirebilme.

İçerik

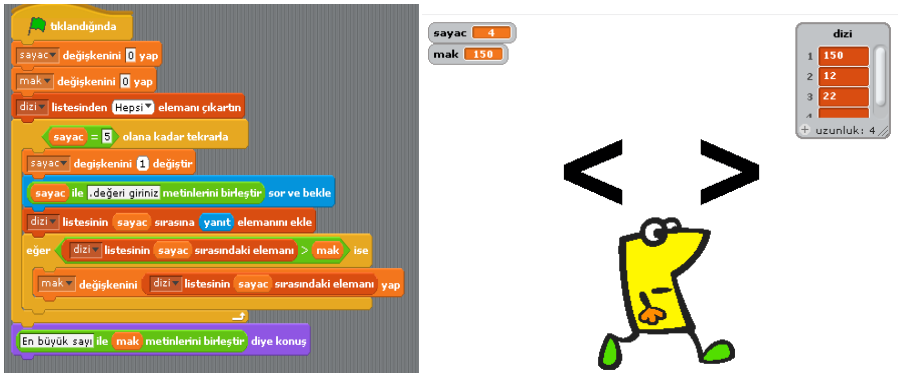
1. Diziler
2. Scratch - Listeler
   - 2.1. listesine nesne ekle
   - 2.2. listesinden elemanı çıkartın
   - 2.3. listesinin sırasına nesne ekle
   - 2.4. listesinin sıradaki elamanını ile değiştir
   - 2.5. listesinin sıradaki elemanı
   - 2.6. listenin eleman sayısı
   - 2.7. listesinin bir elemanıdır

Etkinlikler

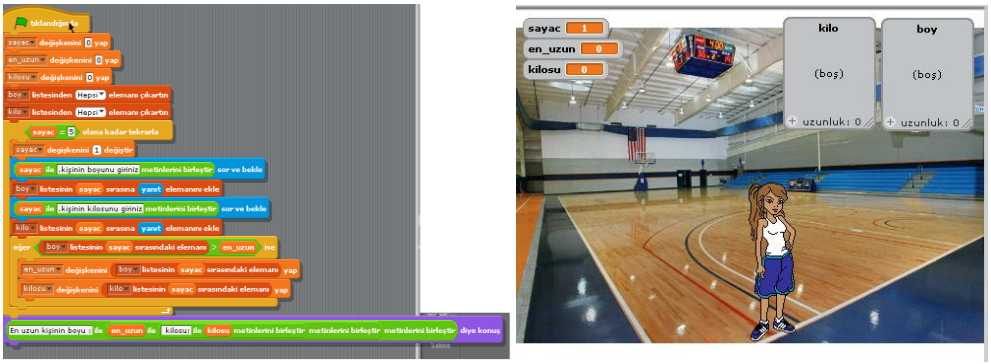
1. 10 elemanlı bir diziye girilecek olan değerleri giriş sırasına ters olarak ekrana yazdıran programı Scratch ile tasarlayınız.



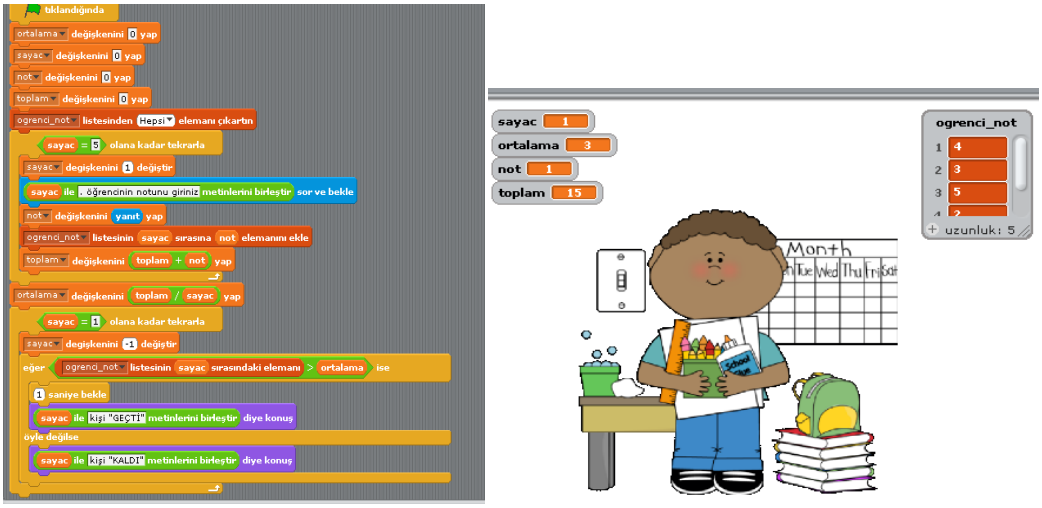
2. Klavyeden değerleri girilen 5 elemanlı bir dizinin en büyük değerini bulan ve ekrana yazan programı Scratch ile tasarlayınız.



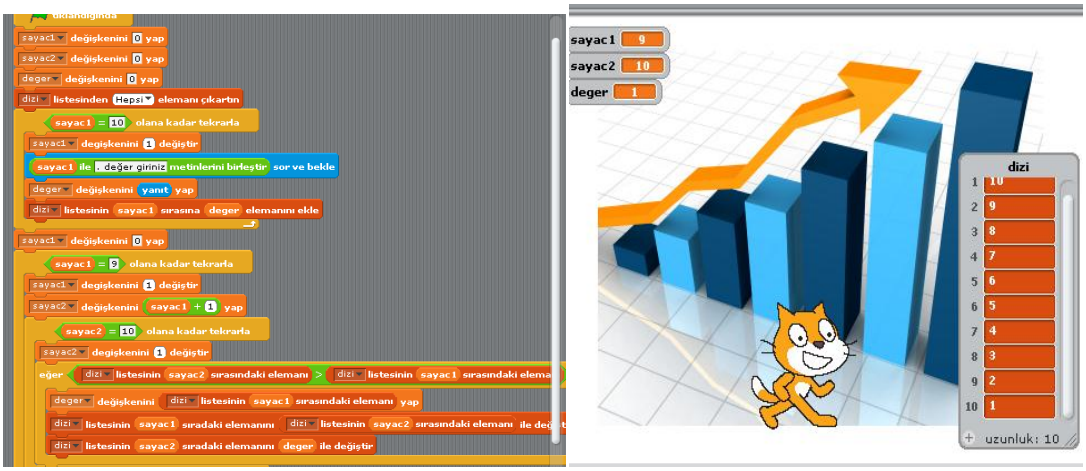
3. Okul basketbol takımındaki 5 kişinin sırasıyla önce boyları, sonra da kiloları klavyeden girilecektir. En uzun boylu oyuncunun hem boyunu hem de kilosunu ekrana yazan programı Scratch ile tasarlayınız.



4. Bir öğretmen, başarıyı ölçmek için çan eğrisi uygulamaktadır. 20 öğrencinin bulunduğu bir sınıfın notları sırasıyla girilmektedir. Buna göre notlar girildikten sonra her bir öğrencinin çan eğrisine göre (ortalamaya göre) geçip geçmediğini (Geçti/Kaldı) ekrana yazan programı Scratch ile tasarlayınız.



5. Klavyeden rastgele girilecek 10 değerli bir dizinin elemanlarını küçükten büyüğe doğru sıralayarak ekrana yazdıran programı yapınız



Serbest Tasarım Etkinlikleri

1. Oyun tasarımının sonlandırılması

EK H- KONTROL GRUBU DERS İÇERİĞİ

TEMEL PROGRAMLAMA YAPILARI VE AKIŞ DİYAGRAMLARI

HAFTA 1

Kazanımlar

- Temel programlama yapıları ile ilgili olarak değişken tanımlayabilme
- Temel programlama yapıları ile ilgili olarak değişkenlere değer atama
- Temel programlama yapıları ile ilgili olarak girdi ve çıktı eylemlerini gerçekleştirebilme

İçerik

1. **Temel Kavramlar**
   - 1.1. Veri - Data- Bilgi
   - 1.2. Bilgisayar Çalışma Mantığı
   - 1.3. İkili sayı sistemi (BinaryDigitSystem)
2. **Problem Çözme ve Algoritma**
   - 2.1. Sözde Kod yazımı
   - 2.2. Örnek hayat problemleri ile sözde kod yazma
3. **Akış Diyagramları**
   - 3.1. Semboller ve Anlamları
   - 3.2. Başla ve Dur
4. **Değişken Kavramı**
   - 4.1.1. Değişken Oluşturma
   - 4.1.2. Değer Atama
   - 4.1.3. Değişken Değerini Değiştirme (Artırma/azaltma)

Etkinlikler

5. Çay pişirme işleminin basamaklarını yazınız.
6. Bir kavşakta trafik ışığı ile karşılaşan sürücünün ışığın durumuna göre (kırmızı, sarı, yeşil) hareketini basamaklar halinde yazınız.
7. Klavyeden girilen iki farklı metinsel ifadeyi birlikte ekrana yazan programı akış diyagramları kullanarak hazırlayınız. (Merhaba Dünyalı).
8. Adınız sorusunu klavyeden cevaplayarak "Adım …….dır" şeklinde ekrana yazdıran programı akış diyagramları kullanarak tasarlayınız.

HAFTA 2 (Değişken Kavramı)

Kazanımlar

- Temel programlama yapıları ile ilgili olarak değişken tanımlayabilme,
- Temel programlama yapıları ile ilgili olarak değişkenlere değer atama,
- Temel programlama yapıları ile ilgili olarak girdi ve çıktı eylemlerini gerçekleştirebilme

İçerik

5. Operatörler 1
6. Aritmetik İşlemler
7. Yaz Komutu
8. Oku Komutu

Etkinlikler

1. Klavyeden girilen 3 sayının toplamını bulup ekrana yazdıran programı akış diyagramları kullanarak hazırlayınız.
2. Klavyeden girilen vize notunun %40' ını ve final notunun % 60' ını alarak toplam notu hesaplayarak ekrana yazan programı akış diyagramları kullanarak hazırlayınız.
3. Kenar değerleri (taban/ yükseklik) verilen üçgenin alanını hesaplayan programı akış diyagramları kullanarak hazırlayınız.

HAFTA 3 (Karar Verme ve Kontrol İşlemleri- Eğer)

Kazanımlar

- Temel programlama yapıları ile ilgili olarak karar verme ve kontrol işlemlerini gerçekleştirebilme.

İçerik

7. Akış Diyagramları -Karar verme ve kontrol işlemleri
   7.1. Mantıksal İşlemler - Operatörler
8. Akış Diyagramları -Koşul İfadeleri
   8.1. eğer
   8.2. eğer değilse
9. Akış Diyagramları -İç içe eğer
10. Sayaç Mantığı

Etkinlikler

1. Ekrandan girilen iki sayıyı büyüklük ve küçüklük durumuna göre karşılaştıran ve ekrana "Büyük sayı = ..." şeklinde yazan programı akış diyagramları kullanarak tasarlayınız.
2. Klavyeden girilen bir sayının negatif mi, pozitif mi yoksa "sıfır" mı olduğunu test eden ve sonucu ekrana yazan programı akış diyagramları kullanarak tasarlayınız.
3. Klavyeden girilen 100 lük sistemdeki bir notun 5' lik sistemdeki karşılığını ekrana yazan programı akış diyagramları kullanarak tasarlayınız.
   * 0- 39 arası 1
   40 - 54 arası 2
   55- 69 arası 3
   70- 84 arası 4
   85- 100 arası 5
4. Klavyeden girilen bir kelimeyi ekrana 10 kez ekrana yazan programı akış diyagramları kullanarak tasarlayınız. (Sayaç Mantığı)
5. 1’ den 100 e kadar olan tek sayıların toplamını ekrana yazan programıakış diyagramları kullanarak tasarlayınız.
6. Ekrandan girilen sayının faktöriyelini hesaplayan programı akış diyagramları kullanarak tasarlayınız.
7. Klavyeden girilen 5 sayının en büyüğünü bulan programı akış diyagramları kullanarak tasarlayınız.
8. Tahmin etme oyunu: 1 ile 100 arasında rastgele üretilen bir sayıyı, klavyeden girilen sayıya göre tahmin eden ve "Yukarı" yada "Aşağı" şeklinde yönlendiren tahmin oyununu akış diyagramları kullanarak tasarlayınız.

HAFTA 4. Döngüler

Kazanımlar

- Temel programlama yapıları ile ilgili olarak döngü işlemlerini gerçekleştirebilme.

İçerik

5. Akış Diyagramları -Döngü kavramı
6. Akış Diyagramları -Sayaç Mantığı

Etkinlikler

1. 1' den 10 a kadar sayıların toplamını ekrana yazan programı akış diyagramları kullanarak tasarlayınız (Döngü ile).
2. Bir iş yerinde çalışan ve klavyeden girilen 10 kişinin maaşının ortalamasını hesaplayan programı akış diyagramları kullanarak tasarlayınız.
3. Klavyeden girilen bir sayının faktöriyelini hesaplayan programı akış diyagramları kullanarak tasarlayınız.(Döngü ile)
4. Bir ilin 1970 yılında nüfusu 30000 dir. Her 4 yılda nüfus artışı %5 ise bu ilin 2010 daki nüfusunu bulan programı akış diyagramları kullanarak tasarlayınız.
5. Ekrandan girilen 10 tane sayı içinde 50 ile 70 arasında kaç tane sayı oluğunu bulan ve bu sayıların toplamını ve aritmetik ortalamasını bularak ekrana yazan programı akış diyagramları kullanarak tasarlayınız.
6. Bir kömür kuyusu 960 m derinliğindedir. En dipte yer alan vagona 2586 kg kömür yüklenmektedir. Kepçe yukarı çekilirken her 40 metrede 58 kg kömür dökülmektedir. Kepçe yüzeye çıkınca ne kadar kömür kalacağını hesaplayan programı akış diyagramları kullanarak tasarlayınız.
7. Bir kuruluşta çalışan 10 kişiye ikramiye verilecektir. Hizmet yılı 5 ve üzeri olanlara 500 TL, hizmet yılı 5 yılın altında olanlara 250 tl ikramiye verilecektir. Personele ödenecek toplam ikramiye bedelini bulan ve ekrana yazan programıakış diyagramları kullanarak tasarlayınız.
8. Klavyeden girilen sayının asal sayı olup olmadığını test eden programı akış diyagramları kullanarak tasarlayınız.

Hafta 5. İç içe Döngüler
Kazanımlar

- Temel programlama yapıları ile ilgili olarak döngü işlemlerini gerçekleştirebilme.

İçerik

4. Akış Diyagramları -İç içe döngüler
   4.1. İç içe Döngü problemleri

Etkinlikler

1. İç içe döngüler kullanarak 24 lü saat dilimini kullanan bir dijital saat tasarlayınız.
2. 10' lara kadar çarpım tablosunu hesaplayarak ekrana yazdıran programı akış diyagramları kullanarak tasarlayınız (Eker, 2011).
3. Bir ilde 7 tane ilçe, her ilçede 12 tane köy, her köyde 3 tane okul bulunmaktadır. Klavyede her okulda bulunan öğrenci sayısı sırası ile girilmektedir. Tüm ilçe, köy, ve okuldaki toplam öğrenci sayılarını bulan programı akış diyagramları kullanarak tasarlayınız.
4. Bir öğrencinin Matematik, Fen, İngilizce ve Sosyal bilgiler derslerinden 5' er yazılı sınav notunu klavyeden girilerek, her derse ait not ortalamasını hesaplayarak yazan programı akış diyagramları kullanarak tasarlayınız.

HAFTA 6. Diziler

Kazanımlar

- Temel programlama yapıları ile ilgili olarak dizi işlemlerini gerçekleştirebilme.

İçerik

3. Akış Diyagramları -Diziler
   3.1. Dizi - Değişken farkları

Etkinlikler

1. 10 elemanlı bir diziye girilecek olan değerleri giriş sırasına ters olarak ekrana yazdıran programı akış diyagramları kullanarak tasarlayınız.
2. Klavyeden değerleri girilen 5 elemanlı bir dizinin en büyük değerini bulan ve ekrana yazan programı akış diyagramları kullanarak ile tasarlayınız
3. Okul basketbol takımındaki 5 kişinin sırasıyla önce boyları, sonra da kiloları klavyeden girilecektir. En uzun boylu oyuncunun hem boyunu hem de kilosunu ekrana yazan programı akış diyagramları kullanarak tasarlayınız.
4. Bir öğretmen, başarıyı ölçmek için çan eğrisi uygulamaktadır. 20 öğrencinin bulunduğu bir sınıfın notları sırasıyla girilmektedir. Buna göre notlar girildikten sonra her bir öğrencinin çan eğrisine göre (ortalamaya göre) geçip geçmediğini (Geçti/Kaldı) ekrana yazan programı akış diyagramları kullanarak tasarlayınız.
5. Klavyeden rastgele girilecek 10 değerli bir dizinin elemanlarını küçükten büyüğe doğru sıralayarak ekrana yazdıran programı akış diyagramları kullanarak yapınız.

EK I- C# DERS İÇERİĞİ

C# PROGRAMLAMA İÇERİĞİ

Hafta 1

Kazanımlar

- C# programlama dilinde değişken tanımlayabilme,
- C# programlama dilinde değişkenlere değer atayabilme

İçerik

1. C# Programlamaya giriş
   1.1. Visual Studio NET
2. Değişkenler Ve Sabitler
   2.1. Değişkenler
   2.2. Değişkenleri İsimlendirme Kuralları
   2.3. Sabitler
   2.4. Veri tipleri
   2.5. Değer atama
3. Operatörler
   3.1. Aritmetiksel Operatörler
   3.2. Mantıksal Operatörler
   3.3. İşlem Önceliği
4. Tür dönüşümleri
5. Örnek problemlerin çözümü

Hafta 2

Kazanımlar

- C# programlama dilinde girdi ve çıktı eylemlerini gerçekleştirebilme

İçerik

1. C# Girdi kodları
   1.1. Console.Read()
   1.2. Console.ReadLine()
2. C# Çıktı kodları
   2.1. Console.Write()
   2.2. Console.WriteLine()
3. Örnek problemlerin çözümü

Hafta 3

Kazanımlar

- C# programlama dilinde karar verme ve kontrol işlemlerini gerçekleştirebilme.

İçerik

1. C# - Karar Verme ve Kontrol Deyimleri
   1.1. If-Else Deyimi
   1.2. Switch-Case Deyimi
2. Örnek problemlerin çözümü

Hafta 4

Kazanımlar

- C# programlama dilinde döngü işlemlerini gerçekleştirebilme.

İçerik

1. C# - Döngü Yapıları
   1.1. For Döngüsü
   1.2. While Döngüsü
   1.3. Do…While Döngüsü
   1.4. Foreach Döngüsü
   1.5. Jump (Dallanma – Atlama) Komutları
   1.6. Break Anahtar Sözcüğü
   1.7. Continue Anahtar Sözcüğü
   1.8. Goto Anahtar Sözcüğü
   1.9. Return Anahtar Sözcüğü

Hafta 5

Kazanımlar

- C# programlama dilinde dizi işlemlerini gerçekleştirebilme.

İçerik

1. Diziler
   1.1. Dizi Oluşturma
   1.2. Diziye Değer Girme
   1.3. Diziyi Yazdırma
   1.4. Dizi Metotları
      1.4.1. Length
      1.4.2. Reverse
      1.4.3. Sort
      1.4.4. Clear
      1.4.5. IndexOf
   1.5. Capacity Özelliği
   1.6. Count Özelliği

1.7. Add Metodu
1.8. Insert Metodu
1.9. Remove Metodu

Hafta 6

Kazanımlar

- C# programlama dilinde metotları kullanabilme

İçerik

1. Metotlar
   1.1. Metot Kavramı - Tanımlama
   1.2. Metotlarda Parametre Kullanımı
   1.3. Özyineli (Rekürsif-Recursive) Metotlar
   1.4. Main( ) Metodu
2. Metinsel (String) Metotları
   2.1. Compare( )
   2.2. Concat( )
   2.3. Copy( )
   2.4. Format( )
   2.5. IsNullOrEmpty( )
   2.6. CompareTo ( )
3. Matematiksel (Math) Metotları
   3.1. Pow( )
   3.2. Max( ) / Min ()
   3.3. Ceiling ()
   3.4. Round ()
   3.5. Sqrt( )
   3.6. Cos ( ) Sin ( ) Tan ( )
   3.7. ToLower ( ) / ToUpper ( )
   3.8. Tarih/Saat (DateTime) Metotları
4. C# Dosya İşlemleri

EK J-PROBLEMLER PUANLAMA ANAHTARI

Okuyucu: **Öğrenci No:**

Problem 1: Bir X firması çalışanlarının maaşını hesaplayan bir programa ihtiyaç duymaktadır. Bir yazılımcı olarak sizden "Bir çalışanın çalışma saati ve saatlik ücreti klavyeden girildikten sonra aşağıdaki kurallara göre çalışanın net maaşını hesaplayarak ekrana yazan programı" hazırlamanız istenmektedir.

Kurallar:

- Brüt maaş çalışma saati ile saatlik ücretin çarpımı ile bulunur.
- Brüt maaştan %10 SSK kesintisi kesilmektedir
- Brüt maaştan %15 gelir vergisi kesilmektedir.
- Net maaş; brüt maaştan kesintilerden sonra geriye kalan maaştır.

Çözüm Adımları	**Örnek Kod Bloğu (Algoritma)**
Değişkenleri tanımlama	*Başla* *Çalışma saati, saatlik ücret, brut ücret, net ücret, brüt ücret, SSK, Gelir vergisi*
Değişkenlere girdi sağlama	*OKU Çalışma saati // Çalışma saatini giriniz* *OKU Saatlik ücret //Saatlik ücretini giriniz*
Değişkenlere değer atama (İşlem1)	*Brüt Ücret = çalışma saati * saatlik ücret*
Değişkenlere değer atama (İşlem2)	*SSK= 0,10*Brüt ücret*
Değişkenlere değer atama (İşlem3)	*Gelir vergisi= 0,15*Brüt ücret*
Değişkenlere değer atama (İşlem4)	*Net Ücret= Brüt ücret - (SSK + Gelir vergisi)*
Sonuca ulaşma ve ekrana yazdırma	*Yaz Net Ücret // Net Ücret değişkenini yazınız* *Bitir*

Çözüm Adımları	**Örnek Kod Bloğu (C#)**
Değişkenleri tanımlama	*{ int calisma_saati;* *int saat_ucreti;* *double ssk;* *double vergi;* *int brut_ucret;* *double net_ucret;*
Değişkenlere girdi sağlama	*saat_ucreti= Convert.ToInt32(Console.ReadLine());* *calisma_saati= Convert.ToInt32(Console.ReadLine());*
Değişkenlere değer atama (İşlem1)	*brut_ucret= saat_ucreti*calisma_saati;*
Değişkenlere değer atama (İşlem2)	*ssk= brut_ucret*0.1;*
Değişkenlere değer atama (İşlem3)	*vergi= brut_ucret*0.15;*
Değişkenlere değer atama (İşlem4)	*net_ucret= brut_ucret-(ssk+vergi);*
Sonuca ulaşma ve ekrana yazdırma	*Console.WriteLine(net_ucret);* *}*

Problem 2 Bir matematik öğretmeni kenar değerlerine göre bir üçgenin çizilip çizilmeyeceğini belirleyen bir eğitim yazılımına ihtiyaç duymaktadır. Buna göre bir yazılımcı olarak sizden; klavyeden 3 adet kenar uzunluğu girildikten sonra aşağıdaki üçgen çizilme kurallarına göre;

c. Üçgenin çizilip çizilemeyeceğini belirleyen ve üçgen çizilemez ise bunu ekrana yazan;
d. Eğer üçgen çizilirse üçgenin çeşidini(ikizkenar, çeşitkenar, eşkenar) ekrana yazan programı hazırlamanız istenmektedir.
Lütfen aşağıdaki kurallara dikkat ediniz.

Üçgenin çizilip çizilemeyeceği ile ilgili kurallar:

c. Bir kenarı diğer iki kenarının toplamından küçük olmak zorundadır. (Örneğin a> b+c gibi)
d. Bir kenar diğer iki kenarın farkından büyük olmak zorundadır. (Örneğin a<b-c gibi)

Çözüm Adımları	**Örnek Kod Bloğu (Algoritma)**
Değişkenleri tanımlama	*Başla* *A kenarı, B Kenarı, C Kenarı*
Değişkenlere girdi sağlama	*Oku A kenarı //A kenarını giriniz* *Oku B kenarı //B kenarını giriniz* *Oku C kenarı //C kenarını giriniz*
1. Mantıksal sınamayı gerçekleştirme	*Eğer (A kenarı < B kenarı + C kenarı) VE (B kenarı < A kenarı + C kenarı) VE (C kenarı < A kenarı + B kenarı) VE (A kenarı > B kenarı -C kenarı) VE (B kenarı > A kenarı - C Kenarı) VE (C kenarı > A kenarı - B kenarı)*
2. Mantıksal sınamayı gerçekleştirme ve ekrana yazdırma	*Eğer (A kenarı = B kenarı) VE (A kenarı = C kenarı) VE (C kenarı = B kenarı) ise* *Eşkenar Üçgen Yazınız*
3. Mantıksal sınamayı gerçekleştirme ve ekrana yazdırma	*Değilse* *Eğer (A kenarı = B kenarı) VEYA (A kenarı = C kenarı) VEYA (C kenarı = B kenarı) ise* *İkizkenar Üçgen yazınız* *Değilse* *Çeşitkenar Üçgen Yazınız* *Eğer Sonu* *Eğer Sonu*
Mantıksal sınamayı sonlandırma ve ekrana yazdırma	*Değilse* *Üçgen Kuralına uymuyor yazınız.* *Eğer Sonu* *Bitti*

Çözüm Adımları	**Örnek Kod Bloğu (C#)**
Değişkenleri tanımlama	int a; int b; int c;
Değişkenlere girdi sağlama	a= Convert.ToInt32(Console.ReadLine()); b= Convert.ToInt32(Console.ReadLine()); c= Convert.ToInt32(Console.ReadLine());
1. Mantıksal sınamayı gerçekleştirme	if (a<b+c && b<a+c && c<b+a && a>b-c && b>a-c && c>a-b) {
2. Mantıksal sınamayı gerçekleştirme ve ekrana yazdırma	if (a == b && b == c && a== c) {Console.WriteLine("Eşkenar Üçgen"); }
3. Mantıksal sınamayı gerçekleştirme ve ekrana yazdırma	Else { if (a == b || b == c || a== c) {Console.WriteLine("İkizkenar Üçgen");} else {Console.WriteLine("Çeşitkenar Üçgen");} } }
Mantıksal sınamayı sonlandırma ve ekrana yazdırma	else {Console.WriteLine(" Üçgen değil");} } }

Problem 3 Bir lise öğrencisi bir kimya problemi ile ilgili bir yazılım oluşturmak istemektedir. Probleme göre; 4,5 kg (4500 gram) ağırlığındaki buz kütlesi sabit sıcaklıktaki oda koşullarında 30 saniyede 5 gram erimektedir. Buna göre 1 saat sonunda buz kütlesinden kaç gram kaldığını hesaplayıp ekrana yazan programı tasarlamasına yardımcı olunuz.

Yukarıdaki kuralları dikkate aldığınızda:

a)Programın algoritmasını yazınız

b) Programın C# kodlarını yazınız.

c) Bu programı hazırlarken hangi programlama yapılarını kullandınız? Neden bu yapıları seçtiniz?(Kullandığınız her bir yapı için lütfen açıklayınız?)

Çözüm Adımları	**Örnek Kod Bloğu (Algoritma)**
Değişkenleri tanımlama	*Başla* *Kutle,zaman*
Değişkenlere Değer Atama	*Kutle= 4500* *Zaman= 0*
Döngü bloğunu Oluşturma	*Döngü Zaman = 300 olana kadar tekrarla // Döngü zaman= 3600, 0,30* *Zaman= Zaman + 30 // Zaman değişkenini 30 artır* *Kutle = Kutle - 5 // Kutle değişkenini 5 azalt* *Döngü sonu*
Sonuca ulaşma ve ekrana yazdırma	*Yaz "Kalan kütle=", Kutle* *Bitir*

Çözüm Adımları	**Örnek Kod Bloğu (C#)**	
Değişkenleri tanımlama ve değer atama	*{* *int g= 4500;* *int t= 0;*	
Döngü bloğunu Oluşturma	*for (t = 3600;t >= 0; t-=30)* *{* *g-= 5;* *}*	*do* *{* *t+= 30;* *g-= 5;* *}* *while(t<=3600);*
Sonuca ulaşma ve ekrana yazdırma	*Console.WriteLine(g);* *}*	

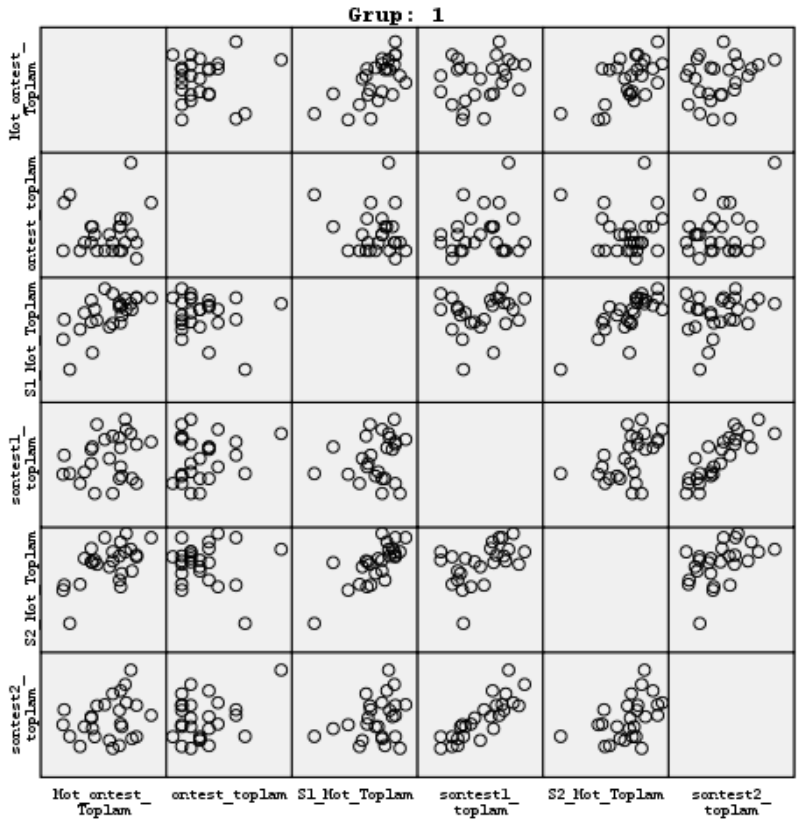
Problem4: Bir futbol antrenörü futbolcuların performanslarına göre sahaya süreceği ilk 11' i oluşturmaktadır. Performans puanı en büyük ilk 11 kişi maçta ilk 11 de yer almaktadır. Buna göre bir yazılımcı olarak sizden; 18 kişilik takımın performansının klavyeden sırayla girilerek, futbolcuların performans puanını büyükten küçüğe sıralayan ve ilk 11 kişinin performans puanlarını ekrana sırasıyla yazan programı tasarlamanız istenmektedir.

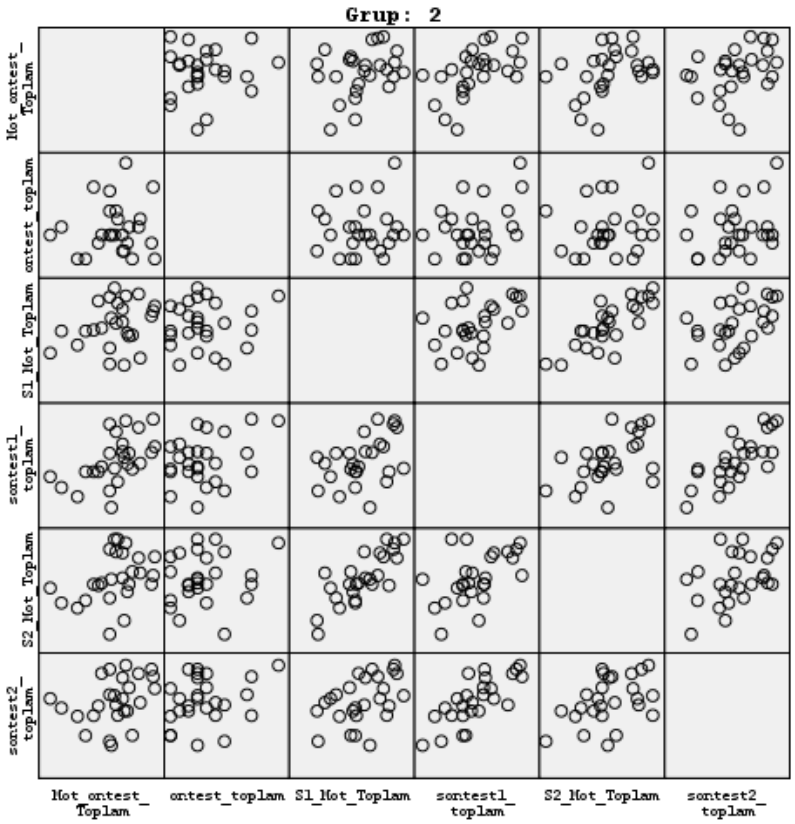
Çözüm Adımları	**Örnek Kod Bloğu (Algoritma)**
Değişkenleri tanımlama	*Başla* *i,j,sayı* *i = 0; // i değişkenini 0 yap* *j= 0; // j değişkenini 0 yap* *sayı=0; // sayı değişkenini 0 yap*
Diziyi tanımlama	*Dizi D (18); // D dizisini oluştur*
1. Döngü bloğunu Oluşturma	*Döngü i=1,18,1 // i= 18 olana kadar tekrarla* *i değişkenini 1 arttır*
Diziye değer atama	*Yaz i ".değeri giriniz" // i ".değeri" giriniz* *Oku D (i); // sayı değişkenini yanıt yap* *sayı değişkenini D dizisinin i. sırasına ekle* *Döngü sonlandır*
2. Döngü bloğunu Oluşturma	*Döngü i=1,17,1 // i değişkenini 0 yap* *i = 17 olana kadar tekrarla* *i değişkenini 1 arttır*
3. Döngü bloğunu Oluşturma (İç döngü)	*Döngü j=i+1, 1,18 // j değişkenini i+1 yap* *j değişkeni 18 olana kadar tekrarla* *j değişkenini 1 artır*
Mantıksal sınamayı gerçekleştirme	*Eğer D(j) > D(i) ise*
Dizi değerinin takas işlemini gerçekleştirme	*sayı = D(i) // sayı değişkenini D dizisinin i. değeri yap* *D(i) = D (j) // D dizisinin i. değerini D dizisinin j. değeri yap* *D (j) = sayı // D dizisinin j. değerini sayı değişkeni yap* *Eğer sonu* *Döngü Sonu*
4. Döngü bloğunu Oluşturma	*Döngü i=0; 11,1 // i değişkenini 0 yap* *i = 11 olana kadar tekrarla* *i değişkenini 1 arttır*
Sonuca ulaşma ve ekrana yazdırma	*Yaz D(i) // D dizisinin i. değişkenini yaz* *Döngü Sonu* *Bitir*
Değişkenleri tanımlama	*int takas =0;* *int i,j;*
Diziyi tanımlama	*int[] A = new int[5];*
1. Döngü bloğunu Oluşturma	*for (i = 0; i < 5; i++)* *{*
Diziye değer atama	*Console.Write(i+1+".Sayı giriniz = ");* *A[i] =Convert.ToInt32(Console.ReadLine());* *}*
2. Döngü bloğunu Oluşturma	*for (i = 0; i < 4; i++)* *{*
3. Döngü bloğunu Oluşturma (İç döngü)	*for (j = i+1; j < 5; j++)* *{*
Mantıksal sınamayı gerçekleştirme	*if (A[j]<A[i])* *{*
Dizi değerinin takas işlemini gerçekleştirme	*takas = A[i];* *A[i] = A[j];* *A[j] = takas;* *}* *}* *}*
4. Döngü bloğunu Oluşturma	*for (i = 0; i< 5;i++)* *{*
Sonuca ulaşma ve ekrana yazdırma	*Console.WriteLine(i+1 + ".sayı" + A[i]);*

EK K- NORMAL DAĞILIM

	Grup	**Basıklık Kurtosis**	**Çarpıklık Skewness**	**Kolmogorov-Smirnov**	**Shapiro-Wilk**
Başarı Testi- Öntest	Kontrol	2.723	1.533	.056	.049
	Deney	.663	.980	.068	.085
GÖS- Öntest	Kontrol	.096	-.822	.200	.074
	Deney	.588	-.720	.103	.270
Başarı Testi- Sontest	Kontrol	-1.163	.007	.200*	.332
	Deney	-.438	.092	.195	.692
GÖS- Sontest	Kontrol	2.072	-1.314	.172	.067
	Deney	-.839	-.205	.200	.518
Başarı Testi- Sontest 2	Kontrol	-.233	.524	.200	.422
	Deney	-.669	-.274	.196	.279
GÖS- Sontest 2	Kontrol	1.989	-1.065	.200	.085
	Deney	-.250	-.226	.200	.549

EK L- DOĞRUSALLIK SAÇILIM GRAFİKLERİ





EK-M GÖNÜLLÜ KATILIMCI İZİN FORMU

Aşağıda imzası olan ben "Scratch İle Programlama Öğretiminin Motivasyon Ve Başarıya Etkisi " başlıklı çalışmaya katılmayı kabul ediyorum. Bu çalışmayı yürüten Osman EROL çalışmanın yapısı, amacı ve muhtemel süresi, ne yapmam gerektiği hakkında bana ayrıntılı sözlü ve yazılı bilgi verdi. Araştırmacıya çalışmasıyla ilgili her türlü soruyu sorma fırsatını buldum. Cevapları ve bana verilen bilgiyi anladım. Araştırmacı Osman EROL' a bilgilerin ayrıntılarını açıklamama ve benimle ilgili sırları koruması şartıyla benimle bu çalışmayı yapmasına izin veriyorum. Çalışma boyunca tüm kurallara uyacağıma, araştırmacı ile tam bir uyum içinde çalışacağıma ve konuyla ilgili herhangi bir sorun çıktığında hemen onu arayacağımı kabul ediyorum. Çalışma sürecinde çalışmada kullanılmak üzere benimle ilgili elde edilmiş yazılı, sözlü, ses kaydı ve video kaydı ile yapılan her türlü dokümanı araştırmacının kullanabileceğini kabul ediyorum.

OKUDUM VE ONAYLADIM.

Katılımcının Adı ve Soyadı:

İmza:

Adres:

Araştırmacının Adı ve Soyadı: Öğr. Gör. Osman EROL

Adresi: Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi

BURDUR

İmza:

KAYNAKÇA

Adleberg, B. M. (2013). *Scratch programming and remix culture: Gender differences in interaction and motivation for pre-adolescents*. Yayınlanmamış Yüksek Lisans Tezi, Georgetown University, Washington, D.C.

Akbulut, Y. (2010). *Sosyal bilimlerde SPSS uygulamaları: Sık kullanılan istatistiksel analizler ve açıklamalı SPSS çözümleri.* İstanbul: İdeal Kültür & Yayıncılık.

Akcaoglu, M. ve Koehler, M. J. (2014). Cognitive outcomes from the Game-Design and Learning (GDL) after-school program. *Computers & Education, 75*, 72-81.

Akcay, T. (2009). *Perceptions of students and teachers about the use of a kid" s programming language in computer courses*. Yayınlanmamış Yüksek Lisans Tezi, ODTÜ, Ankara.

Al-Bow, M., Austin, D., Edgington,J., Fajardo, R., Fishburn, J., Lara, C., Leutenegger, S. ve Meyer, S.(2009). Using game creation for teaching computer programming to high school students and teachers. *SIGCSE Bulletin*, *41*(3), 104-108.

Alimisi R. ve Winters N.(2010). Programming playfully for a real- life problem: conditional statements on the stage of Scratch. *Proceedings for Constructionism*, 16-20.

Al-Imany, S., Alizadeh, J. ve Nour, M. A. (2006). On the development of a programming teaching tool: the effect of teaching by templates on the learning process. *Journal of Information Technology Education*, *5*, 271-283.

Allan, V. ve Kolesar, M. V. (1997). Teaching computer science: a problem solving approach that works. *Sig CUE Outlook*, *25*, 2-10.

Anastasiadou, S.D. ve Karakos, A.S. (2011). The beliefs of electrical and computer engineering students' regarding computer programming. *The International Journal of Technology, Knowledge and Society*, *7*(1), 37-51.

Baldwin, L.P. ve Kuljis, J. (2000). Visualization techniques for learning and teaching programming. *Journal of Computing and Information Technology*, *8*(4), 285–291.

Basawapatna, A. R., Koh, K. H. ve Repenning, A. (2010). Using scalable game design to teach computer science from middle school to graduate school. *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education,*ACM, (224-228), New York, NY, USA . http://dl.acm.org/citation.cfm?id=1822154 adresinden 20.02.2015 tarihinde erişilmiştir.

Başer, M. (2013). Bilgisayar programlamaya karşı tutum ölçeği geliştirme çalışması. *The Journal of Academic Social Science Studies, 6* (6), 199 - 215.

Bayman, P.ve Mayer, R. (1988). Using conceptual models to teach BASIC computer programming. *Journal of Educational Psychology*, *80*(3), 291-298

Baytak, A. ve Land, S. (2011) An investigation of the artifacts and process of constructing computers games about environmental science in a fifth grade classroom. *Educational Technology Research and Development*, *59*, 765-782.

Becker, K. ve Parker, J.R. (2007). Serious games + computer science = serious CS. *Journal of Computing Sciences in Colleges*, *23*(2), 40–46

Bennedsen, J. ve Carpersen, M. E. (2008). Exposing the programming process. Bennedsen, J.,Carpersen, M. E. ve Kolling, M. (Eds.). *Reflection on the theory of programming: Methods and implementation* içinde (s.6-16). Springer Berlin Heidelberg New York

Bergin J. ve Martinez M. P. (1996), An overview of visualization: its use and design, Report of the Working Group on Visualization. *Integrating Tech. into C.S.E*., 6(96).

Bishop-Clark, C., Courte, J. ve Howard, E. V. (2007). A quantitative and qualitative investigation of using Alice programming to improve confidence, enjoyment and achievement among non-majors. *Journal of Educational Computing Research*, *37*(2) 193-207.

Brandt, E., Messeter, J. ve Binder, T. (2008). Formatting design dialogues – games and participation. *CoDesign*, *4*(1), 51–64.

Brennan, K. (2011). *Creative Computing: A design-based introduction to computational thinking*, Scratch Curriculum Guide, Harvard Graduate School of Education, http://scratched.media.mit.edu/sites/default/files/CurriculumGuide-v20110923.pdf adresinden 10.06.2014 tarihinde edinilmiştir.

Brennan, K., Balch, C. ve Chung, M. (2014). *Creatıve computıng an introductory computing curriculum using Scratch.* Scratch Curriculum Guide, Harvard Graduate School of Education http://scratched.gse.harvard.edu/guide adresinden 10.06.2014 tarihinde edinilmiştir.

Briggs, J., D. (2013). *Programming with Scratch software: The benefits for year six learners*.Yayınlanmamış Yüksek lisans tezi, Bath Spa University, İngiltere.

Brown, T. A. (2006). *Confirmatory factor analysis for applied research*. New York: Guilford Press.

Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *The Journal of Media Literacy Education*, *4*(2).

Büyüköztürk, Ş. (2009). *Sosyal bilimler için veri analizi el kitabı* (10. Baskı). Pegem Akademi Yayıncılık.

Büyüköztürk, Ş., Akgün, Ö. E., Kahveci, Ö. ve Demirel, F. (2004). Güdülenme ve öğrenme stratejileri ölçeğinin Türkçe formunun geçerlik ve güvenirlik çalışması. *Kuram ve Uygulamada Eğitim Bilimleri,4*(2), 207-239.

Büyüköztürk, Ş., Çakmak, E. K., Akgün, Ö. E., Karadeniz, Ş. ve Demirel, F. (2011). *Bilimsel araştırma yöntemleri* (8. Baskı). Ankara: Pegem Akademi Yayıncılık.

Calder, N. (2010). Using Scratch: an integrated problem-solving approach to mathematical thinking. *Australian Primary Mathematics Classroom*, *15*(4), 9-14.

Can, A. (2013). *SPSS ile bilimsel araştırma süresince nicel veri analizi*. Ankara: Pegem Akademi.

Carbonaro, M., Szafron, D., Cutumisu, M. ve Schaeffer, J. (2010). Computer-game construction: A gender-neutral attractor to Computing Science. *Computers and Education*, *55*(3), 1098-1111.

Chaffin, A., Doran, K., Hicks, D. ve Barnes, T. (2009). Experimental evaluation of teaching recursion in a video game. *Proceedings of the ACM SIGGRAPH Symposium on Video Games*, ACM, 79–86,New York.

Chiu, C.F. (2014). Teaching programming concepts to K-12 teachers with Scratch. *Journalism and Mass Communication*, *4*(2), 125-132.

Claypool, M. (2013). Dragonfly: strengthening programming skills by building a game engine from scratch. *Computer Science Education*, *23*(2), 112-137.

Cohen, J.W. (1988). *Statistical power analysis for the behavioral sciences* (2. Baskı). Hillsdale, NJ: Lawrence Erlbaum Associates.

Cooper, S.,Dann, W. ve Pausch, R. (2003). Using animated 3D graphics to prepare novicesfor CSI. *Computer Science Education*, *13*, 3-30.

Cooper, S., Powers, K., McNally, M., Goldman, K.J. Proulx, V. ve Carlisle M. (2006). Tools for teaching introductory programming: What works? *ACM SIGCSE Bulletin*, *38*, 560-561.

Çağıltay, N. E. (2007). Teaching software engineering by means of computer-game development: Challenges and opportunities. *British Journal of EducationalTechnology*, *38*(3), 405-415.

Çağıltay, N. E. ve Fal, M. (2013). *Scratch ile programlamayı öğreniyorum*. Ankara: ODTÜ Yayıncılık.

Çamoğlu, K. (2009). *Programlama ve veri tabanı mantığı*. İstanbul: Kodlab.

Çepni, S.,Bayrakçeken, S., Yılmaz, A., Yücel, C., Semerci, Ç., Köse, E., Sezgin, F., Demircioğlu, G. ve Gündoğdu, K. (2008). *Ölçme ve değerlendirme*. Ankara: Pagem Akademi.

Çokluk, Ö., Şekercioğlu, G. ve Büyüköztürk, Ş. (2010). *Sosyal bilimler için çok değişkenli istatistik SPSS ve LISREL uygulamaları*. Ankara: Pegem Akademi.

Deci, E. L., Kostner, R. ve Ryan, R. M. (2001). Extrinsic rewards and intrinsic motivation in education: Reconsidered once again. *Review of Educational Research*, *71*, 1–27.

Deek F. ve Espinosa, I. (2005). An evolving approach to learning problem solving and program development: The distributed learning model. *International Journal on E-Learning*, *4*, 409-426.

Denner, J. (2007). The girls creating games program: An innovative approach to integrating technology into middle school. *Meridian: A Middle School Computer Technologies Journal*, *1*(10).

Denner, J., Werner, L. ve Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts?.*Computers & Education*, *58*(1), 240-249.

Dönmez, O. (2008). *Algoritma öğretimi için bir görselleştirici geliştirimi*. Yayınlanmamış Yüksek Lisans Tezi, Ege Üniversitesi, İzmir.

Eker, M. (2011). *Algoritmayı anlamak (4. Baskı)*. Ankara: Nirvana Yayınları.

Ellis, P. D. (2010). *The essential guide to effect sizes: An introduction to statistical power, Meta-Analysis and the interpretation of research results*. Birleşik Krallık: Cambridge Üniversitesi Basımevi

Eryılmaz, S. (2003). *Algoritma tasarlama ve programlamaya giriş*. Ankara: Detay Yayıncılık.

Fadjo, C. L. (2012). *Developing computational thinking through grounded embodied cognition.* Yayınlanmamış Doktora Tezi, Columbia University, ABD.

Farkas, D. ve Murthy, N. (2005). Attitudes toward computers, the introductory course and recruiting new majors: Preliminary results. *17th Workshop of the Psychology*

of Programming Interest Group, 268–277, Citeseer. http://www.ppig.org/papers/17th-farkas.pdf adresinden 03.03.2014 tarihinde edinilmiştir.

Forte, A. ve Guzdial, M. (2005). Motivation and nonmajors in computer science: identifying discrete audiences for introductory courses. *IEEE Transactions on Education, 48*(2), 248–253.

Futschek, G. (2006). Algorithmic thinking: The key for understanding computerscience. *Informatics Education–The Bridge between Using and Understanding Computers Lecture Notes in Computer Science, 4226,* 159-168

Garner, S. (2007). A program design tool to help novices learn programming. *ICT: Providing choices for learners and learning*, 321-324 Singapore, http://www.ascilite.org.au/conferences/singapore07/procs/garner.pdf adresinden 03.03.2014 tarihinde edinilmiştir.

Garner, S., Haden, P. ve Robins, A. (2005). My program is correct but it doesn't run: a preliminary investigation of novice programmers' problems. *Proceedings of the 7th Australasian conference on Computing education, ACM*,173 -175, Newcastle: Australia, http://crpit.scem.uws.edu.au/confpapers/CRPITV42Garner.pdf adresinden 11.12.2013 tarihinde edinilmiştir.

Garris, R., Ahlers, R. ve Driskell, J.E. (2002). Games, motivation, and learning: A research and practice model. *Simul Gaming*, *33*(4), 441–467.

Gee, J. P. (2005). Learning by design: Good video games as learning machines. *ELearning*, *2*, 5-16.

Genç, Z. ve Karakuş, S. (2011). Tasarımla öğrenme: Eğitsel bilgisayar oyunları tasarımında Scratch kullanımı. *International Computer and Instructional Technologies Symposium ICITS, 981-987*, Elazığ, Türkiye.

Ginat, D. (2003). The novice programmers' syndrome of design-by-keyword. *ACM SIGCSE Bulletin Proceedings of the 8th annual conference on Innovation and technology in computer science education,* 154 -157, Thessaloniki, Greece.

Giordano, D. ve Maiorana, F. (2014). Use of cutting edge educational tools for an initial programming course. *Global Engineering Education Conference (EDUCON) IEEE*, 556-563, İstanbul, Türkiye. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6826147 adresinden 30.01.2015 tarihinde edinilmiştir.

Gülbahar, Y. ve Kalelioğlu, F. (2014). The Effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education-An International Journal, 13*(1), 33-50.

Gülmez, I. (2009). *Programlama öğretiminde görselleştirme araçlarının kullanımının öğrenci başarı ve motivasyonuna etkisi*. Yayınlanmamış Yüksek Lisans Tezi, Marmara Üniversitesi, İstanbul.

Habgood, J. ve Overmars, M. (2006). *The game maker's apprentice: Game development for beginners*. Berkeley, CA: Apress.

Harel, I. ve Papert, S. (1991). Software design as a learning enviornment. *Interactive Learning Environments*, *1*(1), 1-30.

Hava, K. (2012). *Eğitsel bilgisayar oyunu tasarlama yönteminin, ilköğretim 4.sınıf öğrencilerinin akademik başarısına etkisi*. Yayınlanmamış Yüksek Lisans Tezi, Gazi Üniversitesi, Ankara.

Hayes, E. R. ve Games, I. A. (2008). Making computer games and design thinking a review of current software and strategies. *Games and Culture*, *3*(3–4), 309–332.

Heersink, D. ve Moskal, B. M. (2010). Measuring high school students' attitudes toward computing. *Proceedings of the 41st ACM technical symposium on Computer science education*, 446–450, NY, USA, http://ims.mii.lt/ims/konferenciju_medziaga/SIGCSE%2710/docs/p446.pdf adresinden 10.12.2013 tarihinde edinilmiştir.

Hooper, D., Coughlan, J. ve Mullen, M. R. (2008). Structural equation modelling: Guidelines for determining model fit. *The Electronic Journal of Business Research Methods*, *6*(1),53 - 60.

Hovardoğlu, S. (2000). *Davranış bilimleri için araştırma teknikleri*. Ankara: Ve-Ga Yayınları.

Howland, K. ve Good, J. (2015). Learning to communicate computationally with Flip: A bi-modal programming language for game creation. *Computers & Education*, *80*, 224-240.

Hsu, H. ve M., J. (2014). Gender differences in Scratch game design. *2014 International Conference on Information, Business and Education Technology (ICIBET), Taiwan, http://www.atlantis-press.com/php/download_paper.php?id=11390* adresinden 25.02.2015 tarihinde edinilmiştir.

Hu, L.T. ve Bentler, P.M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling*, *6* (1), 1-55.

Huck, S. W. (2000). *Reading statistics and research (3. Baskı).* New York: Addison Wesley Longman.

Jenkins, T. (2002). On the difficulty of learning to program. Proceedings of 3rd annual conference of the LTSN-ICS, 53-58, Loughborough, United Kingdom, http://www.heacademy.ac.ukevents/conf2002/tjenkins.pdf adresinden 11.12.2013 tarihinde edinilmiştir.

Kafai, Y. B. (2006). Playing and making games for learning instructionist and constructionist perspectives for game studies. *Games and Culture*, *1*(1), 36–40.

Kafai, Y. B., Franke, M., Ching, C. ve Shih, J. (1998). Game design as an interactive learning environment fostering students' and teachers's mathematical inquiry. *International Journal of Computers for Mathematical Learning*, *3*(2), 149–184.

Kazimoglu, C., Kiernan, M., Bacon, L. ve MacKinnon, L. (2011). Understanding computational thinking before programming: Developing guidelines for the

design of games to learn introductory programming through game-play. Int*ernational Journal of Game-Based Learning (IJGBL)*, *1*(3), 30-52.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education, 73*, 26–39.

Kelleher, C. (2006). *Motivating programming: Using storytelling to make computer programming attractive to middle school girls.* Yayınlanmamış Doktora Tezi, Carnegie Mellon University, Pittsburg.

Kelleher, C. ve Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys, 37*(2), 83-137.

Kelloway, K. E. (1998). *Using LISREL for structural equation modeling: A researcher's guide*. London: Sage.

Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. *The Internet and Higher Education*, *8*(1), 13-24.

Kim, S., Chung, K. ve Yu, H. (2013). Enhancing digital fluency through a training program for creative problem solving using computer programming. *The Journal of Creative Behavior, 47*(3), 171-199.

Kinnunen, P. ve Malmi, L. (2008). CS minors in a CS1 course. *Proceeding of the Fourth International Workshop on Computing Education Research, ICER* (s.79 - 90). New York, USA, http://dl.acm.org/citation.cfm?id=1404529 adresinden 11.12.2013 tarihinde edinilmiştir.

Kirriemuir, J. ve McFarlane, A. (2004). *Literature review in games and learning (Nesta Futurelab Series, Report 8).* Bristol, UK: Nesta Futurelab.

Kline, R.B.(2005). *Principles and practice of structural equation modeling*. NY: Guilford Yayıncılık.

Koster, R. (2005). *A theory of fun for game design*. Scottsdale: Paraglyph Press.

Koorsse, M., Cilliers, C. ve Calitz, A. (2015). Programming assistance tools to support the learning of IT programming in South African secondary schools. *Computers & Educatio*n, *82*, 162-178.

Kurkovsky, S. (2009). Can mobile game development foster student interest in computer science? *Proceedings of the International IEEE Consumer Electronics Society's*, (92–100), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5293601&tag=1 adresinden 20.02.2015 tarihinde edinilmiştir.

Kurland. D. M.,Pea. R. D., Clement. C. ve Mawby, R. (1989). A study of the development of programming ability and thinking skills in high school students. E. Soloway ve J.C. Spohrer (Ed.). *Studying the novice programmer* içinde (s.83-112). Hillsdale. NJ: Lawrence Erlbaum.

Ladd, B. C. (2006). The curse of monkey island: Holding the attention of students weaned on computer games. *CCSC-Northeastern Conference*, 2006, 162-174.

Lahtinen, E., Ala-Mutka, K. ve Järvinen, H. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin, 37*(3), 14-18.

Leiva, F. A. J. ve Salas, C. A. C. (2013). Practices of advanced programming: Tradition versus innovation. *Computer Applications in Engineering Education, 21*(2), 237-244.

Li, Q. (2012). Understanding enactivism: a study of affordances and constraints of engaging practicing teachers as digital game designers. *Educational Technology Research & Development, 60*, 785–806.

Linn, M. C. ve Dalbey, J. (1985). Cognitive consequences of programming instruction: Instruction, access, andability. *EducationalPsychologist, 20,* 191-206.

McGill, T. J., Volet, S. E. (1997). A conceptiual framework for analyzing students' knowledge of programming. *Journal of Research on Computing in Education, 29*(3), 276-197.

Malan, D. J. ve Leitner, H. H. (2007). Scratch for budding computer scientists. SIGCSE *Bulletin, 39*(1), 223-227.

Malone, T.W.(1981). What makes things fun to learn? A study of intrinsically motivating computer games. *Pipeline, 6*(2).

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M.ve Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. *ACM SIGCSE Bulletin, 40*(1), 367-371.

Maloney, J., Resnick, M., Rusk, N., Silverman, B. ve Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE),10*(4).

Mayer, R. E. (1981). The psychology of how novices learn computer programming. *Computing surveys, 13*, 121-141.

Miles, M. B. ve Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook (2. Baskı)*. California: SAGE Yayınları.

Mishra, P. ve Girod, M. (2006). Designing learning through learning to design. *The High School Journal. 90*(1). 44 – 51.

Molins-Ruano, P., Sevilla, C., Santini, S., Haya, P.A., Rodríguez, P. ve Sacha, G.M. (2014). Designing videogames to improve students' motivation. *Computers in Human Behavior, 31*, 571-579.

Monroy-Hernández, A. ve Resnick, M. (2008). Empowering kids to create and share programmable media. *Interactions, 15*(2).

Moreno, J. (2012). Digital competition game to improve programming skills. *Educational Technology & Society, 15*(3), 288–297.

Muller, M. J. (2003). Participatory design: the third space. HCI. J. A. Jacko ve A. Sears (Eds.). *Human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications. Human factors and Ergonomics* içinde (1051–1068). Hillsdale, NJ: L. Erlbaum Associates.

Munson, A., Moskal, B., Harriger, A., Karriker, T. ve Heersink (2011). Computing at the high school level: Changing what teachers and students know and believe. *Computers & Education, 57,* 1836–1849.

Navarrete, C. C. (2013). Creative thinking in digital game design and development: A case study. *Computers & Education, 69,* 320-331.

Nelson, J. ve Braafladt, K. (2012). *Technology and literacy: 21st century library programming for children and teens*. Chicago: American Library Association.

Nikou, S. A. ve Economides, A. A. (2014). Transition in student motivation during a Scratch and an App Inventor course. *Global Engineering Education Conference (EDUCON) IEEE* (1042-1045), İstabul, Türkiye. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6826234 adresinden 28.01.2015 tarihinde edinilmiştir.

Oblinger, D. (2004). The next generation of educational engagement. *Journal of Interactive Media in Education, 8,* 1–18.

Osman, M. A., Zakaria, M. N., Loke, S. P. ve Downe, A. G. (2012). Secondary students' perfectionism and their response to different programming learning tools. *Humanities, Science and Engineering (CHUSER*) *IEEE* (584-588), Malaysia, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6504380 adresinden 25.02.2015 tarihinde edinilmiştir.

Özçelik D.A. (2010). *Test hazırlama kılavuzu (4.Baskı)* .Ankara: Pegem Akademi.

Özdener, N. (2008). A comparison of the misconceptions about the time efficiency of algorithms by various profiles of computer programming students. *Computers and Education, 51,* 1094–1102.

Özkan, Y. (2003). *Programlama dilleri: C ile programlama*. İstanbul:Alfa Yayınları.

Özoran, D., Cagiltay, N. E. ve Topallı, D. (2012). Using Scratch in introduction to programming course for engineering students. *Proceedings of 2nd International Engineering Education Conference (IEEC2012),* Antalya, Turkey, http://www.atilim.edu.tr/~nergiz/pp/b31.pdf adresinden 28.01.2015 tarihinde edinilmiştir.

Pallant, J. (2001). *SPSS survival manual* .Maidenhead, PA: Open University Press.

Papastergiou, M. (2009). Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation *Computers & Education, 52,*1–12

Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. New York: Basic Books.

Papert, S. (1987). Microworlds: Transforming education. R.W., Lawler ve M., Yazdani, (Eds.) *Artifical Intelligence and Education* içinde. Ablex, Norwood, USA

Papert, S. (1991). Situating constructionism. S. Papert ve I. Harel (Eds.). *Constructionism* içinde, (s.1-11). Norwood, NJ: Ablex Publishing Corporation

Pearson, E.S. ve Hartley, H.O.(1958). *Biometrika tables for statisticians*. Newyork: Cambridge University Press.

Peppler, K. ve Kafai, Y.B. (2007). What video game making can teach us about literacy and learning: Alternative path ways into participatory culture. A. Baba (Ed.), *Situated Play: Proceedings of the Third International Conference of the Digital Games Research Association (DiGRA)(*369-376), Tokyo, Japan.

Pintrich, P. R., Smith, D., Garcia, T., ve McKeachie, W. J. (1993). *Reliability and predictivevalidity of the motivated strategies for learning questionnaire* (MSLQ). Washington, DC: Office of Educational Research and Improvement.

Porter, R. ve Calder, P. (2004). Patterns in learning to program - An experiment? A*ustralian Computing Education Conference.* Dunedin*:* NewZeland, http://crpit.com/confpapers/CRPITV30Porter.pdf adresinden 11.12.2013 tarihinde edinilmiştir.

Prensky, M. (2001). Fun, play and games: What makes games engaging? *Digital Game-Based Learning* içinde, McGraw-Hill, New York, NY, USA.

Rajaravivarma, R. (2005). A games-based approach for teaching the introductory programming course. *ACM Inroads - The SIGSCE Bulletin*, *37*, 98-102.

Ramadhan, H. A. (2000). Programming by discovery. *Journal of Computer Assisted Learning*, *16*, 83-93.

Resnick, M. (2007). Sowing the seeds for a more creative society. *Learning and Leading with Technology*, *35*(4), 18-22.

Resnick, M. (2012). Reviving Papert's dream. *Educational Technology*, *52*(4), 42-46.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J.,Silverman, B. ve Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60-67.

Rizvi, M., Humphries, T., Major, D., Jones, M. ve Lauzun, H. (2011). A CS0 course using scratch. *Journal of Computing Sciences in Colleges*, *26*(3), 19-27.

Robertson, J. ve Howells, C. (2008). Computer game design: opportunities for successful learning. *Computers & Education*, *50*(2), 559–578.

Robins, A., Rountree, J. ve Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, *13*(2), 137-172.

Salleha, S.,M., Shukura, Z. ve Judib, H. M. (2013). Analysis of research in programming teaching tools: An initial review. *Social and Behavioral Sciences 103*, 127 – 135.

Salant, O. M., Armoni M.,ve Ben-Ari M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239-264.

Scratch. (2014). *Scratch hakkında*, https://scratch.mit.edu/about/ adresinden 10.06.2014 tarihinde edinilmiştir.

Scratch. (2015). *Bir bakışta topluluk istatistikleri*, https://scratch.mit.edu/statistics/ adresinden 14.05.2015 tarihinde edinilmiştir.

Schulte, C. ve Bennedsen, J. (2006). What do teachers teach in introductory programming? *The Second International Computing Education Research Workshop,* University of Kent, Canterbury, United Kingdom.

Schunk, D. (2003). *Learning theories. An educational perspective*. Upper Saddle River: Merrill.

Shneiderman, B. ve Mayer, R. (1979). Syntactic/semantic interactions in programmer behavior: a model and experimental results. *International Journal of Computer and Information Sciences, 8*(3), 219-238.

Soloway, E.(1986). Learning to program = learning to construct mechanisms and explanations. *Communications of the ACM,29,* 850-858.

Soloway, E.ve Spohrer, J. C. (1989). *Studying the novice programmer*. Hillside, N.J. Erlbaum.

Smeets, E. (2005). Does ICT contribute to powerful learning environments in primary education. *Computers and Education, 44*(3), 343–355.

Sümer, N. (2000). Yapısal eşitlik modelleri. *Türk Psikoloji Yazıları, 3*(6), 49-74.

Tabachnick, G.B. ve Fidell, L.S. (2001). *Using multivariate statistics (4. Baskı).* MA: Allyon and Bacon Press.

Tüzün, H., Soylu, Y.M., Karakus, T., İnal, Y. ve Kızılkaya, G. (2009).The effects of computer games on primary school students' achievement and motivation in geography learning. *Computers & Education, 52,* 68–77.

Tekerek, M. ve Altan, T. (2014). The effect of Scratch environment on student's achievement in teaching algorithm. *World Journal on Educational Technology, 6*(2), 132-138.

Utting, I., Cooper, S., Kölling, M., Maloney, J. ve Resnick, M. (2010). Alice, greenfoot and scratch –A discussion. *ACM Transactions on Computing Education, 10*(4).

Vatansever, F. (2011). *Algoritma geliştirme ve programlamaya giriş*. Ankara: Seçkin Yayınevi.

Yıldırım, A.ve Simsek H. (2000). *Sosyal bilimlerde nitel arastırma yöntemleri*. Ankara: Seçkin Yayıncılık.

Yucel, I., Zupko, J. ve Seif El Nasr, M.(2006). Education, IT, girls, and game modding. *International Journal of Interactive Technology and Smart Education Journal, Special Issue on Smarter Use of Technology in Education, 3*(2), 143-156.

Yükseköğretim Kurulu. (2007). *Eğitim fakültesi öğretmen yetiştirme lisans programları bilgisayar ve öğretim teknolojileri öğretmenliği lisans programı. http://www.yok.gov.tr/documents/10279/49665/bilgisayar_ogretim/86c99d2e-3973-41c6-9e98-ed6d816391db* adresinden 20.06.2014 tarihinde edinilmiştir.

Wang, K., McCaffrey, C., Wendel, D. ve Klopfer, E. (2006). 3D game design with programming blocks in StarLogo TNG. *Proceedings of the 7th International Conference on Learning Sciences, ACM* (1008-1009).Bloomington,USA, http://dl.acm.org/citation.cfm?id=1150223 adresinden 10.04.2014 tarihinde edinilmiştir.

Westcott, S. (2008). *Effectiveness of using digital game playing in a first-level programming course*. Yayınlanmamış Doktora Tezi, PaceUniversity, ABD.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49,* 33–35.

Wing, J., M. (2008) Computational thinking and thinking about computing. *Philosophical Transactions of Royal societyA*, *366*, 3717-3725.

Winslow, L. E. (1996). Programming pedagogy – A psychological overview. *SIGCSE Bulletin, 28*, 17-22.

Wolz, U., Leitner, H. H., Malan, D. J. ve Maloney, J. (2009). Starting with Scratch in CS1. *SIGCSE Bulletin*, *41*, 2–3.

Wyffles, F., Martens, B. ve Lemmens, S. (2014). Starting from scratch: experimenting with computer science in Flemish secondary education. *Proceedings of the 9th Workshop in Primary and Secondary Computing EducationACM* (12-15), Berlin, Germany, http://dl.acm.org/citation.cfm?id=2670763 adresinden 25.02.2015 tarihinde edinilmiştir.