

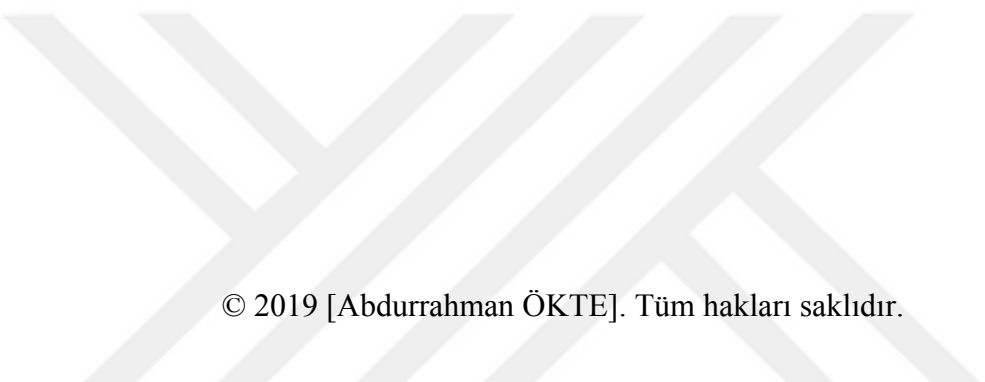
T.C.
SÜLEYMAN DEMİREL ÜNİVERSİTESİ
EĞİTİM BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ ANABİLİM DALI

**ÖN LİSANS ÖĞRENCİLERİNİN PROGRAMLAMA DİLİ ÖĞRETİMİNE
YÖNELİK GENEL GÖRÜŞLERİ, KARŞILAŞTIKLARI ZORLUKLAR VE
ÖNERİLERİNİN İNCELENMESİ**

Abdurrahman ÖKTE

Danışman: Prof. Dr. Mustafa KOÇ

YÜKSEK LİSANS TEZİ
ISPARTA 2019



© 2019 [Abdurrahman ÖKTE]. Tüm hakları saklıdır.

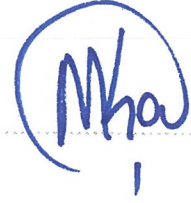
TEZ ONAYI

Abdurrahman ÖKTE tarafından hazırlanan “Ön Lisans Öğrencilerinin Programlama Dili Öğretimine Yönelik Genel Görüşleri, Karşılaştıkları Zorluklar ve Önerilerinin İncelenmesi” adlı tez çalışması aşağıdaki jüri üyeleri önünde Süleyman Demirel Üniversitesi Eğitim Bilimleri Enstitüsü Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı’nda **YÜKSEK LİSANS TEZİ** olarak başarı ile savunulmuştur.

Danışman

Prof. Dr. Mustafa KOÇ

Süleyman Demirel Üniversitesi



Jüri Üyesi

Doç. Dr. Muhammet DEMİRBİLEK

Süleyman Demirel Üniversitesi



Jüri Üyesi

Doç. Dr. Rabia Meryem YILMAZ

Atatürk Üniversitesi



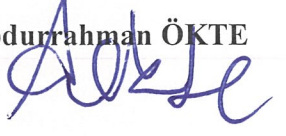
Enstitü Müdür Vekili

Dr. Öğr. Üyesi Ahmet ÖNAL



TAAHHÜTNAME

Bu tezin akademik ve etik kurallara uygun olarak yazıldığını ve alanyazından yapılan tüm alıntıların atıf yapılarak ve kaynakça bilgileri gösterilerek tezde yer aldığını beyan ederim.

Abdurrahman ÖKTE


İÇİNDEKİLER

İÇİNDEKİLER	i
ÖZET	iii
ABSTRACT	iv
TEŞEKKÜR	v
TABLolar DİZİNİ	vi
ŞEKİLLER DİZİNİ	vii
SİMGELER VE KISALTMALAR DİZİNİ	viii
1. GİRİŞ	1
1.1. Problem Durumu	1
1.2. Araştırmanın Amacı	1
1.3. Araştırmanın Önemi	2
1.4. Varsayımlar	3
1.5. Sınırlılıklar	3
2. KAVRAMSAL ÇERÇEVE VE İLGİLİ ÇALIŞMALAR	5
2.1. Programlama	5
2.1.1. Programlama dilleri	6
2.1.2. Programlama öğretimi	10
2.1.3. Programlama öğretiminin önemi	12
2.1.4. Programlama öğretiminde yaşanan zorluklar	13
2.1.5. Türkiye ve dünyada programlama öğretimi	17
2.1.5.1. Türkiye’de programlama öğretimi	18
2.1.5.2. Türkiye’de programlama öğretimi üzerine yapılan çalışmalar	26
2.1.5.3. Dünyada programlama öğretimi	32
2.1.5.4. Dünyada programlama öğretimi üzerine yapılan çalışmalar	36
3. YÖNTEM	40
3.1. Araştırma Modeli	40
3.2. Araştırmanın Çalışma Grubu	41
3.3. Veri Toplama Süreci	42
3.4. Veri Toplama Aracı	44
3.5. Verilerin Analizi	47
4. BULGULAR	49
4.1. Katılımcıların Demografik Özellikleri	49
4.2. Katılımcıların Programlama Dili Öğreniminde Yaşadığı Problemler	50
5. TARTIŞMA VE SONUÇ	58
KAYNAKÇA	62
EKLER	73

Ek A. Odak Grup Görüşmesi Soruları	74
ÖZGEÇMİŞ	75



ÖZET

ÖN LİSANS ÖĞRENCİLERİNİN PROGRAMLAMA DİLİ ÖĞRETİMİNE YÖNELİK GENEL GÖRÜŞLERİ, KARŞILAŞTIKLARI ZORLUKLAR VE ÖNERİLERİNİN İNCELENMESİ

Abdurrahman ÖKTE

**Yüksek Lisans Tezi, Süleyman Demirel Üniversitesi, Eğitim Bilimleri Enstitüsü,
Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı**

Danışman: Prof. Dr. Mustafa KOÇ

2019, 75 sayfa

Bu araştırmanın amacı, ön lisans öğrencilerinin nesne tabanlı programlama dili öğretimine yönelik görüşleri, karşılaştıkları zorlukları ve önerilerini ortaya çıkarmaktır. Yöntem olarak nitel araştırma paradigmasına dayalı durum çalışması modeli kullanılmıştır. Çalışma grubu Burdur Mehmet Akif Ersoy Üniversitesi Bilgi Yönetimi ön lisans programlarında öğrenim görmekte olan 22 öğrenciden oluşmaktadır. Veriler odak grup görüşmesi yoluyla 4 farklı grup üzerinden toplanmıştır. Elde edilen bulgular programlama diline ilişkin görüşleri, programlamada karşılaşılan zorluklar, programlamanın zevkli yanları, programlamanın nasıl öğretilmesine yönelik görüşler, lise eğitiminin programlamaya katkısına yönelik görüşler, programlamada hata ile karşılaşınca yapılanlar ve programlamaya ayrılan vakit temaları altında toplanmıştır. Önceki öğrenim yaşantılarında programlama eğitimi almayanların başlangıç düzeyinde zorlandıkları ama ilerleyen süreçte eksikliklerini giderdikleri belirlenmiştir. Öğrencilerin en fazla problem yaşadığı konunun programlama dili yazım kuralları ile ilgili olduğu belirlenmiştir. Katılımcılar programlamanın; birebir eğitim yöntemleriyle ve uygulamalı olarak öğretilmesinin daha etkili olacağını inanmaktadır. Bu çalışmanın programlama dili öğretiminde yaşanan zorlukları ortaya çıkarması yönüyle kodlama eğitimine katkıda bulunacağı ve bu alanın eğitimcilerine daha etkili bir programlama öğretimi yapma noktasında yön gösterme ve rehberlik edeceği düşünülmektedir.

Anahtar Kelimeler: Programlama dili, kodlama, öğretim zorlukları, üniversite öğrencileri, görüşler

ABSTRACT

EXAMINATION OF ASSOCIATE’S DEGREE STUDENTS’ GENERAL OPINIONS, DIFFICULTIES AND SUGGESTIONS FOR TEACHING PROGRAMMING LANGUAGES

Abdurrahman ÖKTE

**Master’s Thesis, Süleyman Demirel University, Graduate School of Educational
Sciences, Department of Computer Education and Instructional Technologies**

Advisor: Prof. Mustafa KOÇ

2019, 75 pages

The aim of this study is to reveal associate’s degree students’ general opinions, difficulties and suggestions for teaching programming languages. A case study model based on the qualitative research paradigm was used as a research methodology. The study group was composed of 22 students who have been studying at Burdur Mehmet Akif Ersoy University. The data were collected through focus group interviews with 4 different groups. The findings were grouped under such themes as opinions on programming languages, difficulties in programming, enjoyable aspects of programming, opinions on how to teach programming, opinions on the contribution of high school education to programming, actions taken when faced with errors, and time devoted to programming. It was determined that those who did not have programming education in their previous education life had difficulty at the beginning level, but they were able to overcome their deficiencies during the process. In addition, it was found that the subject that students suffered most were related to the programming language writing rules. Participants believe that programming will be more effective with one-to-one training methods and practical teaching. It is thought that this study will contribute to the ongoing efforts on the development of coding curriculum and will guide the educators in this field on the point of having a more effective programming teaching.

Keywords: Programming language, coding, teaching difficulties, university students, opinions

TEŞEKKÜR

Tezimin hazırlanmasındaki her türlü destek ve katkılarından dolayı değerli tez hocam Prof. Dr. Mustafa KOÇ'a, yüksek lisans eğitimim boyunca faydalandığım saygıdeğer hocalarım Doç. Dr. Muhammet DEMİRBİLEK, Doç. Dr. Veysel DEMİRER ve Dr. Öğr. Üyesi Mehmet ALBAYRAK'a teşekkürü bir borç bilirim.

Yüksek lisans eğitimim boyunca birlikte çalışmaktan zevk aldığım yüksek lisans arkadaşlarıma desteklerinden dolayı teşekkür ederim.

Tez yazım sürecinde göstermiş oldukları yardımlarından dolayı Hüseyin FİDAN, Halil İbrahim DEMİR, Şaban HÖL, Murat KOÇAK ve Ali ÖKTE'ye de teşekkür ederim.

Beni akademik hayata teşvik eden ve her türlü desteğini esirgemeyen değerli aileme, sevgili eşim Ayşin ve oğlum Agah'a sonsuz teşekkürlerimi sunarım.

TABLÖLAR DİZİNİ

Tablo 1. Programlama dillerin tasarım nedenleri	8
Tablo 2. Blok tabanlı görsel programlama ortamları.....	9
Tablo 3. Programlama öğretimi zorlukları	15
Tablo 4. BTY dersi 5. sınıf problem çözme ve programlama ünitesi.....	19
Tablo 5. Türkiye’de kullanılan programlama dilleri	23
Tablo 6.Odak grup görüşme takvimi	44
Tablo 7. Katılımcıların cinsiyet, sınıf düzeyi ve lise türlerine göre dağılımı	49
Tablo 8. Önceki programlama deneyimleri	49
Tablo 9. Öğrencilerin kişisel bilgisayara sahip olma durumları	50
Tablo 10. Programlama diline ilişkin görüşleri	50
Tablo 11. Programlamada karşılaşılan zorluklar	51
Tablo 12. Programlamanın zevkli yanları.....	52
Tablo 13. Programlamanın nasıl öğretilmesine yönelik görüşler	53
Tablo 14. Lise eğitiminin programlamaya katkısına yönelik görüşler	54
Tablo 15. Programlamada hata ile karşılaşıncı yapılanlar	56
Tablo 16. Programlamaya ayrılan vakit.....	57

ŞEKİLLER DİZİNİ

Şekil 1. Programlama dilleri sınıflandırması	7
Şekil 2. Programlama deneyimleri başarısızlık nedenleri	16
Şekil 3. Programlamada karşılaşılan zorluklar	17
Şekil 4. Kodlama saati Türkiye etkinlikleri	18
Şekil 5. Github 2008-2017 Türkiye il istatistikleri	24
Şekil 6. Türkiye’de kodlama projesi yapan iller	24
Şekil 7. Programlama dilleri sıralaması	34
Şekil 8. En iyi programlama dilleri	35
Şekil 9. 2002-2018 programlama dilleri kullanım istatistikleri	35
Şekil 10. Ağustos 2017-2018 programlama dilleri sıralaması	36
Şekil 11. Odak grup görüşme mekanı tasarımı	43
Şekil 12. Odak grup görüşmesi planlaması	46

SİMGELER VE KISALTMALAR DİZİNİ

BTP	Blok Tabanlı Programlama
BTY	Bilişim Teknolojileri ve Yazılım
BY	Bilgi Yönetimi
EGTBMYO	Emin Gülmez Teknik Bilimler Meslek Yüksekokulu
f	Frekans
HTMYO	Hikmet Tolunay Meslek Yüksekokulu
IEEE	Elektrik ve Elektronik Mühendisleri Enstitüsü
İŞKUR	Türkiye İş Kurumu
MEB	Milli Eğitim Bakanlığı
MAKÜ	Mehmet Akif Ersoy Üniversitesi
SAP	Systems Analysis and Program Development
TBD	Türk Bilişim Derneği
TIOBE	The Importance of Being Earnest
YÖK	Yüksek Öğretim Kurulu
YSD	Yüksek Seviyeli Diller
%	Yüzde

1. GİRİŞ

1.1. Problem Durumu

Bilişim ve iletişim teknolojileri günlük hayatımızın hemen hemen her yönünü büyük ölçüde değiştirmiş (Demirbilek, 2014) insan hayatı üzerindeki büyümesi ve etkisi dikkat çekecek seviyelere gelmiştir (Koç, 2017). Buna bağlı olarak günümüz dünyasında programlama dili bilmek ve bilgisayar programları yazabilmek giderek önem kazanmakta ve yabancı dil öğrenmek kadar önemli bir unsur olarak görülmektedir (Aytekin, Sönmez Çakır, Yücel ve Kulaözü, 2018). Bilgisayar programlama işlemi bireylerin yaratıcılık, eleştirel ve mantıksal düşünme gibi üst düzey bilişsel becerilerini geliştirilebilmekte (Lye ve Koh, 2014; Saygıner ve Tüzün, 2017) ve aynı zamanda iş dünyasının çeşitli alanlarında anahtar bir yetkinlik olarak görülmektedir (Sayın ve Seferoğlu, 2016). Bu nedenle bazı ülkelerdeki eğitim politikaları, ilköğretim ve genel eğitime programlama öğretimini entegre ederek bilgisayar bilimi derslerini güncellemektedir (Chen vd., 2017; Gülbahar ve Kalelioğlu, 2018; Han, Bae ve Park, 2016; Heintz, Mannila ve Farnqvist, 2016; Lewis ve Shah, 2012). Dünya genelinde programlama alanındaki uzmanlar ve eğitimciler tarafından yapılan çalışmalar incelendiğinde, programlama eğitiminde birtakım zorlukların yaşandığı ve bu alandaki akademik başarıda arzu edilen düzeye henüz ulaşamadığı görülmektedir. Ayrıca programlama dili eğitimi için yapılan çalışmalara rağmen programlama dili öğrenciler için hala en çok zorlanılan ve başarısız olunan ders olarak görülmektedir (Özmen ve Altun, 2014). Algoritma ve programlama öğretiminde yaşanan problemler, programlama derslerinde öğrenci motivasyonunu düşürebilmekte ve ortaya çıkan olumsuz tutumlar, programlama ile ilgili öğrencilerin başarısını olumsuz yönde etkileyebilmektedir (Anastasiadou ve Karakos, 2011; Lahtinen, Ala-Mutka ve Jarvinen, 2005). Ön lisans öğrencilerinin programlama dili öğrenme de zorluklar yaşamasından yola çıkarak programlama dili öğrenmede yaşanan problemlerin ve bu problemlere çözüm önerilerinin araştırılması düşünülmüştür.

1.2. Araştırmanın Amacı

Bu araştırmanın temel amacı 2017-2018 eğitim-öğretim yılında Burdur Mehmet Akif Ersoy Üniversitesi (MAKÜ) Hikmet Tolunay Meslek Yüksekokulu (HTMYO) ve Emin Gülmez Teknik Bilimler Meslek Yüksekokulu (EGTBMYO) bünyesinde öğrenim gören

ön lisans öğrencilerinin programlama dili öğrenmeye yönelik görüşlerini, karşılaştıkları zorlukları ve programlama öğretimine yönelik önerilerini belirlemektir. Çalışmada kullanılan bazı değişkenler dayanak alınarak aşağıdaki sorulara cevap aranmaktadır;

1. Ön lisans öğrencilerinin nesne tabanlı programlama dili öğrenmeye yönelik genel görüşleri nelerdir?
2. Ön lisans öğrencilerine göre nesne tabanlı programlama dili öğrenmede sıklıkla karşılaşılan zorluklar nelerdir?
 - a. Ön lisans öğrencilerinin programlama diliyle kod yazarken bir hata ile karşılaştıklarında takip ettikleri adımlar nelerdir?
3. Ön lisans öğrencilerine göre nesne tabanlı programlama dili öğrenmenin zevkli yönleri nelerdir?
4. Ön lisans öğrencilerine göre nesne tabanlı programlama dili öğretimi nasıl olmalıdır?
5. Ön lisans öğrencilerine göre lise döneminde programlama dersi almanın yeni bir programlama dili öğrenmeye katkısı var mıdır?
6. Ön lisans öğrencileri kodlama yaparken yabancı dildeki ifadelerle sıklıkla karşılaşmalarını öğrenme performanslarına etkisi konusunda nasıl değerlendirmektedir?
7. Ön lisans öğrencileri kodlama esnasındaki karşılaştıkları zorluklardaki davranışları nasıldır?

1.3. Araştırmanın Önemi

Shin, Park ve Bae (2013) öğrencilerin yaratıcılıklarını ve problem çözme yeteneklerini geliştirmek için bilgisayar bilimi ve programlamayı öğrenmeleri gerektiğini savunmaktadır. Araştırmacılar farklı bakış açılarına sahip olsalar da öğrencilerin programlama öğrenmeleri esnasında birtakım zorluklarla karşılaştıkları konusunda hemfikirdirler (Çetin, 2013). Ayrıca, çoğu öğrenci ve öğretmen, programlamayı öğrenmenin zorlu bir görev olduğunu belirtmiştir (Govender ve Grayson, 2008; Kordaki, 2010; Yang, Hwang, Yang ve Hwang, 2015). Bilgisayar programı yazmak artık her meslekte kendine özgü uygulama geliştirme araçları sayesinde her alana girmiştir (Yaşar, 2012). Günümüzde ülkelerin kalkınmalarını sürdürebilmeleri için gereken önemli sektörlerden birisi de doğrudan ve dolaylı etkilerinden ötürü yazılım sektörüdür (Alican,

2006). Günümüzde çoğu sektörün yazılım çalışmalarına ihtiyacı vardır. Rekabet ortamında yazılım sektörünü dikkate almayan kurumlar zamanla konumlarını kaybetmektedir. Bu yönüyle yazılım bütün sektörlerin gelişmesinde etkili rol oynamaktadır. Bunun bilincinde olan gelişmiş ülkeler yazılım sektörünü teşvik etmektedir. Ülkemizde de son dönemlerde bilişim ve özellikle de yazılım sektörüne devletin önem verdiği ve bu doğrultuda çeşitli politikalar geliştirdiği görülmektedir. Özellikle üniversite bünyelerinde açılan teknokentler ve devletin teknokentlerde faaliyet gösteren firmalar için hazırlamış olduğu teşvik programları, yazılım sektörünün gelişmesinde ve bu sektörde çalışan insanlar için istihdam yaratılmasında büyük pay sahibi olmuştur (Konuk ve Öztürk, 2010). Türkiye’de her yıl 10 bin yeni yazılım elemanına ihtiyaç vardır. Ülkenin dijital dönüşümü ve ekonominin gelişmesinde önemli katkısı olacak (Sayın ve Seferoğlu, 2016) bu ihtiyacın karşılanması için yakın gelecekte kodlama eğitiminin okullarda zorunlu hale gelmesi beklenmektedir (Baz, 2018). Bu bağlamda bu tez çalışmasının, öğrencilerin programlama dili öğreniminde yaşadıkları zorlukları ortaya çıkarması yoluyla halen devam etmekte olan kodlama müfredatı geliştirme çabalarına katkıda bulunacağı ve bu alanın eğitimcilerine daha etkili bir programlama öğretimi yapma noktasında yön göstereceği ve rehberlik edeceği düşünülmektedir.

1.4. Varsayımlar

- Araştırma sonucunu etkileyecek dış etkenlerin tüm gruplardaki katılımcıları eşit düzeyde etkilediği,
- Araştırmaya katılan öğrencilerin, odak grup görüşmelerinde sorulara içtenlikle cevap verdikleri varsayılmıştır.

1.5. Sınırlılıklar

- Bu araştırmanın evreni MAKÜ HTMYO ve EGTBMYO Bilgi Yönetimi programlarında okuyan öğrencilerle sınırlıdır.
- Araştırmanın örneklemi 2017-2018 eğitim-öğretim yılında MAKÜ HTMYO ve EGTBMYO Bilgi Yönetimi programlarında okuyan öğrenciler arasından amaçsal örnekleme yoluyla oluşturulan gruplar ile sınırlıdır.

- Arařtırma bulgularının elde edilmesi, oluřturulan grřme formu ve yarı yapılandırılmış sorular ile sınırlıdır.



2. KAVRAMSAL ÇERÇEVE VE İLGİLİ ÇALIŞMALAR

2.1. Programlama

Bilgisayarların belli amaçlar doğrultusunda kullanılabilmesi için önce onların programlanması gerekmektedir. Başka bir deyişle bilgisayarların yapması istenen işlemler, onlara anlayacakları şekilde aktarılmalıdır. Bu talimat verme işlemi programlama yoluyla yapılmaktadır. Programlama, problem çözmek ya da bilgisayarın adım adım takip etmesi için talimat yazmaktan başka bir şey değildir (Wang, 2008). Ayrıca programlama sadece bir program yazmak değil, aynı zamanda kişinin tarzının yansımasıdır (Barlet ve Bruno, 2003). Programlama, önemli miktarda çaba ve farklı beceriler gerektiren problem çözme yöntemidir (Ersoy, Madran ve Gülbahar, 2011; Gomes ve Mendes, 2007). Rahmat vd. (2012) programlamayı programcı olmayan birçok kişinin tek bir etkinlik gibi gördüğünü, ancak gerçekte farklı problemleri çözmeyi amaçlayan çok çeşitli faaliyetler olarak tanımlamaktadır. Programlama, bir problemin anlaşılması, çözüm oluşturulması ve çözümün bilgisayarın problemi çözmek için kullanacağı şekilde yazılmasıdır.

Barlet ve Bruno (2003) programlamayı sadece bilgisayara mesaj ileten değil programınızı kim kullanır ya da modifiye ederse ona da mesaj ilettiğini ve bu yönüyle şiire benzediğini belirtmiştir. Usta programcılar bir program ile kendi dünyalarını kendi kurallarına göre yaratırlar. En iyi programcılardan biri olarak bilinen Donald Knuth, programlamanın nasıl bir şey yapılacağını bilgisayara anlatmak olmadığını, ancak bir kişiye bilgisayara bir şeyleri nasıl yapacağını öğretmek olarak tarif etmiştir. Buradaki önemli nokta, programların sadece bilgisayarlar tarafından değil insanlar tarafından da okunmasıdır. Bir programcı; problem çözücü, şair ve aynı zamanda eğitmendir (Barlet ve Bruno, 2003).

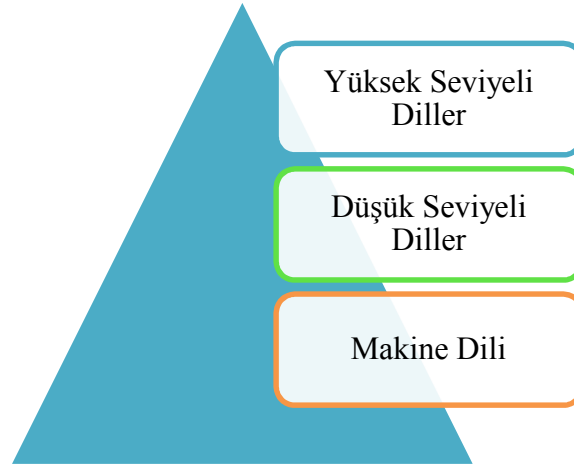
Wang (2008) bilgisayar programlamayı en basit düzeyde, bir bilgisayarın adım adım takip etmesi için talimat yazmak olarak tanımlamıştır. Programlamanın en önemli kısmını ise bir programın nasıl yazılacağını veya belirli bir programlama dilinin nasıl kullanılacağını bilmemek değil, ilk etapta neyin yaratılacağını bilmek olarak belirtmektedir. Önemli olan, bir bilgisayarın nasıl programlanacağını herkesin anlayabilmesidir. Bir bilgisayarı nasıl programlayacağınızı bilmek, programlama becerilerinizle ne yapacağınızı bilmekten daha önemlidir. Albert Einstein'ın dediği gibi, “Hayal, bilgidен daha önemlidir.” bir

program hakkında fikir sahibi olduktan sonra, fikrimizi gerçeğe dönüştürmek için programlama kullanılabilir (Wang, 2008). Programlamada teorik bilgi kadar uygulama da çok önemlidir. Lahtinen vd. (2005) öğrencilerin kavramları anlamak için pratik deneyime ihtiyaçları olduğunu, öğrenme durumları ve materyalleri ne kadar pratik ve somut olursa o kadar çok öğrenme gerçekleşeceğini ve yaparak öğrenmenin her zaman çalışmaların bir parçası olması gerektiğini belirtmektedir. Wang (2008), *“Bilgisayarsız, bilgisayar programlamada yetenekli olmaya çalışmak, su olmadan yüzmede yetenekli olmaya çalışmak gibidir.”* sözleriyle bunu vurgulamıştır. Programlama öğrenilmeye başlanan andan en son ana kadar bireyin kendi bilgilerini ve becerilerini sergilemesi gereken bir süreç işidir (Ersoy vd., 2011).

2.1.1. Programlama dilleri

Programlama dili, program yazmak için kullanılan yazılımlara (Arı, 2016) ve bilgisayar programları yazımında takip edilen kurallar dizisinin oluşturduğu gruba denir (Bal, 2009). Programlama dili aslında bilgisayara hükmetmek için kullanılan bir araçtır (Tungut, 2013). Bilgisayarlar icat edildikleri zamandan beridir farklı düzeyde programlama dilleri vasıtasıyla programlanarak insanlığa fayda sağlayan araçlar haline getirilmiştir (Algan, 2010). Bilgisayar üreten şirketlerin kendi ürettikleri bilgisayarlarını çalıştıracak programlama yöntemlerini bulmalarıyla, programlama dilleri çeşitlenmiştir (Bal, 2009). Programcılar tarafından birçok programlama dili kullanılarak yazılan bazı komutlar bilgisayarlar tarafından doğrudan doğruya anlaşılırken bazı komutlar ise çeviri (translation) sayesinde anlaşılabilir. Programlama dilleri mikroişlemciye yakınlığına göre gruplandırılabilir. Bir dil mikroişlemciye ne kadar yakınsa seviyesi o kadar düşüktür (Arı, 2016).

Günümüzde kullanılan yüzlerce bilgisayar dilini üç genel gruba ayırabiliriz (Şekil 1). Bunlar kullanım sıralarına göre Makine Dili, Düşük Seviyeli Diller (Assembly) ve Yüksek Seviyeli Diller olarak sıralanmaktadır.



Şekil 1. Programlama dilleri sınıflandırması

Makine Dili: Bilgisayarların ilk ortaya çıktığı zamanlarda programlama dili olmadığından dolayı bilgisayarlara komut verebilmek, bir şeyler yaptırabilmek için onun anlayabileceği makine dilinin kullanıldığı görülmektedir. Makine dili işlemcinin devrelerine bağımlı olduğu için makine değişince dili de değiştiğinden dolayı makine dilini öğrenenlerin, başka bilgisayarlarda yazamadıkları için zorluk çektikleri gözlemlenmiştir (Karaçay ve Karaçay, 2016). Makine dilinin 1 ve 0'lerden oluşan ve bilgisayara istenilen işlemleri yaptırmak için kullanılan ilk dil olduğu anlaşılmaktadır (Arıkan, 2017; Yumuşak ve Adak, 2018).

Düşük Seviyeli Diller: 1950'li yıllarda programcılar, makine dilleriyle programlamanın yavaş, zahmetli ve hata olasılığı yüksek olmasından dolayı İngilizce bazı kısaltmalar (Dikici, 2017) kullanarak sembolik makine dillerini “assembly dilleri” oluşturmuştur (Arıkan, 2017; Deitel ve Deitel, 2008). Assembly kullanılarak yazılan programları makine dilinde çalıştırabilmek için ise “Assembler” olarak adlandırılan çevirici programlar kullanılmıştır (Deitel ve Deitel, 2008). Ancak çevirme işlemine girdiği için programların makine diline göre yavaş çalıştığı görülmüştür (Yumuşak ve Adak, 2018).

Yüksek Seviyeli Diller: Assembly ile her ne kadar program yazmak kolaylaşmış gibi görünse de basit bir program yazmak için bile birçok kod yazılması gerekmektedir (Yumuşak ve Adak, 2018). Bundan dolayı programlamayı hızlandırmak için yüksek seviyeli diller (YSD) geliştirilmiştir. YSD, tek bir ifade ile birden fazla görev yerine getirilebilmekte ve günlük İngilizceye oldukça yakın sözcükler kullanılmaktadır. Dolayısıyla çoğu programcının makine dili ve assembly kullanmak yerine YSD tercih

ettiği görülmüştür (Deitel ve Deitel, 2008). YSD’ye örnek olarak Java, Python, C#, Delphi, PHP, R, Ruby, Go dilleri verilebilir.

Weinberg (2011) programlama dilleri arasında kıyaslama yaparken en fazla dikkat edilen konulardan birinin, hızlı derleme yapması olduğunu ve hız dışında değerlendirilen diğer boyutları da tanımlama, dilin kapsamı, nesne kodunun güvenilirliği ve yürütme süresi denetimi olarak sıralamaktadır. Toal, Rivera, Schneider ve Choe (2016) tüm dillerin kusurları olduğunu belirtmişlerdir. Çoğu programlama dilleri bazı işlemleri çok iyi yaparken bazı işlemleri yapmada yetersiz kalmaktadır. Bir dilin belirli görevler için uygunluğunun değerlendirilebileceğini ancak genel bir dil derecelendirmesi yapmaya gerek olmadığını ve bir dili yargılamak için neden tasarlandığını bilmemiz gerektiğini belirtmekte ve programlama dillerinin tasarım nedenlerini Tablo 1’deki gibi açıklamaktadır (Toal vd., 2016).

Tablo 1. Programlama dillerin tasarım nedenleri (Toal vd., 2016)

Programlama Dili	Tasarım Nedeni
Fortran	Sayısal hesaplama
COBOL	İş hesaplama
Pascal	Yapılandırılmış programlama öğretimi
C	Sistem programlama
SQL	Veritabanı uygulamaları
MATLAB	Sayısal hesaplama için alternatif bir Fortran olmak
C++	Simülasyon
Perl	Metin işlemeye odaklanan komut dosyaları
Python	Genişletilebilir bir dil ile komut dosyası oluşturma
Ruby	Matz'ın istediği dil olmak ve Perl'den iyi olmak
R	İstatistiksel ve grafiksel hesaplama
Java	Kompakt indirilebilir çalıştırılabilir bileşenler
C#	.NET platformunda kurumsal bilgi işlem
JavaScript	Web uygulamaları için istemci tarafı komut dosyası
PHP	Web uygulamaları için sunucu tarafı komut dosyası
Go	Büyük dağıtılmış, birbirine bağlı sistemler
Swift	Modern dil özelliklerine sahip iOS geliştirme

Yükseltürk ve Altıok (2018) blok tabanlı programlamanın (BTP) görsel araçlar içerisinde en fazla tercih edilen tür olarak görüldüğünü ve hazır blok şekillerin “sürükle-bırak” şeklinde programların oluşturulmasını sağladığını ifade etmişlerdir. BTP doğal dile yakın

olduğundan dolayı programlamaya yeni başlayanlar için programlama öğretiminde kolaylık sağlayacağı varsayılmaktadır (Chao, 2016; Lye ve Koh, 2018). Armoni, Meerbaum-Salant ve Ben-Ari (2015) ortaokulda nesnel programlama yerine Scratch öğrenen öğrencilerin programlama bilgisi ve deneyiminin daha ileriki dönemlerde öğrenmeyi büyük ölçüde kolaylaştırdığını belirtmişlerdir. Hendrix ve Weeks (2018) benzer şekilde BTP'nin yeni başlayanlar için nesne yönelimli programlama dillerinden daha uygun olduğunu belirtmiştir. Lye ve Koh (2018) BTP ortamlarının Scratch, Alice, Lego Mindstorms ve Blockly gibi benzer özelliklere sahip olduğunu söylemiştir.

Özmen ve Altun (2014) 7 adet BTP'yi (Tynker, Kodable, Scratch Junior, Light bot, MIT App Inventor ve Scratch); programlama ortamı, internet adresi, uygulamayı geliştirenler, geliştirildiği yıl ve hangi yaş aralığındaki kullanıcılara uygun olduğuna göre incelemiş ve bulgularını Tablo 2'deki gibi özetlemiştir.

Tablo 2. Blok tabanlı görsel programlama ortamları (Özmen ve Altun, 2014)

Ortam	Adresi	Geliştirenler	Yıl	Düzey
Tynker	tynker.com	Tynker şirketi	2013	7 yaş ve üzeri
Kodable	kodable.com	Louisville Üniversitesi'nden Grechen Huebner ve Jon Mattingly	2012	5 yaş ve üzeri (5. sınıfa kadar)
Scratch Junior	scratchjr.org	MIT Medya Laboratuvarı, Tufts Üniversitesi ve Playful şirketinin ortak projesi	2014	5-7 yaş arası
Light bot	lightbot.com	Waterloo Üniversitesi'nden Daniel Yaroslavski	2008	4-8 yaş ve 9+
Alice	alice.org	Carnegie Mellon Üniversitesi	1998	8 yaş ve üzeri
MIT App Inventor	appinventor.mit.edu	Google ve Massachusetts Teknoloji Enstitüsü (MIT) işbirliğiyle	2012	K-12 ve üzeri
Scratch	scratch.mit.edu	MIT Medya Lab'ında yer alan Lifelong Kindergarten grubunun bir projesi	2003	8 yaş ve üzeri

Günümüzde programlama dillerinin özellikleri çok gelişmiş ve bunların yanında birçok yeni programlama dili ortaya çıkmıştır. En iyi programlama dili hangisidir sorunun cevabı, programlama diliyle ne yapmak istediğimize bağlı olarak değişmektedir. En iyi programlama dili; program yazacağımız konuya bağlı, bildiğimiz ve aynı zamanda lisanslı derleyicisine sahip olduğumuz dil olarak ifade edilmektedir (Weinberg, 2011).

2.1.2. Programlama öğretimi

Yeni bir şey öğrenmenin en iyi yolu baştan başlamaktır. Herhangi bir programlama dilini öğrenmenin en iyi yolu da işe en baştan ve en basit örnek kodları çalıştırarak başlamaktır. Ayrıca örnek bir programın nasıl çalıştığını ve ne yaptığını anladıktan sonra, o mutlaka değiştirilmeli, ona yeni komutlar eklenmeli, var olan komutlar silinmeli ve bunların yerine yenileri yazılmalıdır. Bu sırada ne olduğu anlaşılmalı çalışılmalıdır. Programlar değiştirilmeli ve geliştirilmeli yani, onlarla oynanmalıdır (Weinberg, 2011). Programlama öğretiminde başarıyı etkileyen diğer bir etken ise hangi programlama dilinin öğretildiğidir (Brusilovsky, Calabrese, Hvorecky, Kouchnirenko ve Miller, 1997).

Programlama derslerinin çoğu bilgisayar laboratuvarında dersin eğitmeni tarafından önce düz anlatım şeklinde, devamında ise uygulamaya yönelik etkinlikler yoluyla öğretilmektedir. Her ne kadar kişilerin kendi imkanları dahilinde programlama dillerini öğrenme şansları olsa da genellikle öğrenme aşamasında ihtiyaç duydukları donanım ve yazılım gereksinimleri ile programlama dillerini öğrenmek için gerekli ön yeterliliklerinin olmaması nedeniyle programlama dilleri, sınıf ve laboratuvar ortamlarında daha kalıcı ve daha etkili bir şekilde öğrenilmektedir (Ersoy vd., 2011).

Yazılan programın kolay okunabilmesi için, birçok programcı tarafından titizlikle uyulan bazı basit kurallar vardır. Bunlardan bazıları:

- Her satırda bir ifade bulunması iyi olur.
- Bir bloğun başını ve sonunu belirten {} küme parantezleri göze batacak şekilde yerleştirilmelidir.
- Bir fonksiyonun argümanlarını sınırlayan parantezler aynı satırda olmalıdır.
- Bir fonksiyonun ismi ile parantezler arasında boşluk olmamalıdır.
- Bir ifadenin içindeki deyimler arasında boşluk bulunmalıdır.

- Aynı blok içindeki ifadeler aynı hizada olmalıdır.

Yukarıda sıralanan kurallara uyulursa bir süre sonra alışma ve yarar görme gerçekleşeceği aksi halde, uzun bir süre geçtikten sonra, kendi yazdığımız programı bile çözmekte zorlanacağımız sonucuna varılmaktadır (Weinberg, 2011).

Programlama öğretimi; soyutlama, genelleme, aktarım ve eleştirel düşünme gibi becerileri gerektirir (Gomes ve Mendes, 2007; Gundurao, Manjunath ve Nachappa, 2010). Bu konudaki deneyimler, problemin genel olarak, başlangıç öğrenme aşamasında, öğrencilerin problemleri çözmek için kontrol yapıları gibi belirli soyut programlama kavramlarını anlamadıklarından kaynaklandığını göstermiştir (Gomes ve Mendes, 2007). Programlamayı öğrenmenin en iyi yolunun, programları sıfırdan yazmak olduğu ifade edilmektedir. Ancak, tam programları incelemek ve test etmek, öğrencilerin programların nasıl çalıştığını anlamalarına yardımcı olabilmektedir. Ayrıca, bazı problemleri çözmek için kullanılan stratejileri analiz etmek, öğrenmenin ilk aşamalarında öğrenmeye destek sağlamaktadır. Bir başka yararlı etkinlik, genellikle öğrenciler tarafından yapılan mantıksal hataları içeren programları analiz etmektir. Tamamlanmamış programların tamamlanması, öğrencilerin başlangıçtan itibaren tüm programları yazmasını beklemek yerine, başlangıç aşamalarında yararlı bir etkinlik olabilir (Gomes ve Mendes, 2007).

Programcılardan bazıları karışık ve farklı biçimlerde program yazmanın övünülecek bir durum olduğunu düşünmektedirler. Oysaki bu övünülecek değil zayıf bir programcılık örneğidir. Ayrıca bu durum programların anlaşılmasında, okunmasında, test edilmesinde, hataların ayıklanmasında ve değişen durumlara uyum sağlamasında zorluklara yol açmaktadır. Programlama ile yeni tanışan kişiler için programların gayet açık olması temel hedef olarak belirtilmiştir (Deitel ve Deitel, 2008). Gomes ve Mendes'e (2007) göre programlama, yüksek düzeyde bir soyutlama gerektirir. Öğrencilerin farklı problemlerdeki kalıpları tanıması ve genelleştirme becerilerini geliştirmeyi öğrenmesi önemlidir. Hesaplamalı modelin, öğrencilerin soyutlama kapasitesinin kademeli gelişimini desteklediğini ve yeni durumlarını önceki deneyimleriyle ilişkilendirmelerine yardımcı olduğunu düşünmektedirler.

2.1.3. Programlama öğretiminin önemi

Yazılımın bir gider olarak değil, ülkelerin kalkınması için çok önemli bir yatırım olduğu görülmektedir. Yazılım alanında gelişmiş ve gelişmekte olan çoğu ülkeler yazılım için harcadıkları giderleri yatırım olarak sınıflandırmaktadır (Alican, 2006). Eğitim alanında son zamanlarda bütün dünyada, özellikle de gelişmiş ülkelerde kodlama eğitimi en önemli konular arasında görülmektedir. Erdem (2017), kodlama eğitimine yönelik çalışmaları sadece gelişmiş ülkeler değil, önde gelen yazılım şirketleri de yürüttüğünü ve bunun en önemli nedenlerinden birisinin; programlama veya kod yazmanın öğrenciler arasında son yıllarda hızla yayılan ve iyi bir kariyer vaat eden bir alan olarak görülmesi olduğunu belirtmiştir. Ayrıca programlama, yaşamın her alanına yayılmış olan bilgisayar teknolojilerinin üretim odaklı olarak kullanılmasında önemli bir yeterlilik alanı olarak görülmektedir. Bu bağlamda ülkeler eğitim programlarına mümkün olduğunca erken yıllarda kodlama öğretimini koyarak çocukların algoritmik ve üst düzey düşünme becerilerini desteklemek ve kendi çaplarında bilgisayar yazılımları üretmelerini istemektedir. Aytekin vd. (2018) kodlama eğitimlerinin mümkün olduğunca erken yaşlarda eğitim sistemine dâhil olması gerektiğini savunmaktadır. Nüfusunun çoğunluğu genç olan ülkemizin, yazılıma öncelik vermesi ve eğitime entegre etmesinin çok kritik olduğunu belirtmiştir. Günümüzde çocuklar, teknolojiyle doğdukları andan itibaren tanışmaktadır. Bu durumu olumlu ve yararlı hale getirmek için, çocuklarımızın teknolojiyi daha doğru ve etkili kullanmalarını sağlamak için küçük yaştan itibaren kodlama eğitimi verilmesi gerektiği düşünülmektedir (Aytekin vd., 2018; Demirer ve Sak, 2016).

Bilgiyi tüketen değil üretmek zorunda olan ülkemizin bilgi toplumu olabilmesi için yazılım sektörünü geliştirmek için adımlar atması gerekmektedir (Alican, 2006). Dünyaca ünlü yazılım şirketi Systems Analysis and Program Development (SAP) Türkiye'deki müşterilerine kendi yazılımı üzerinden İran ile ticaret yapılmamasını isteyen bir uyarı göndermiştir. Yerli yazılımın önemi bu olay ile daha da artmıştır (Türk Bilişim Derneği [TBD], 2018).

Programlama karmaşık problemlere basit parçaları bir araya getirerek çözümler üretmeyi sağlayan zihinsel bir etkinliktir. Bu yönüyle programlama çocukların parça-bütün ilişkilerini öğrenmelerine, detaylara dikkat etmelerine, düşüncelerini organize ve ifade

etmelerine ve zekâlarının gelişmesine katkıda bulunacaktır (Akyüz, 2018). Yeni programlama dilleri öğrenmek, bazen farklı şekillerde problemleri düşünmeyi ve çözmeyi sağlayacaktır. Yeni dillerle tanışma her zaman kullandığımız dillerde daha iyi kod yazmamıza yardımcı olabilir (Toal vd., 2016). Programlama dillerinin incelenmesi program yazarken farklı yapıları kullanma kapasitemizi arttırması, yeni projeler için dilleri daha akıllıca seçmemizi ve yeni dilleri öğrenmemizi kolaylaştırdığı için önemlidir (Sebesta, 2016). Bir programlama dilini çok iyi öğrenen bir kişi diğer bilmediği başka bir dili çok kısa bir sürede kolayca öğrenecektir (Yaşar, 2012).

Aspinal (2015) kodlama öğrenmek için 10 neden belirlemiştir:

1. Kodlama, öğrencilerin sadece tüketmek değil, içerik oluşturmasını sağlar.
2. Kodlama, öğrencileri güçlendirir ve kendilerini gerçekten havalı yollarla ifade edebilecekleri araçları sunar.
3. Kodlama, oyunlar ve animasyonlarla hikaye anlatmayı öğretir.
4. Kodlama, öğrencilerin risk alabileceği ve güvenli bir şekilde hata yapabilecekleri bir yerdir.
5. Kodlama kapsamlıdır ve kendine güven sağlar.
6. Kodlama, matematiğin birçok ilkesini destekler.
7. Kodlama problem çözme ve eleştirel/analitik düşünme becerilerini öğretir.
8. Kodlama yeni bir okuryazarlık türüdür ve gelecekteki işlerin büyük bir kısmı olacaktır.
9. Kodlama, takım çalışması ve işbirliği becerileri geliştirir.
10. Kodlama insanlığa yardımcı olabilir.

2.1.4. Programlama öğretiminde yaşanan zorluklar

Bilgisayar programcılığının öğretilmesi ve öğrenilmesi, Bilgisayar Eğitimi alanında büyük zorluklardan biri olarak belirtilmektedir (Renumol, Jayaprakash ve Janakiram, 2009). Birçok öğrenci konunun doğası gereği öğrenme problemleri yaşamaktadır. Karmaşık ve soyut yapıda olan programlama dillerinin öğrenilmesi ve öğretilmesinde zorluklar yaşanması doğaldır (Renumol vd., 2009) ve bununla birlikte yeterli kaynak olmaması ve öğrencilerin kişisel gelişim eksikliği problemleri arttırmaktadır (Demir, 2015). Ayrıca öğrenci gruplarının büyük ve heterojen olması eğitimin kalitesini

düşürmektedir (Renumol vd., 2009). Etkili öğrenme için, kullanıcıların motivasyon düzeyleri ve sosyal varlığı önemlidir (Yılmaz, Topu, Göktaş ve Çoban, 2013). Programlama sadece soyut kavramlardan dolayı zor değildir. Öğrencilerin program yapımına ilişkin farklı konularda da problemleri vardır. Öğrencilerin kendi başlarına programlamayı öğrenmeleri önemlidir. Dikkatlice tasarlanmış materyaller ve yaklaşımlar ile öğretmenler, öğrencilere bilgi ve beceri inşasını yönlendirebileceğini ifade etmiştir (Lahtinen vd., 2005).

Lahtinen vd. (2005) acemi programcıların tipik olarak programların yüzey bilgisi ile sınırlı olduğunu ve anlamlı program yapıları kullanmaktan ziyade genellikle “satır satır” programlamaya yaklaştıklarını belirtmiştir. Acemilerin bilgisi bağlamsal olmaya eğilimlidir ve aynı zamanda elde ettikleri bilgileri yeterince uygulayamazlar. Bireysel ifadelerin sözdizimini ve anlambilimini bilebilirler, ancak bunları geçerli programlarla nasıl birleştireceklerini bilmemektedir (Lahtinen vd., 2005).

Bir programlama dilini öğrenirken zorlanılan konuların en başında yer alan husus, öğelerin hiç birinin tek başına mevcut olmadığı gerçeğidir. Bu şekilde birbiriyle bağıntılı olmak, programlama dilinin diğer özelliklerine dokunmadan tek özelliğini anlatmayı zorlaştırır. Genellikle, bir özelliği ele alırken bir başkası hakkında ön bilgi sahibi olmak gerekmektedir (Herbert, 2010).

Gomes ve Mendes (2007) programlamayı, öğrenilmesi ve öğretilmesi için çaba ve özel bir yaklaşım gerektiren çok karmaşık bir konu olarak tanımlamışlardır. İyi bir programcı olmak için; bir öğrenci, bir programlama dilinin sözdizimini bilmenin ötesine geçen bir dizi yetenek kazanmalıdır. Öğrenciler, hızlı bir şekilde olası bir çözüm bulamıyorlarsa problem çözmekten vazgeçerler. Genellikle programlama problemlerini çözmek, çaba ve sebat gerektirir. Ancak, herhangi bir zorlukla karşılaştığınızda, çoğu öğrenci problemi çözmeye çalışmak yerine çözümü bir meslektaşına sormayı ya da sadece pes etmeyi tercih eder. Çözümü buldukları zaman ise öğrenmenin daha etkili olacağı sonucuna ulaşmıştır. (Gomes ve Mendes, 2007).

Gomes ve Mendes (2007) öğrencilerin tipik programlama derslerine girmeden önce problem çözmeyi amaçlayan en az bir dersi almaları gerektiğini düşünmektedir. Ayrıca öğrencilere, problemleri çözme konusunda belirli yönleri öğreterek, onlara yardımcı

olacak ipuçları vermenin faydalı olacağını belirtmiştir. Öğrencilere çok fazla pratik yapma fırsatı vermek ve bir etkinlik gerçekleştirdikten sonra, öğrencinin çözümü tam olarak anladığından emin olmak için ek sorular veya faaliyetler sağlaması da önemlidir. Bu, öğrencilerin çözümlerini ve bu çözümleri nasıl geliştirilebileceğini düşünmelerini sağlayacaktır (Gomes ve Mendes, 2007).

Costa ve Aparicio (2014) bilgisayar programlama öğrenme sistemi başarısını ölçmek için bir değerlendirme modeli önerdikleri çalışmada çeşitli çalışmalarda programlama öğretimi zorluklarını özetleyerek tablo haline getirmiştir (Tablo 3). Araştırmalarda ulaşılan sonuçlardan yola çıkarak karşılaşılan çoğu zorluğun beceri eksikliğinden ve bilgiyi bütünleştirmedeki zorluklardan (program tasarımı ve program anlama) kaynaklandığı sonucuna ulaşılmıştır.

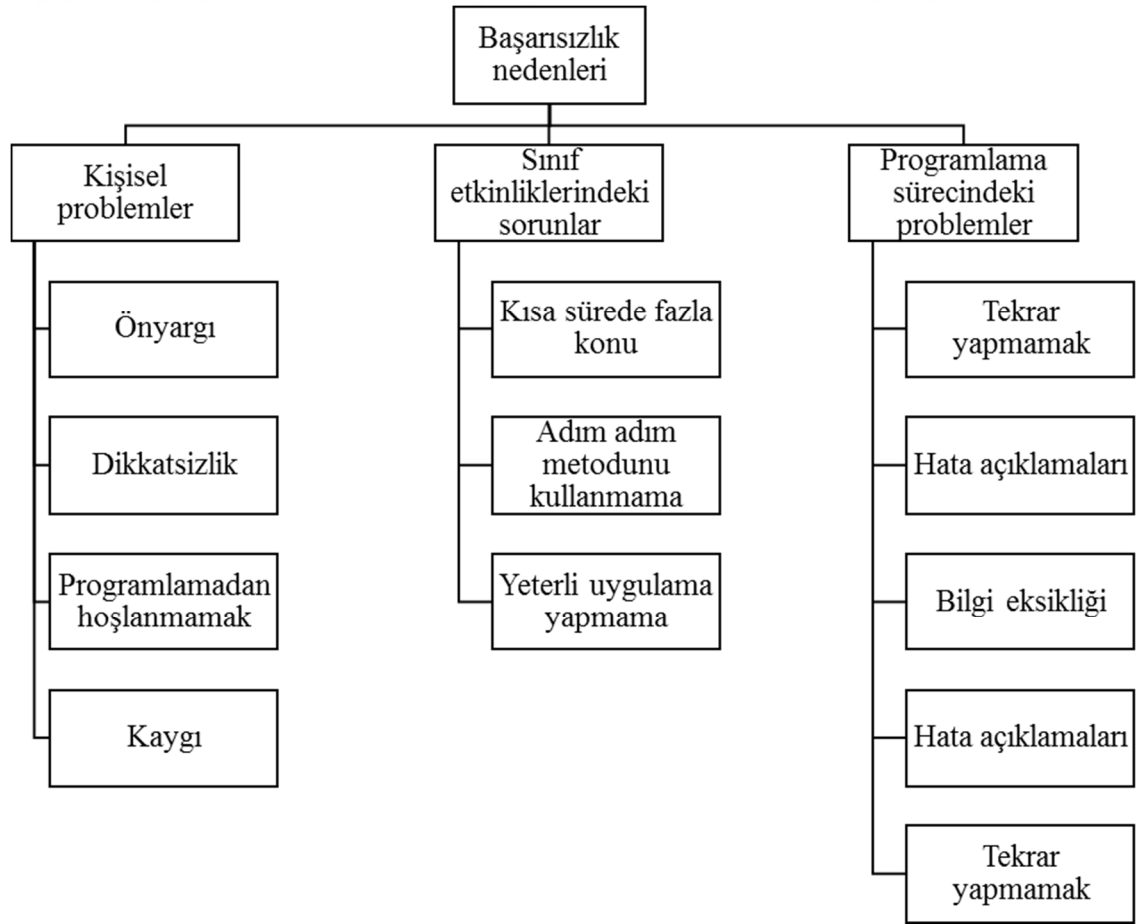
Tablo 3. Programlama öğretimi zorlukları (Costa ve Aparicio, 2014)

	Tecrübe	Yenilik	Beceriler	Program Tasarımı	Program Anlama	Programlama Dili Seçimi	Zamanlama Dersi Yapısı	Farklı Yetenekler	Öğrenmeye Karşı Tutum	Bilişsel Faktörler
Jenkins, 2002			√	√	√	√	√			√
Robins vd., 2003		√		√	√	√		√	√	√
Lahtinen vd., 2005				√	√	√		√	√	
Sajaniem & Hu, 2006				√	√	√		√	√	
Radosevic vd., 2009				√	√	√		√	√	√
Renumol vd., 2009				√	√	√		√	√	√
Cummmings, 2010			√	√	√	√	√	√	√	
Kordaki, 2010		√		√	√	√	√	√	√	

Birçok öğrenci yeterli matematiksel ve mantıksal bilgiye sahip değildir. Matematiksel bilginin programlama için çok önemli olduğunu ve programlama becerileri ile matematik arasında anlamlı bir ilişki olduğunu gösteren çalışmalar mevcuttur (Gomes ve Mendes,

2007). Devletin ve özel sektörün bilim ve teknolojiye yaptığı yatırımlara rağmen çocukların bu alanda istenilen seviyeye gelemediği görülmekte ve bunun nedeni olarak teknolojiyi nasıl ve hangi amaç için kullanacaklarının farkında olmamaları ve teknolojiyi üretmek yerine çoğunlukla eğlence amacıyla kullanmaları gösterilmektedir (Coşar, 2013'den akt. Demirel ve Sak, 2016).

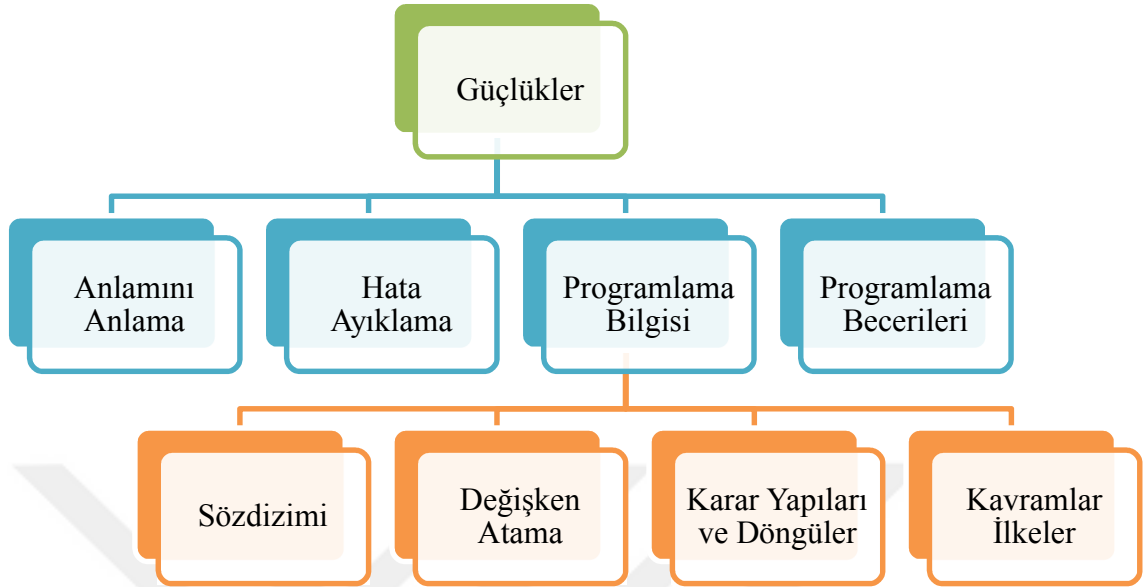
Özmen ve Altun'un (2014) üniversite öğrencilerinin programlamada karşılaştıkları güçlükleri ve engelleri araştırdıkları çalışmalarında programlama deneyimleri başarısızlık nedenlerini 3 başlık altında Şekil 3'de sunulmuştur. Bunlar kişisel problemler, sınıf etkinliklerindeki sorunlar ve programlama sürecindeki problemlerdir.



Şekil 2. Programlama deneyimleri başarısızlık nedenleri (Özmen ve Altun, 2014)

Özmen ve Altun'un (2014) güçlükler ile ilgili ulaştıkları sonuçları Şekil 3'de görülmektedir. Araştırma sonucunda karşılaşılan güçlükleri; anlamını anlama, hata ayıklama, programlama bilgisi ve programlama becerileri olarak dört başlık altında toplamıştır. Bu başlıklar içerisinde de programlama bilgisi altında güçlükleri belirtirken

sözdizimi, değişken atama, karar döngüleri ile yapılar, ilke ve kavramlar olarak belirtmiştir.



Şekil 3. Programlamada karşılaşılan zorluklar (Özmen ve Altun, 2014)

2.1.5. Türkiye ve dünyada programlama öğretimi

Programlama eğitiminin, ülkenin ilerlemesine yapacağı katkılar göz ardı edilemez. Programlama eğitimi, sorgulayan ve üreten bir nesil yetiştirken aynı zamanda programlama alanında eleman ihtiyacını da gidereceği düşünülmektedir. Ayrıca ülkelerin bu eğitimler sayesinde gelişmişlik düzeyleri olumlu yönde etkilenecek teknolojik olarak ve yazılım alanında üst sıralarda yer alacaklarını söylemek mümkündür. Bu durumun farkında olan gelişmiş ülkeler, eğitim müfredatlarında değişikliklere gitmişler (Bers, Flannery, Kazakoff ve Sullivan, 2014; Lee, Martin ve Apone, 2014) ve küçük yaşlardan itibaren öğrencilere programlama eğitimi vermeye başlamışlardır (Demirer ve Sak, 2016).

Kodlama saati (HourOfCode), Türkiye'nin de içerisinde bulunduğu 180'den fazla ülkede milyonlarca öğrenciye ulaşmış küresel bir hareket olarak nitelendirilmektedir. Code.org tarafından organize edilen bu etkinliğe Microsoft, Apple, Amazon gibi şirketler de ortaklık ederek katkıda bulunmaktadır. Kod Saatleri, her yıl 3-9 Aralık tarihleri arasında gerçekleştirilmekte fakat istenilen zamanda yıl boyunca uygulanabilmektedir. Bu etkinlikte bir saatlik bilgisayar bilimine giriş yaparak, kodlamanın temellerini

öğrenilebilir ve sırrı çözülebilir. Herkes her yerde 45'in üzerinde farklı dilde bir kod saati etkinliği düzenleyebilir. Yaşları 4 ile 104 arasında değişen kullanıcılar bulunmaktadır.

2018 yılı içerisinde kodlama saati olarak 220 bin üzerinde etkinlik oluşturulmuş ve ülkemizde gerçekleştirilen etkinliklerin 2709 tanesi Şekil 4'te gösterilmiştir. Ülkemizde gerçekleştirilen bu etkinliklerin büyükşehirlerde yoğunlaştığı ve doğu bölgelerinde etkinliklerin çok az olduğu görülmektedir.



Şekil 4. Kodlama saati Türkiye etkinlikleri (HourOfCode, 2018)

2.1.5.1. Türkiye’de programlama öğretimi

Türkiye’de ilköğretim düzeyinde programlama eğitime, 2006 yılında 7. ve 8. sınıf Bilgisayar dersi içerisindeki Algoritma ve Programlama konusunda başlanmıştır. Bu ders için ayrılan süre haftada bir saat olduğu için planlanan uygulamaların yapılmasının mümkün olmadığı görülmüştür (Seferoğlu, 2007). 2012 yılına kadar bu şekilde devam ettikten sonra ilköğretim 8. sınıf seçmeli dersi Bilişim Teknolojileri ve Yazılım (BTY) öğretim planı kapsamında yıl boyunca öğrencilerin programlama eğitimi almaları kararlaştırılmıştır. Ortaöğretim de ise mesleki ve teknik liseler bilgisayar bölümlerinde programlama öğretimi ilk kuruldukları yıllardan itibaren yapılmaktadır. Üniversite düzeyinde ise Bilgisayar Mühendisliği, Yazılım Mühendisliği, Makine Mühendisliği, Elektrik – Elektronik Mühendisliği, Bilgisayar ve Öğretim Teknolojileri Öğretmenliği, Bilgisayar Kullanımı, Bilgisayar Programcılığı gibi bölümlerde yapılmaktadır (Akçay ve Çoklar, 2016).

Programlama öğretimi 5. ve 6. sınıf müfredatında BTY dersi içerisindeki problem çözme ve programlama ünitesi altında yer almaktadır (Tablo 4). Problem Çözme ve Programlama ünite başlığı altında; algoritma tasarımına ilişkin (arama, sıralama vb.) anlayış geliştirme; sözel ve görsel olarak ifade etme, problem çözmek için değişken, atama, sıralı mantık, karar yapısı, döngü ve fonksiyon yapılarını kullanma, problemleri çözmek için uygun programlama yaklaşımını seçme ve uygulama konusunda beceriler kazandırılması amaçlanmıştır.

Tablo 4. BTY dersi 5. sınıf problem çözme ve programlama ünitesi (MEB, 2018)

ÜNİTE ADI	KONU ADI	TOPLAM KAZANIM SAYILARI	SÜRE / DERS SAATİ	ORAN %
BİLİŞİM TEKNOLOJİLERİ	1. Bilişim Teknolojilerin Günlük Yaşamdaki Önemi 2. Bilgisayar Sistemleri 3. Dosya Yönetimi	12	6	8
ETİK VE GÜVENLİK	1. Etik Değerler 2. Dijital Vatandaşlık 3. Gizlilik ve Güvenlik	9	8	12
İLETİŞİM ARAŞTIRMA İŞBİRLİĞİ	1. Bilgisayar Ağları 2. Araştırma 3. İletişim Teknolojileri ve İş Birliği	12	8	11
ÜRÜN OLUŞTURMA	1. Görsel İşleme Programları 2. Kelime işlemci Programları 3. Sunu Programları	15	14	19
PROBLEM ÇÖZME VE PROGRAMLAMA	1. Problem Çözme Kavramları ve Yaklaşımları 2. Programlama	27	36	50
TOPLAM		75	72	100

Türkiye’de eğitim alanında programlama öğretimi 5. ve 6. sınıflarda haftada iki saat zorunlu olarak okutulan BTY dersi öğretim programında yer almaktadır. Öğretim programında öğrencilerin temel bilgisayar kullanımı ve programlama becerileri kazanmaları hedeflenmiştir. BYT dersi öğretim programının özel amaçları içerisinde

programlama ile ilgili özel amaçlar belirlenmiştir. Bunlar aşağıdaki şekilde sıralanmıştır (Milli Eğitim Bakanlığı [MEB], 2018, s. 10).

1. Problemleri çözmek için uygun programlama yaklaşımını seçerek uygulayabilmelerini,
2. Programlama konusunda teknik birikim oluşturmalarını,
3. Programlama dillerinden en az birini kullanabilmeleridir.

BTY dersi problem çözme ve programlama ünitesi kazanımları aşağıdaki şekilde belirlenmiştir (MEB, 2018, s. 14).

1. PROBLEM ÇÖZME VE PROGRAMLAMA

1.1. Problem Çözme Kavramları ve Yaklaşımları

- 1.1.1. Verileri toplayarak türlerine göre sınıflandırır.
- 1.1.2. Sabitleri ve değişkenleri problem çözümünde kullanır.
- 1.1.3. Bir problemi alt problemlere böler.
- 1.1.4. Temel fonksiyonları problem çözme sürecinde kullanır.
- 1.1.5. Problemin çözümü için bir algoritma geliştirir.
- 1.1.6. Bir algoritmanın çözümünü test eder.
- 1.1.7. Farklı algoritmaları inceleyerek en hızlı ve doğru çözümü seçer.
- 1.1.8. Hatalı bir algoritmayı doğru çalışacak biçimde düzenler.
- 1.1.9. Problemin çözümünü benzer problemler için geneller.
- 1.1.10. Matematik ve bilgisayar bilimi arasındaki ilişkiyi tartışır.

1.2. Programlama

- 1.2.1. Blok tabanlı programlama aracının ara yüzünü ve özelliklerini tanır.
- 1.2.2. Blok tabanlı programlama aracında sunulan bir programın işlevlerini açıklar.
- 1.2.3. Blok tabanlı programlama aracında sunulan bir programın hatalarını ayıklar.
- 1.2.4. Blok tabanlı programlama aracında sunulan bir programı verilen ölçütlere göre geliştirerek düzenler.
- 1.2.5. Doğrusal mantık yapısını içeren programlar oluşturur.
- 1.2.6. Doğrusal mantık yapısını içeren programları test ederek hatalarını ayıklar.
- 1.2.7. Karar yapısını içeren programlar oluşturur.
- 1.2.8. Karar yapısını içeren programları test ederek hatalarını ayıklar.
- 1.2.9. Çoklu karar yapıları içeren programlar oluşturur.

- 1.2.10.Çoklu karar yapısını içeren programları test ederek hatalarını ayıklar.
- 1.2.11.Döngü yapısını içeren programlar oluşturur.
- 1.2.12.Döngü yapısını içeren programları test ederek hatalarını ayıklar.
- 1.2.13.Bir algoritmayı uyarlamak için en uygun karar yapılarını seçer.
- 1.2.14.Farklı programlama yapılarını kullanarak karmaşık problemlere çözüm üretir.
- 1.2.15.Tüm programlama yapılarını içeren özgün bir proje oluşturur.

Lise ve dengi okullarda ise Bilgisayar Bilimi dersi öğretim programında programlama, robot programlama, web tabanlı programlama ve mobil programlama olmak üzere farklı bölümlerde yer almaktadır. Bilgisayar bilimi dersi öğretim programının özel amaçları içerisinde bu kategorilerde yer alan maddeler:

1. Algoritma tasarımına ilişkin anlayış geliştirerek sözel ve görsel olarak ifade etmelerini,
2. Problem çözmek için değişken, atama, sıralı mantık, karar yapısı, döngü ve fonksiyon yapılarını kullanmalarını,
3. Problemleri çözmek için uygun programlama yaklaşımını seçmelerini ve uygulamalarını,
4. Programlama konusunda teknik birikim oluşturmalarını,
5. Programlama dillerinden en az birini iyi düzeyde kullanmalarını,
6. Robot programlama konusunda temel bilgilerle donanmalarını,
7. Mobil programlama konusunda deneyim kazanmaları şeklinde sıralanmıştır (MEB, 2018, s. 11).

Bilişim Garaj Akademisi 2012 yılında bir grup akademisyen tarafından oluşturulan çevrimiçi bir hizmettir. 100 bine yaklaşan 5-16 yaş arası öğrenci topluluğuna “bilişimle üretim” konusunda eğitimler veren ülkemizin en büyük çevrimiçi eğitim platformlarından. Web tasarımı, kodlama, akıllı cihaz tasarımı ve programlama, üç boyutlu tasarım alanlarında beceri kazandıran eğitim platformudur. (Bilişim Garaj Akademisi, 2014).

Bilişim teknolojileri tüm dünyada hayatın her alanında yer almaktadır. Bundan dolayı devamlı yeni yazılımlara ihtiyaç duyulmaktadır. Bu ihtiyacı karşılayabilmek için çocuklara erken yaşlardan itibaren kodlama eğitimine ağırlık verilmesi gerekmektedir.

Son yıllarda Türkiye’de MEB bu konuda adımlar atmaya başlamıştır. İlgili akademisyenler, öğretmenler ve yazılımcılar ile birlikte çalışarak ilköğretim düzeyinde okutulan BTY dersi müfredatını kodlama eğitimiyle güncelleme çalışmaları yürütmekte ve bu bağlamda Google şirketi ile ortak "İlköğretimde Bilişim Teknolojileri ve Yazılım Dersi İçin İş Birliği Protokolü" imzalamıştır (MEB, 2017).

Microsoft Türkiye tarafından sosyal sorumluluk projesi olarak Açık Akademi adı altında çevrimiçi uygulama geliştirme okulu ücretsiz olarak kullanıcılara sunulmaktadır. Açık akademi, yazılıma ilgi duyan ve kendini bu alanda geliştirmek isteyen herkese açık bir platformdur. Bu platform sayesinde geliştirilmek istenen uygulama için gerekli eğitimler alınabilmekte ve sorular sorarak diğer öğrencilerden ya da Microsoft uzmanlarından yardım alınabilmektedir. Programlamanın temellerinden başlayıp daha sonra kullanıcının ilgi duyduğu alana göre mobil, masaüstü veya bulut uygulamaları geliştirebilmek için gerekli videolara ulaşılabilenkte ayrıca sanal sınıflarda öğrenilenler pekiştirilebilmektedir. Ayrıca Açık Akademi içerisinde özellikle çocukların basit şekilde kullanarak bilgisayar oyunları tasarlayabilmeleri için geliştirilmiş görsel ağırlıklı bir programlama dili olan Kodu yer almaktadır. Kodu eğitimleri Microsoft ile Doğa Okulları işbirliğiyle ve “Minik Parmaklar Geleceği Programlıyor Projesi” kapsamında QNB Finansbank’ın katkılarıyla geliştirilmiştir (Açık Akademi, 2018).

Milli kodlama eğitimi programı olan Hacker Can projesi, programlama öğrenmek isteyen herkes için geliştirilmiş yerli bir platformdur. Kullanıcıların kendi dilinde kodlamayı öğrenme ve Türkçe kod yazarak yazılım geliştirme becerisini sağlamak için hazırlanan Türkiye'nin ilk eğitim platformudur. MEB tarafından BTY dersi öğretim müfredatına girmeyi başaran ilk yerli yazılım platformudur. Bu platformda öğrenciler kendi seviyesine ve yaş grubuna göre programlama alanında kendini geliştirebilir. Türkçe ve İngilizce olarak kullanılabilen platform öğrencilere algoritma, problem çözme yeteneği ve programlama (gerçek kod yazma) eğitimi vermeyi hedeflemektedir. Oyun üzerine kurgulanmış 6 yaş üstü her öğrencinin yararlanabileceği keyifli bir platformdur. Öğretmenler için gerekli akademik materyalleri (özel paneller, öğretim planları, sunumlar) sunan Hacker Can; eğitimlerin sonunda yabancı karakter ve noktalama işaretlerinin az olduğu CoffeeScript modelinde bir dil öğretir. Öğrenciler bu eğitimler sayesinde diğer dillere adapte olmakta zorluk çekmezler. Web ve masaüstü yazılımları, mobil uygulamalar ve oyun geliştirebilirler (Hacker Can, 2018).

Kod Adı 2023 projesi, Gençlik Hizmetleri Genel Müdürlüğü tarafından tüm Türkiye’de uygulanan bilişim eğitimlerini ve seminerleri kapsamaktadır. Proje başlangıcında gençlik liderleri ve eğitmen adaylarına algoritma, web ve mobil uygulama geliştirme, elektronik ve robotik eğitimleri verilmiştir. Ayrıca çeşitli seminer programları yapılmıştır. Bu seminerlerde; erken yaşta kodlama eğitimi, dijital bilgi güvenliği, etkili sosyal medya yönetimi, siber saldırılar, milli yazılımlar ve dijital pazarlama üzerine seminer programları gerçekleştirilmiştir. Seçilen başarılı gençlere uzmanlar tarafından siber güvenlik ve ileri düzey yazılım geliştirme eğitimleri verilmiştir. Bu eğitimleri başarıyla bitiren gençlere sertifika verilmiştir ve ayrıca referans mektubu yazılarak TÜBİTAK ve BİLSEM gibi kurumlara bu gençler tavsiye edilmiştir. Kod Adı 2023 projesi gençlik merkezleri vasıtasıyla bütün il merkezlerinde başlatılmıştır. 2018 yılı sonuna kadar daha da yaygınlaştırılarak tüm gençlik merkezlerinde başlatılması planlanmıştır. Her yıl 40 bin üzerinde öğrencinin eğitim alması hedeflenmiştir (Kod Adı 2023, 2018).

Tablo 5’de proje geliştirme alanında faaliyet gösteren dünyaca ünlü Github sitesinin, konumu Türkiye olan 12 binin üzerindeki geliştiricisinin oluşturduğu 35 binden fazla repo üzerinde gerçekleştirdiği 2017 yılı istatistiklerine göre yayınladığı Türkiye’de kullanılan programlama dilleri sıralaması listelenmiştir. Github üzerinde ki Türkiye konumunda bulunan geliştiriciler arasında en yaygın kullanılan dil olarak Java görülmektedir. Javascript, Python ve C# dilleri de yaygın kullanılan diğer diller olarak sıralanmaktadır.

Tablo 5. Türkiye’de kullanılan programlama dilleri (Github, 2017)

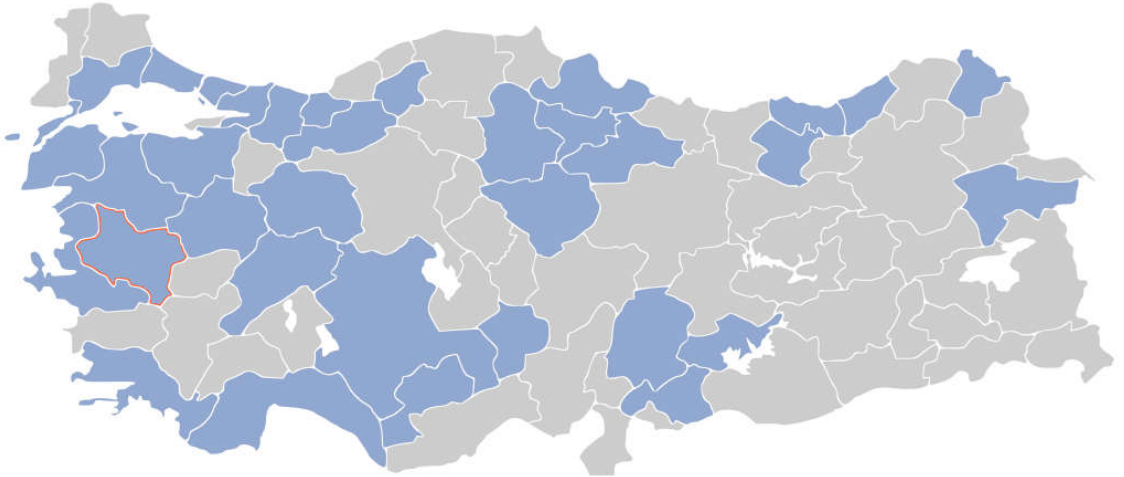
Sıra	Dil	Repo	%
1	Java	5574	21,7
2	JavaScript	4574	17,8
3	Python	2493	9,7
4	C#	2438	9,5
5	PHP	2369	9,2
6	HTML	1745	6,8
7	CSS	1337	5,2
8	Ruby	1062	4,1
9	C+ +	1060	4,1
10	C	925	3,6

Github sitesinin kurulduğu 2008 yılından 2017 yılına kadar konumu Türkiye olan geliştiricilerin illere göre dağılımı Şekil 5’de harita üzerinde görülmektedir. Java dili 15 şehirde en fazla kullanılan dil olmuştur. Onu sırasıyla PHP 10, C# 6, Ruby 6, Javascript 5, CSS 5, Python 4, HTML 2, C 2, C++ 1 şehirde üstün olarak görülmektedir. 24 şehirden ise katılan bulunmamaktadır.



Şekil 5. Github 2008-2017 Türkiye il istatistikleri

Türkiye’de kodlama üzerine projeler gerçekleştiren iller Şekil 6’ da Türkiye Haritası üzerinde işaretlenmiştir. Bunlara örnek olarak Kodla(Ma)nisa, Afyon Kodluyor, KodlaRize, KodlaAntep, Kodla Maraş ve Amasya Kodlama Projesi (MİSKOD) projeleri gösterilebilir.



Şekil 6. Türkiye’de kodlama projesi yapan iller (Kodla(Ma)nisa, 2018)

Kodla(Ma)nisa projesi; Manisa Valiliği tarafından 2015 yılında başlanan “Bilişim Teknolojileri Eğitiminde Değişim” Projesi olarak lanse edilmiştir. Bu proje Manisa ilinin 17 ilçesinde, 18 Kodlama Atölyesi, 292 Okul, 300 Öğretmen ile birlikte, toplam 81.505 öğrenciye, BTY derslerinde temel kodlama eğitimi verilerek gerçekleştirilmiştir. Projenin amacı bilgisayar kullanım becerilerini öğrencilere kazandırıp, tüketen değil üreten bireyler olarak yetiştirmek ve kodlamayı öğretmek sadece kullanan değil yeni uygulamalar üretebilen nesil oluşturmaktır. KodlaRize projesi ise Rize valiliği tarafından 2016-2017 eğitim öğretim yılında ortaokul öğrencilerinin analitik düşünme, sebep sonuç ilişkisi kurma ve kod okuryazarlığı bilincinin gelişmesi için başlattığı bir proje olarak görülmektedir (Gültepe, 2018). Afyon Kodluyor Projesi, il milli eğitim müdürlüğü tarafından Bilişim Teknolojileri Eğitiminde Değişim adı altında 2018 yılında başlatılan bir projedir. Bu projede okul öncesi ve ilkokullarda algoritma ve kodlama, ortaokullarda ise kodlama, 3 boyutlu tasarım ve robotik konularında eğitimler verilmektedir. 6 yaşından itibaren bilgisayarla tanışan bilgisayar oyunları ile vakit geçiren çocukların; eğlenerek öğrenmesi ve bilgisayar karşısında tüketici konumundan üretici konumuna geçmesi için ve tüm öğrencilerin ayırım olmadan programlama ile tanışması için oluşturulmuş bir projedir (Afyon Kodluyor, 2018).

Kodlama Festivali 2018 adında, “Kodlama okuryazarlığı” kavramını yaygınlaştırmayı ve kodlamaya gönül veren kitleleri bir araya getirmeyi amaçlayan kapsamlı bir kodlama organizasyonu ülkemizde düzenlenmektedir. Festivalin yapılma amacı; günümüzde hayatımızın her alanına bilişim teknolojilerinin dâhil olduğu, geleceği kodlamanın şekillendireceği öngörüsü olarak belirtilmiştir. Bundan dolayı, her yaştan bireyin kodlama eğitimine erişebilmesi ve kodlamanın öneminin tüm kitlelerce kavranabilmesi için çalışmalarını sürdürmektedir (Kodlama Festivali, 2018).

Yükseköğretim Kurulu (YÖK) Başkanı, Türkiye İş Kurumu (İŞKUR) desteğiyle YÖK’ün üniversitelerdeki ilgili öğretim elemanları tarafından üniversite mezunlarına ya da halen üniversite de okuyan gençlerimize ücretsiz kodlama eğitimleri verileceğini söylemiştir. Ayrıca kodlama eğitiminin giderek önem kazandığını ve bundan dolayı üniversitelerde birçok farklı bölümde programlama dersinin müfredata dâhil edilerek başka disiplinlere de yaygınlaştırılmaktadır (Yükseköğretim Kurulu [YÖK], 2018).

Lisans düzeyinde programlama öğretimi 1991 yılında MEB tarafından bilgisayar formatör öğretmenliği adı altında eğitim fakülteleri ve bilgisayar mühendisliği bölümlerinde başlatılmıştır (Akpınar ve Altun, 2014). Bunun yeterli olmayacağı düşünülerek daha sonra MEB ortaöğretimde bilgisayar öğretmenliği bölümünün açılmasını teklif etmiş ve Orta Doğu Teknik Üniversitesi'nde (ODTÜ) Bilgisayar Öğretmenliği Bölümü, fen bilimleri eğitimi bünyesinde açılmıştır. Eğitim fakülteleri 1997 yılında yapılanmaya gitmiş ve bu yapılanma neticesinde eğitim fakültelerinde Bilgisayar ve Öğretim Teknolojileri Eğitimi adı altında eğitim vermeye başlanmıştır (Akpınar ve Altun, 2014).

2.1.5.2. Türkiye’de programlama öğretimi üzerine yapılan çalışmalar

Koçak (2006) tez çalışmasında iki yıllık meslek yüksekokulu Bilgisayar Teknolojisi ve Programlama bölümündeki akademisyenler, mezun olacak olan öğrenciler ve sektör temsilcilerinin görüşleri ışığında sektörün sorunlarını saptamaya çalışmıştır. Araştırmasının örneklemini 6 meslek yüksekokulunda görevli toplam 22 akademisyen, mezun olabilecek durumdaki toplam 214 öğrenci ile bilişim sektöründe faaliyet gösteren işletme sahipleri oluşturmuştur. Çalışma sonunda çok çeşitli ders içeriklerinin seçiminin akademisyenlere bırakılması, ön lisans kurumları arasında karmaşıklığa sebep olmakta ve öğrenciler arasında haksızlığa yol açtığı sonucuna ulaşmıştır. Bunun göstergesi olarak anket formunu cevaplayan öğrencilerden birisinin “Biz bir programlama dili öğrenirken diğer sınıf başka bir programlama dili görmektedir.” ifadesini belirtmiştir. Akademisyenler ve mezun olacak durumdaki öğrencilerin görüşleri sonucunda Bilgisayar Teknolojisi ve Programlama bölümünde sektörün ihtiyaç duyduğu alana yönelik derslerin yetersiz olduğu ve akademisyen sayısının yetersiz olmasından dolayı seçmeli dersleri sektör istekleri doğrultusunda ayarlanamadığı sonucuna ulaşmıştır. Sektörün ihtiyacı olan alandaki ders sayısının az olmasının ve öğrencilere seçmeli derslerde belirli derslerin seçilmesinin zorunlu tutulmasının öğrencilerin ilgi alanlarına yönelik kabiliyetlerini okul ortamında geliştiremeyeceğini belirtmiştir. Sektörün ihtiyaçları ile öğrencilerin eğitim hayatında kazandıkları deneyimler arasında farklılık olmasının başarının düşmesine ve mezun olan öğrenciler de iş bulma da zorluklar yaşanabileceğini söylemiştir. Bundan dolayı müfredatta sektörün ihtiyaçlarına uygun derslerin artırılmasını, bunun için gereken düzenlemelerin yapılmasının gerekliliğinden bahsetmiştir.

Mazman (2013) programlama performansını etkileyen bilişsel faktörlerle ilgili model geliştirmiştir. Bu model de performansın etki eden bireysel farklılıkların diğer demografik özelliklerin etkisiyle göstermiştir. Çalışmasında uzamsal beceri ve çalışma belleğini ele almış ve bunların programlama performansını yordama etkisini ölçmüş ve değişken sıralamasını önem durumuna göre açıklamıştır. Çalışmasında üç ayrı üniversitedeki Bilgisayar ve Öğretim Teknolojileri Eğitimi (BÖTE) bölümü 2. sınıf öğrencilerinden “Programlama Dilleri-I” dersine kayıtlı 129 öğrenci ile yürütmüştür. Analiz kısmında veri madenciliğine dayalı “Artırılmış Regresyon Ağaçları” ve “Rastgele Ormanlar” tekniklerini kullanmıştır. Bulgular bireylerin bilişsel becerileri ve herhangi bir konudaki başarısı arasındaki etkileşimin dinamik olabileceğini, yani bazı becerileri yüksek bireylerin, bazı becerilerinin düşük olabileceğini göstermiştir. Ön deneyim ve cinsiyetin programlama performansı üzerinde anlamlı bir etkisinin olmadığı görülmüştür. Programlama performansında bilişsel farklılıkların göz önüne alınması gerektiği vurgulanmıştır. Uzamsal becerilerin programlama performansının açıklanmasında önemli bir faktör olduğu bulunmuştur. Çalışma belleğinin de hem görsel-uzamsal bellek hem de sözel bellek boyutları programlama performansını anlamlı düzeyde açıklamıştır. Çalışma, programlama dilinin yazım kurallarını bilme, kod okuma, hata ayıklama gibi birçok sözel becerinin ve kod parçacıkları arasında gezinim, döngüler arası geçişler ve ilişkileri kavrama, algoritmanın akışını takip edebilme gibi görsel-uzamsal becerilerin programlamadaki önemli rolünü ortaya koymuştur. Ayrıca üniversite faktörü açısından; üniversitelerde verilen programlamaya giriş dersinin içeriği her ne kadar benzer olsa da derse giren öğretim üyesi, dersin değerlendirilmesi, sınavların puanlanması gibi çeşitli faktörlerin farklılık gösterdiği ifade edilmiştir.

Yılmaz (2013), üç farklı üniversitedeki meslek yüksekokullarında Bilgisayar Programcılığı programı 2. sınıf öğrencileri üzerinde programlama başarılarını etkileyen faktörleri ortaya çıkarmış ve öğrencilerin programlama başarıları ile bu faktörler arasında anlamlı bir ilişki bulunup bulunmadığını incelemiştir. İnceleme sonucunda programlama başarıları ve bu faktörler arasında anlamlı ilişki olduğunu bulmuştur. Araştırmasında ilişkisel tarama yöntemini kullanmıştır. Kız öğrencilerin erkek öğrencilere göre çok yüksek bir fark olmasa da daha başarılı olduğunu, öğrencilerin daha önce programlamayla ilgili deneyimlerinin olması ile olmaması arasında her hangi bir fark bulamadığını ve teknik lise mezunlarının daha başarılı olduğunu belirtmiştir. Ayrıca öğrencilerin genel akademik not ortalamaları ile sınav başarısı arasında pozitif bir ilişki görülmüştür.

Çalışmada problem çözme ve analitik düşünme becerilerinin programlama başarısı üzerinde herhangi bir etkisinin olmadığı sonucuna ulaşılmıştır. Ayrıca benzer çalışmaların farklı örneklem gruplarında tekrarlanması önerilmiştir.

Özdiñ ve Altun (2014) bilişim teknolojileri öğretmen adaylarının programlama sürecini izleyerek, bu süreçteki sorunları ve bu sorunların nedenlerini araştırmıştır. Çalışmaya, BÖTE Bölümü'ndeki 2011-2012 yılında İnternet Tabanlı Programlama dersi alan 6 öğrenci katılmıştır. Öğrencilere program yazma ve okuma ile ilgili etkinlikler verilmiştir. Program okuma sayesinde eş-zamanlı düşünme ve program yazma etkinliğinden sonra görüşme yapılmıştır. Görüşme geriye dönük sesli düşünme ile yapılmıştır. Elde edilen bulgulara göre, program yazma ve okuma görevlerinde ayrıntılı düşünme ve dikkatsizliğin hatalara neden olduğu, bireyin bilgi düzeyinin yükselmesinin kararsızlığı azalttığı ve bilgi düzeyi program yazma görevinde doğrudan bir ilişki görülmüştür. Program okuma görevinde ise üst düzey öğrencilerin okurken ayrıntılı düşündüğü için daha karmaşık yapılara odaklandığı ve bu yüzden yüzeysel kodlara programda dikkat etmediklerini belirtmiştir. Sonuç olarak program yazma ve okuma performanslarını etkileyen faktörlerin değişik beceriler ve bilişsel süreçler gerektirdiğini belirtmiştir.

Demir (2015) doktora çalışmasında, bir devlet üniversitesinin Bilgisayar Programcılığı bölümü 1. sınıf öğrencileri üzerinde, Programlama Temelleri dersi kapsamında, programlama öğretiminde eğitsel programlama dilinin derse entegrasyonunun, öğrencilerin akademik başarısına, performansına ve bilgisayar programlamaya yönelik kaygılarına olan etkisini araştırmıştır. Rasgele örneklem yöntemiyle 61 erkek ve 26 kız olmak üzere 87 öğrenciden oluşan üç deney grubu ile çalışmasını yürütmüştür. Birinci grupta eğitsel programlama dili, dersin uygulama kısmına, ikinci grupta teorik kısmına, üçüncü grupta ise hem teori hem de uygulama kısmına entegre edilerek gruplar oluşturmuştur. Eğitsel programlama dilinin derse entegrasyon şeklinin, bilişsel ve üst düzey bilişsel beceri puanları açısından, teorik entegre ve uygulama entegre grupları ile hem teorik hem de uygulama entegre grubu arasında anlamlı bir fark olduğunu belirtmiştir. Bu anlamlı farklılığın hem teorik hem de uygulama entegre grup lehine olduğunu tespit etmiştir. Öneri olarak; akademik başarı ve ders içi performansı artırmak ve bilgisayar programlamaya yönelik kaygıyı azaltmak için, eğitsel programlama dilleri, dersin hem teori hem de uygulamasına entegre edilerek kullanılabileceğini belirtmiştir.

Ayrıca programlama açısından acemi öğrenciler üzerinde yapılan bu çalışmanın programlama ön bilgisi olan öğrenciler üzerinde de gerçekleştirilebileceğini belirtmiştir.

Köse ve Tüfekçi (2015) yapay zeka kullanarak bir eğitim sistemi geliştirerek akış şeması ve algoritma konusunun öğretilmesini amaçlayan bir çalışma yapmıştır. Çalışmada, programlama derslerinin başlangıcında yer alan akış şeması ve algoritma kavramlarını öğreten bir eğitsel yazılımı incelenmiş ve hem yazılımı hem de yazılımla işlenen öğretim sürecini değerlendirerek bu alanda gerçekleştirilen çalışmalara yönelmiştir. Geliştirdikleri yazılım sisteminin akış şeması ve algoritma konularının öğretilmesinde etkin olabileceği sonucuna varılmıştır.

Demirer ve Sak (2016) programlama eğitiminin dünyada ve ülkemizdeki mevcut durumuna bir bakış açısı getirerek bu alanda ortaya çıkan yenilikçi yaklaşım ve uygulamaları incelemiştir. Çalışmada dünyada ve Türkiye’de düzenlenen bilimsel proje, etkinlik ve programlama eğitimleri irdelenmiştir. Ayrıca programlamayı öğrenmek isteyen bireylere ve çocuklara yönelik tasarlanan programlama araçlarına da yer verilmiştir. Yazılım konusunda uzmanlaşmış bireylere ülkemizin geleceği için ihtiyaç duyulduğundan programlama öğretiminin erken yaşlarda verilmesinin bir zorunluluk olduğu sonucu ortaya çıkmıştır.

Ülkemizin geleceği için üretken ve yaratıcı bireylerin yetiştirilmesi gerektiği, yazılım alanında yetişmiş insanlara ihtiyacın arttığı, bunun için de erken yaşta programlama eğitiminin elzem olduğu sonucuna ulaşılmıştır. Bundan dolayı programlama eğitiminin ilköğretim düzeyinde verilmesi için eğitim müfredatlarında gerekli düzenlemeler yapıldığını belirtmiştir. Çocuklara kod yazmanın kolay ve eğlenceli olduğunu ve programlama mantığını öğretebilmek için çeşitli proje ve etkinlikler düzenlendiği (Bilgisayar Programlama Çocuk Oyuncağı gibi) ve programlama araçları geliştirildiği (MIT App Inventor, Alice, Scratch gibi) vurgulanmıştır.

Benzer ve Erümit (2017) deneysel çalışma içeren programlama öğretimine yönelik lisansüstü tezleri analiz yöntemi kullanarak inceledikleri çalışmada; 29 adet tez üzerinde çalışmıştır. Çalışma sonucunda programlama öğretimi üzerine yapılan tezlerin sayısının her geçen yıl arttığı, tezlerin çoğunluğunda karma yöntem kullanıldığı, araştırmaların genellikle lisans düzeyinde ve özellikle bilgisayar ve öğretim teknolojileri öğretmenliği

bölümünde yoğunlaştığı sonucuna ulaşmıştır. Ayrıca araştırmalarda ağırlıklı olarak 31-60 katılımcılı olduğu, veri toplamada en çok ölçek ve başarı testleri kullanıldığı, tezlerin deneysel süresinin 4-6 hafta arasında süren çalışmaların çok olduğu ve nesnel programlama olan çalışmalarda C#, blok temelli çalışmalarda ise Scratch yazılımının tercih edildiği görülmektedir.

Cevahir ve Özdemir (2017) programlama öğretiminde karşılaşılan zorlukları, öğrencilerin hangi konuları öğrenirken zorluklar yaşadığını ve öğretmenlerin önerdikleri öğretim yöntemlerine ilişkin öğretmen görüşlerini incelemiştir. Çalışmanın örneklemini Çanakkale il merkezinde yer alan dört mesleki ve teknik anadolu lisesinde görevli 17 bilişim teknolojileri öğretmeni oluşturmaktadır. Araştırma olgubilim yöntemi ile desenlenmiştir. Öğretmenler ile dört temel sorudan oluşan yarı yapılandırılmış görüşmeler yapılarak veriler toplanmıştır. Bu sorular; programlama öğretiminde karşılaşılan genel zorluklar, öğrencilerin öğrenirken karşılaştıkları güçlükler, hangi konuların öğretiminde zorluklar yaşandığı ve programlama öğretiminde nasıl bir yöntem uygulanması gerektiği konularına odaklanmıştır. Alınan görüşlerin tematik analizi sonrasında beş farklı temaya ulaşılmıştır. Bu temalar, “ezbere dayalı eğitim”, “dizi ve döngü konularını öğrenmede yaşanan zorluk”, “tekrar ve uygulama eksikliği”, “soyut düşünme yeteneğindeki eksiklik” ve “aritmetiksel, matematiksel ve analitik düşünme yetersizliği” olarak isimlendirmiştir. Bilgisayar programlamada her zaman farklı problemlerle karşılaşıldığından ezberci yaklaşım sergileyen öğrencilerin programlama öğreniminde daha fazla zorlanabileceği ifade edilmiştir. Öğrencilerde aritmetiksel, matematiksel ve analitik düşünme yetersizliğinin programlama öğretimini zorlaştırdığına yönelik genel bir görüşün olduğu tespit edilmiştir. Öğretmenlerin programlama derslerindeki tekrar ve uygulama eksikliğinin öğrenmeyi zorlaştırdığı ve öğrencilerin en çok dizi ve döngü komutlarını öğrenmede zorluklar yaşadığı bulunmuştur. Ayrıca öğrencilerin soyut düşünme yeteneğindeki eksikliğin programlama öğretiminde karşılaşılan zorluklardan biri olduğu görüşü üzerinde durulmuştur.

Erol ve Kurt (2017) BÖTE bölümünde öğrenim gören öğrencilerin programlamaya karşı tutumlarını incelemiştir. 2013-2014 bahar yarıyılında Burdur Mehmet Akif Ersoy Üniversitesi BÖTE bölümünde öğrenim gören 128 öğrenciye Bilgisayar Programlamaya Karşı Tutum ölçeği uygulanmıştır. Katılımcıların programlamaya karşı tutumlarının olumlu olduğu, tutumlarının cinsiyet ve lise türüne göre anlamlı farklılık göstermediği

ancak sınıf düzeyine göre anlamlı bir farklılık olduğu bulgularına ulaşmıştır. Birinci sınıfta öğrenim gören öğrencilerin programlamaya karşı tutumları ikinci ve üçüncü sınıfta öğrenim gören öğrencilere göre daha olumlu bulunmuştur. Bunun nedeni olarak da, programlama dersi aldıkça öğrencilerin programlamaya karşı tutumlarının olumsuz hale geldiği ileri sürülmüştür. Ayrıca tutum puanı ortalaması ile oluşturulmuş gruplar ile odak grup görüşmesi yapılarak öğrencilerin programlamaya karşı olan tutumlarına ilişkin görüşleri incelenmiştir. Odak grup görüşmesi bulguları programlamaya karşı olumsuz tutumun sınav kaygısından, başarısız olmaktan ve programlama temel mantığını kavrayamamaktan kaynaklandığını göstermiştir. Programlamaya karşı olumlu tutumun ise programlamayı eğlenceli bir aktivite olarak görme, bilgisayara hükmetme hissi ve bir şeyler üretme duygusundan kaynaklandığı sonucuna ulaşılmıştır. Öğrencileri programlamada zorlayan etmenler ise problem analizinde yaşanan zorluklar, kod yazma zorluğu ve yapıları nerede kullanacağını karıştırma şeklinde sıralanmıştır. Programlama mantığını kavramakla ilgili problemin ise algoritma öğretimi sürecinde uygulama olmaması, bu işe kısa süre ayrılması ve hazırbulunuşluk düzeyine dikkat edilmemesi kaynaklı olduğu belirtilmiştir.

Özyurt ve Özyurt (2017) yaptıkları çalışmada; programlama dersine ilişkin öğrencilerin görüşlerini incelemiştir. Proje tabanlı öğrenme üzerine öğrencilerden aldıkları görüşler çerçevesinde çalışmalarını yapmıştır. Çalışma grubu olarak 2015-2016 yılında Yazılım Mühendisliği Bölümünde okuyan öğrencilerden Görsel Programlama dersi almakta olan 11 öğrenci seçilmiştir. Seçilen öğrencilerden 5 tanesi bağımsız olarak, kalan 6 öğrencide ikişerli grup olarak çalışmıştır. Öğrenciler programlama alanında istedikleri konuya yönelik projeler üzerinde çalışmıştır. Çalışmalar esnasında iki grup ve bağımsız çalışan bir öğrenci çalışmadan çıkmıştır. Nitel araştırma yöntemi kullanılan çalışmada tek grup ve bağımsız çalışan 4 öğrenci ile görüşmeler yapılmıştır. İçerik analizi verilere uygulanmıştır. Çalışmadan çıkarılan sonuçlara göre; öğrenciler yazılım mühendisliğini iş hayatının bir provası olarak değerlendirmişlerdir. Öğrenciler yazılım mühendisliği bölümünde okumanın problem çözme becerisi ve kişinin kendini tanımasına katkıda bulunduğunu belirtmişlerdir. Öğrencilerin bir kısmı proje tabanlı öğretimde araştırmaya ayrılan zamanı dezavantaj olarak görmüştür.

Saygıner ve Tüzün (2017) araştırmalarında “Programlama öğretiminde karşılaşılan zorluklar ve bunlara yönelik muhtemel çözüm önerileri nelerdir” sorularına cevap

vermeyi amaçlamıştır. Bu araştırma, alanyazını taramaya odaklanarak betimsel bir derleme ortaya çıkarma çalışmasıdır. Bu kapsamda çeşitli akademik veri tabanlarında “bilgisayar programlama, öğrenimi, zorluklar” gibi kelimeler aranarak tarama yapılmıştır. Aramalar sonucunda ulaşılan çalışmalardan 37 tanesi incelenmiştir. Araştırmaya göre programlama eğitiminde yaşanan zorlukların nedenleri olarak programlama mantığı ve algoritma konusunun öğretilmesinde yaşanan sıkıntılar olarak belirtmiştir. Programlamanın somut olmayan yapısından ve üst düzey düşünme becerileri gerektirmesinden dolayı başlangıç aşamasındaki bireylerin zorluk yaşamasına sebep olduğu sonucuna ulaşılmıştır.

Yecan, Özçınar ve Tanyeri (2017) bilişim teknolojileri dersinde görsel programlama öğretimi yapan öğretmenlerin gözünden ilköğretimde görsel programlama öğretimini incelemişlerdir. Araştırma karma yöntem yaklaşımıyla yürütülmüş; öğretmenlerle yüz yüze görüşmeler sonrasında bir anket geliştirilerek uygulanmıştır. Öğretmenlerin, programlamaya giriş amacıyla birbirlerinden farklı yöntemler kullandıklarını, gerçek hayattan örnekler sunduklarını, oyun tasarımı ve grup çalışmalarıyla ders işlediklerini, öğrencilerin genel akademik başarı durumları ve programlamaya ilgi düzeyleri arasındaki farklılıkların önemli olduğunu ortaya çıkarmıştır. Programlamanın, öğrencilerin problem çözme, mantık yürütme ve algoritma oluşturma gibi farklı alanlardaki becerilerinin, yaratıcılık ve üreticiliklerinin gelişmesine katkı sağlayabileceği sonucuna varılmıştır. Ayrıca programlama öğretimine küçük yaşlarda başlamanın öğrencilerin gelişimi açısından olumlu katkılar sağlama potansiyeli olduğu belirtilmiştir.

2.1.5.3. Dünyada programlama öğretimi

Codewitz (www.codewitz.net) projesi, öğrencilerin ve öğretmenlerin giriş seviyesi programlama derslerinde yardımcı olmak için etkileşimli web tabanlı öğrenme nesneleri geliştirmekte ve üretmektedir. Projenin temel amacı, sınıfta kullanmak için programlama kavramlarının web tabanlı görselleştirmelerini geliştirmek ve bağımsız öğrenmeyi desteklemektir. Algoritma animasyonuna odaklanan çeşitli yaklaşımlardan farklı olarak, bu proje öğrencilerin öğrenmeye yönelik programlama dillerini ve dil yapılarını, örneğin değişkenleri, döngüleri ve koşullu ifadeleri geliştirmeyi amaçlamaktadır (Lahtinen vd., 2005).

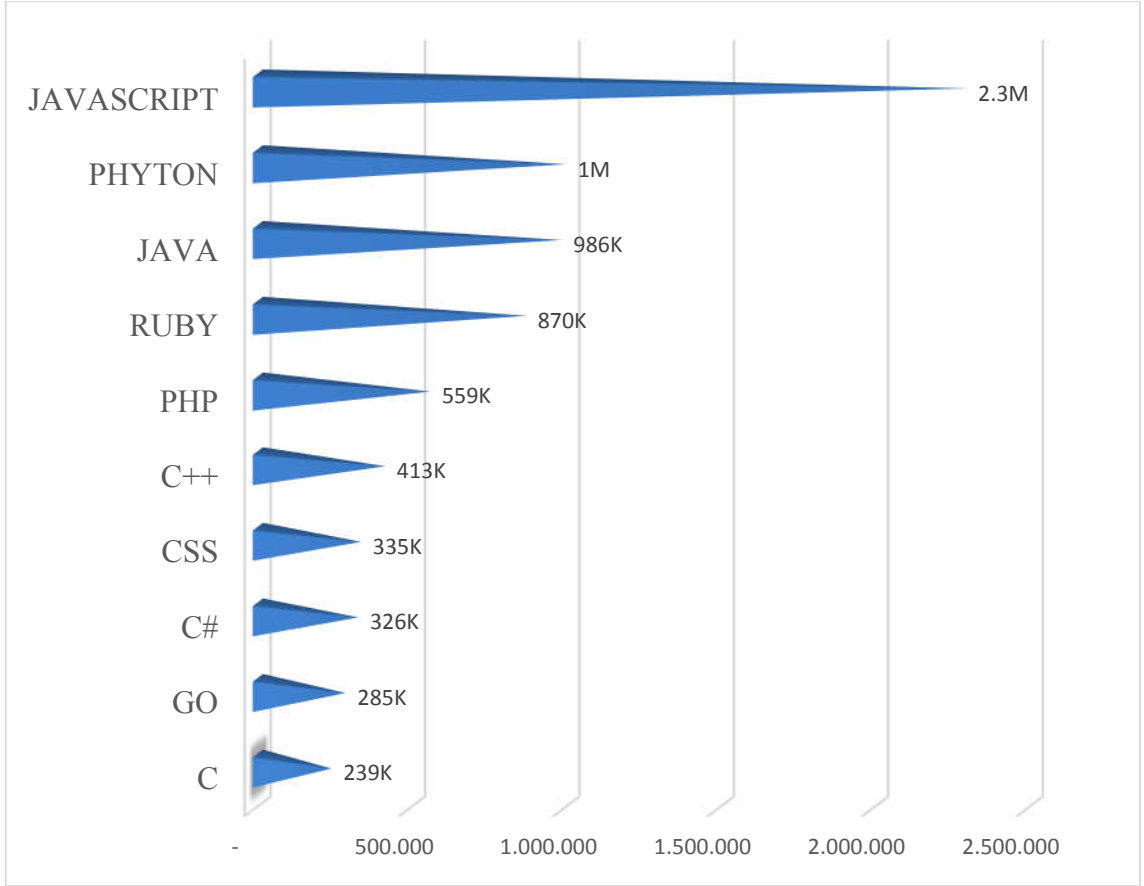
Code Club, 2012 yılında kurulmuş, gençlerin fikirlerini kodla nasıl oluşturacağını öğrenmesine yardımcı olmak için çalışan önemli bir eğitim programıdır. Bu organizasyon 9-13 yaşları arasındaki gençler için ücretsiz kodlama kulüpleri işleten gönüllü ve eğitimcilerden oluşan bir ağıdır. Bu ağ üzerinde 7 bin civarı kod kulübü, 75 proje ve 97 bin öğrenci bulunmaktadır. Gençlerin oyun, animasyon ve web siteleri oluşturarak Scratch, HTML & CSS ve Python öğrenmelerine yardımcı olmaktadır. Şu anda 100'den fazla ülkede 10 binden fazla kulüp var ve bu kulüplerin projeleri 28 dile çevrilmiştir (Code Club, 2014).

Microsoft tarafından 2014 yılında satın alınan Minecraft oyunu, eğitim alanında kullanılmak üzere revize edilerek özellikle kodlama alanında kullanılmaya başlanmıştır. Code.org ve Microsoft arasındaki anlaşma sonrasında 85 milyon öğrencinin Minecraft yardımıyla kodlama eğitimine başladıkları duyurulmuştur. Code.org, okullarda bilgisayar bilimlerine erişimi genişletmeye ve kadınlarında katılımını arttırmaya yönelik bir sivil toplum kuruluşudur. Ayrıca kendi kurslarını tasarlamak, öğretmenleri eğitmek, büyük okul bölgeleriyle işbirliği yapmak, hükümet politikalarını değiştirmek, uluslararası işbirlikleri yoluyla ortaklaşa çalışmak ve kalıplaşmışın dışına çıkmak için çalışan bir kuruluştur. Kuruluşun vizyonu, her okuldaki her öğrencinin bilgisayar bilimlerini öğrenme fırsatına sahip olmasıdır. Code.org, Amerika Birleşik Devletleri'nin bilgisayar bilimi alanında müfredatı sağlamakta ve dünyadaki tüm öğrencilerin %10'unu kapsayan yıllık Code of Hour kampanyaları düzenlemektedir. Code.org; Amazon, Facebook, Google, Infosys Foundation, Microsoft ve daha birçok bağışçı tarafından desteklenmektedir (Code.org, 2018).

Robincode.org (2014), programlamanın temellerini, siber güvenlik ve tekno girişimciliği; kendi geliştirdiği eğitim sistemini kullanarak 5-16 yaş arası çocuklara öğreten bir eğitim hareketidir. Amaçları teknolojiyi sadece kullanan değil üreten bir nesil yaratmak ve algoritmik düşünmeyi çocuklara nasıl geliştireceklerini öğretmektir. Oracle Akademi, Google ForNonProfits, Cisco Networking Akademi ve Paypal bu hareketin ortaklarıdır.

Arsenal kulübü stadyumda her yıl öğrenciler için teknoloji festivalleri düzenlemektedir. Türkiye'deki kulüpler de benzer etkinlikler yapabilir. BBC kanalı kodlama üzerine dersler ve etkinlikler üretmektedir. Allsop (2016) ülkemizde de TRT'nin bu tarz ders ve etkinlik üretip bütün ülkeye ulaşabileceğini belirtmiştir.

Şekil 7’de proje geliştirme alanında faaliyet gösteren dünyaca ünlü Github sitesinin 24 milyon kullanıcılarından elde ettiği 2017 yılı verilerine göre yayınladığı programlama dilleri sıralaması görülmektedir. Github sitesindeki kullanıcıların en çok tercih ettiği dil Javascript olmuştur. Onu sırasıyla Python, Java, Ruby takip etmektedir.



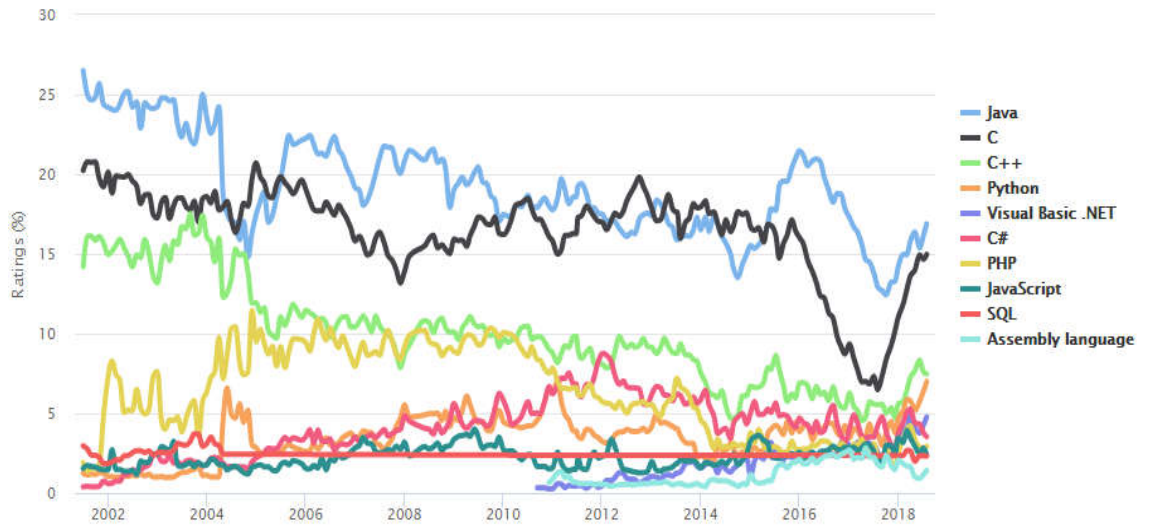
Şekil 7. Programlama dilleri sıralaması (Github, 2017)

160 ülkede 420 bin üyesi olan Elektrik ve Elektronik Mühendisleri Enstitüsü (IEEE), dünyanın en popüler programlama dillerini sıraladığı IEEE Spectrum listesinin 2018 sonuçlarını yayınlamıştır (Şekil 8). Programlama dilleri konusunda önemli bir çalışma hazırlayan IEEE, GitHub sitesi üzerindeki 300 programlama dilini esas aldığı liste için Google Trendler’ den Twitter’ a, Hacker News’ tan Stack Overflow, Dice ve Reddit’ e çok sayıda açık ve kapalı kaynağı tarayarak uzun bir araştırma sürecinin sonucunda listeyi açıklamıştır (Gowrishankar ve Veena, 2018).

Dil Sıralaması	Tipi	Spektrum (Yelpaze) Sıralaması
1. Python	🌐 🖥️ 📱	100.0
2. C++	📱 🖥️ 📱	98.3
3. C	📱 🖥️ 📱	98.3
4. Java	🌐 📱 🖥️	97.2
5. C#	🌐 📱 🖥️	92.7
6. R	🖥️	82.8
7. PHP	🌐	82.7
8. JavaScript	🌐 📱	82.6
9. Go	🌐 🖥️	76.4
10. Assembly	📱	74.2
11. Matlab	🖥️	72.8
12. Scala	🌐 📱	72.1
13. Ruby	🌐 🖥️	71.3
14. HTML	🌐	70.5
15. Arduino	📱	69.0

Şekil 8. En iyi programlama dilleri (Gowrishankar ve Veena, 2018)

TIOBE tarafından 2002-2018 yılları arasındaki programlama dilleri kullanım istatistikleri içeren grafik yayınlanmıştır (Şekil 9). Sıralamadan görüldüğü üzere Java bu yıllar arasında kısa bir dönem hariç hep en çok kullanılan dil olarak yerini almıştır. C ise diğer en çok kullanılan dillerden birisidir. Bazı yıllarda C dilinin en çok kullanılan dil olduğu görülse de 2018 yılında ikinci sırada yer almaktadır. Diğer diller ise C++, Python, Visual Basic .NET, C#, PHP, JavaScript, SQL, Assembly Language olarak sıralanmaktadır.



Şekil 9. 2002-2018 programlama dilleri kullanım istatistikleri (TIOBE, 2018)

Programlama dilleri üzerine sürekli ve güncel raporlar hazırlayan The Importance of Being Earnest (TIOBE) en popüler programlama dilleri için Ağustos 2018 ayı için yayınladığı istatistik raporunda (Şekil 10) Java zirvedeki kullanma oranını artırarak yola devam ettiği görülmektedir. Java’ nın ilk sırada yer almasında Android’ in payı tartışılmaz bir gerçek olarak görülmektedir. Java’nın arkasından C ve C++ dilleri gelmektedir. İlk üç sıranın hemen ardında Python, Visual Basic ve C# yer almaktadır.

2018	2017	Değişim	Programlama Dili	Derecelendirme	Değişim
1	1		Java	16.881%	+3.92%
2	2		C	14.966%	+8.49%
3	3		C++	7.471%	+1.92%
4	5	▲	Python	6.992%	+3.30%
5	6	▲	Visual Basic .NET	4.762%	+2.19%
6	4	▼	C#	3.541%	-0.65%
7	7		PHP	2.925%	+0.63%
8	8		JavaScript	2.411%	+0.31%
9	-	▲	SQL	2.316%	+2.32%
10	14	▲	Assembly language	1.409%	-0.40%
11	11		Swift	1.384%	-0.44%
12	12		Delphi/Object Pascal	1.372%	-0.45%
13	17	▲	MATLAB	1.366%	-0.25%
14	18	▲	Objective-C	1.358%	-0.15%
15	10	▼	Ruby	1.182%	-0.78%

Şekil 10. Ağustos 2017-2018 programlama dilleri sıralaması (TIOBE, 2018)

2.1.5.4. Dünyada programlama öğretimi üzerine yapılan çalışmalar

Lahtinen vd. (2005) yaptıkları araştırmada temel programlama dersleri için öğrenme materyalleri geliştirmeye yardımcı olmak amacıyla programlama öğrenmedeki zorlukları incelemişlerdir. Araştırmaya 559 öğrenci ve 34 öğretmen katılmış ve katılımcılardan, programlama öğrenirken ve öğretirken yaşanan zorluklarla ilgili bilgi sağlayan bir anket kullanılmıştır. Sonuç olarak, öğrencilerin temel kavramları anlamada ve bu bilgileri kullanmada güçlük çektikleri, programlama derslerindeki kavramların öğrenciler için soyut geldiği ve bu problem dikkate alınarak öğrenme materyalleri yapılması gerektiği vurgulanmıştır.

Gomes ve Mendes (2007), “Programlama öğrenme zorlukları ve çözümleri” adlı derleme çalışmalarında; programlama öğrenmede geleneksel öğretim yöntemlerinin yetersizliği, öğrencinin izlediği çalışma yöntemlerinin öğrenmeyi programlamak için uygun olmaması, öğrencilerin yeterli matematiksel ve mantıksal bilgiye sahip olmamaları ve problem çözme becerilerinin yetersiz olması, programlamanın yüksek derecede soyutlamadan oluşması gibi nedenlerden dolayı öğrencilerin öğrenme zorlukları yaşadıklarını belirtmiştir.

Renumol vd. (2009) programlama eğitimi ile ilgili problemleri araştırmak için 137 mühendislik öğrencisinden programlama eğitimi üzerine toplanan çeşitli verileri analiz etmişlerdir. Öğrencilerin programlama dilinden bağımsız olarak programlama ve program anlama sırasında çeşitli seviyelerde bilişsel zorluklara sahip olduklarını ve etkili ve etkisiz öğrencilerin öğrenme stillerinde belirgin bir farklılık olduğunu belirtmişlerdir. Öğrencilerin yaşadığı zorlukların çoğunun bilişsel alanda olduğu ve sadece programlama öğreniminde yaşamadıklarını belirtmişlerdir. Öğrenciler de bilişsel becerilerin güçlendirilmesinin programlama öğreniminde kolaylık sağlayacağı ve programlama derslerini daha etkili, keyifli ve çekici hale getirmek için çığır açan bir araştırmaya ihtiyaç olduğu sonucuna varmışlardır.

Derus ve Ali (2012) programlama dersi alan 105 öğrenci üzerinde yapmış oldukları çalışmada öğrencilerin konuya ilişkin algı düzeyinin orta düzeyde olduğunu ve en zor konunun soyut kavramları (çok boyutlu dizi, döngü ifadesi ve işlev) anlamak olduğunu belirtmişlerdir. Ayrıca öğrencilerin programlama yapısının temel kavramını anlamada ve belirli görevleri çözmek için bir program tasarlamada zorluklarla karşılaştıklarını söylemişlerdir. Öğrencilerin çoğunluğu laboratuvarında uygulama yapmanın öğrenmeyi kolaylaştırdığını ancak öğretim elemanlarının yeterince örnek vermediğini belirtmişlerdir.

Blikstein vd. (2014) bir lisans programına kayıtlı yaklaşık 370 öğrenci tarafından geliştirilmekte olan bilgisayar programlarının 154 bin ekran görüntüsü kaydını incelemişlerdir. Elde edilen verilere dayanarak, öğrencilerin bilgisayar programlamayı nasıl öğrendikleri üzerine odaklanmış çalışmalar sunmuşlardır. Verilerdeki kalıpları bulmak için makine öğrenimi yöntemlerini kullanmışlar ve final sınav notlarını tahmin etmeye çalışmışlardır. Öğrencilerin programlama modellerindeki değişimin sadece ders

performansını zayıf bir şekilde öngördüğü sonucuna ulaşmışlardır. Final sınavları, ara sınav notlarına göre daha iyi tahmin gücüne sahip olduğunu belirtmişlerdir.

Xinogalos (2016) programlama derslerini tasarlama ve sabitlemeyi; stratejiler, araçlar, zorluklar ve pedagoji olarak incelemiştir. Bu durumun şüphesiz zor bir iş olduğunu belirtmiştir. 50 gönüllü öğrenciye anket uygulayarak, öğrencilerin anket sorularına, ilgili çalışmalarına ve öğretmenlerin dersleri öğretme konusundaki deneyimlerine dayanarak analizlerini yapmıştır. Öğretmenlerin programlama öğrenmeyi desteklemek için yüksek kalitede sunumlar, laboratuvar çalışmaları, ödevler ve örnek çözümler hazırlamasının çok önemli olduğu sonucuna varmıştır.

Baist ve Pamungkas (2017) bilgisayar programlamada öğrenci zorluklarının analizini yaptıkları çalışmada programlamada karşılaşılan zorlukları belirlemeyi amaçlamıştır. Tangerang Muhammadiyah Üniversitesi Matematik Eğitimi bölümünden 20 erkek ve 112 bayan toplam 132 öğrenci üzerinde anket uygulamıştır. Matematik sınav puanına göre yüksek, orta ve düşük olarak üç gruba ayırmıştır. Bu gruplarda çok boyutlu dizi materyallerinde, giriş çıkış komutlarında, temel programlama kavramını anlamada, programın hatalarını bulmada öğrencilerin yaşadığı zorluklarda önemli farklılıklar olduğunu belirtmiştir. Laboratuvar çalışmaları, tek başına görev yapma ve gösterilen örnekleri yapmada ise fark olmadığını belirtmiştir.

Zhong, Wang, Chen ve Li (2017) ilkokul öğrencileri üzerinde Çift Programlama (Pair Programming) üzerine bir çalışma yapmışlardır. Bu çalışma da 8 adet ilkokul 6. sınıftan 4 tanesi rastgele seçilmiştir. Bu 4 sınıfta toplam 150 öğrenci bulunmaktadır. Bu öğrenciler 2014 sonbahar ve 2015 güz dönemlerinde Scratch kursu almışlar ve program hakkında temel bilgi ve beceriye sahip olmuşlardır. Uygulamadan önce bir sınav ve bir ön test yapılmış ve tek yönlü ANOVA sonucunda dört sınıf arasında programlama bilgisi olarak anlamlı bir fark bulunamamıştır. 13 hafta boyunca eğitim verilmiştir. Eğitim sonunda 3 başarı testi ve bir tutum araştırması yapılmıştır. Araştırma çift programlama çalışmalarında öğrencilerin hangi sıklıklarda rol değiştiklerinde daha etkili bir öğrenme gerçekleşeceğini ortaya çıkarmak için yapılmıştır. 5 dakika da bir ve yarı özgürce rol değiştiren öğrenciler, her görevde ve her oturumda rol değiştirenlerden daha fazla keyif aldıkları sonucuna ulaşmışlardır.

Yang, Wong ve Dawes (2018), yirmi birinci yüzyıl bilgisayar programcılığına ilişkin öğrenme tutumuna yönelik keşifsel bir çalışma yapmıştır. Çalışmalarında farklı ülkelerden iki yüz otuz 7. ve 8. sınıf öğrencisi için 12 tane programlama atölyesi yapmıştır. Öğrencilerin atölye çalışmalarına katılmadan önce ve sonra programlama öğrenmeye ilişkin tutumları ve görüşlerini incelemiştir. Öğrencilerin büyük bir kısmının okul dışı programlama tecrübesine sahip olmadığı, zengin çevrimiçi öğrenme kaynakları olmasına rağmen, öğrenciler bunlardan yararlanmadığı, kendilerini yetersiz hissettikleri için olumsuz görüşlere sahip oldukları ve kişisel olarak öğrenme programlamasının, öğrencilerin öğrenmelerinde olumlu etki yapacağı sonucuna ulaşmıştır.



3. YÖNTEM

3.1. Araştırma Modeli

Bu tez çalışması nitel araştırma paradigmasına dayalı durum çalışması yöntemiyle desenlenmiştir. Nitel araştırmalarda sıklıkla durum çalışması yöntemi kullanılmaktadır (Exworthy, Peckham ve Powell, 2012; Yıldırım ve Şimşek, 2013). Literatürde durum çalışması, vaka çalışması ya da örnek olay incelemesi gibi farklı isimlerde de kullanılmaktadır (Subaşı ve Okumuş, 2017). Durum çalışması, güncel bir olguyu derinlemesine ve gerçek yaşam bağlamında araştıran deneysel bir sorgulamadır (Exworthy vd., 2012; Thomas, 2015; Yin, 2017). Merriam (2013) ise sınırlı bir durumun ayrıntılı bir şekilde incelenmesi ve betimlenmesi olarak açıklamıştır. Durum çalışmasında amaç belirli bir duruma ilişkin sonuçlar ortaya koymaktır. Bir ya da birkaç durumun derinlemesine araştırılması, nitel durum çalışmasının en temel özelliğidir. Ayrıca durum çalışması “nasıl” ve “niçin” sorularını temel alarak, kontrol edilemeyen bir olgu ya da olayın ayrıntılarıyla incelenmesine imkân veren bir araştırma yöntemidir (Ozan Leylum, Odabaşı ve Kabakçı Yurdakul, 2017). Exworthy vd. (2012) durum çalışmasının araştırma tasarlama ve yürütmede var olan sorunları içerdiğini belirtmiştir. Durum çalışması; programların, olayların, kişilerin, süreçlerin, kurumların, sosyal grupların ve diğer güncel olayların araştırmasında kullanılabilir (Hancock ve Algozzine, 2016). Durum çalışmaları genellikle uygulama süreçlerini belgelemek ve analiz etmek için kullanıldığından dolayı süreç değerlendirmeleri ile ilişkilendirilmiştir (Yin, 2017). Durum çalışması süreci; araştırma sorularının geliştirilmesi, alt problemlerin oluşturulması, analiz biriminin saptanması, çalışılacak durumun belirlenmesi, bireylerin seçimi, verilerin toplanması ve alt verilerle ilişkilendirilmesi, analiz, yorumlama ve raporlaştırma aşamalarından oluşmaktadır (Yıldırım ve Şimşek, 2013). Araştırmanın konusuna bağlı olarak, durum çalışmaları güçlü ve zayıf yönleri dikkate alınarak tek başına ya da diğer yöntemlerle birlikte yapılabilir (Yin, 2017). Bu tez çalışmasında da programlama öğretiminde yaşanan sorunlar bir durum olarak ele alınarak öğrencilerle görüşmeler yapılarak derinlemesine araştırılması amaçlandığından araştırmada kullanılacak en uygun yöntemin durum çalışması olduğu düşünülmüştür.

3.2. Araştırmanın Çalışma Grubu

Çalışmanın katılımcıları MAKÜ HTMYO ve EGTBMYO’nda öğrenim görmekte olan ve en az bir kere bilgisayar programlama ile ilgili bir dersi almış öğrencilerden oluşmaktadır. HTMYO’da 2017-2018 eğitim-öğretim yılı itibariyle toplam 2655 öğrenci, 7 farklı bölüm bünyesinde yürütülen 11 farklı ön lisans programında öğrenim görmektedir. Bu programlar Pazarlama, Dış Ticaret, Halkla İlişkiler ve Tanıtım, Muhasebe ve Vergi Uygulamaları, İşletme Yönetimi, Lojistik, Havacılıkta Yer Hizmetleri Yönetimi, Çocuk Gelişimi, Bilgi Yönetimi, Turizm ve Otel İşletmeciliği programlarıdır. Bilgisayar programlama dersi ise yalnızca Bilgi Yönetimi (BY) programı müfredatında yer almaktadır. BY programında halen 24 birinci sınıf ve 95 ikinci sınıf öğrencisi olmak üzere toplam 119 öğrenci eğitime devam etmektedir. EGTBMYO’nda 2017-2018 eğitim-öğretim yılı itibariyle toplam 1999 öğrenci, 8 farklı bölüm bünyesinde yürütülen 10 farklı ön lisans programında öğrenim görmektedir. Bu programlar BY, Sağlık Bilgi Sistemleri Teknikerliği, Alternatif Enerji Kaynakları Teknolojisi, Elektrik, Hibrid ve Elektrikli Taşıtlar Teknolojisi, Biyomedikal Cihaz Teknolojisi, Elektronik Teknolojisi, Otomotiv Teknolojisi, Coğrafi Bilgi Sistemleri, Grafik Tasarım, Moda Tasarım programlarıdır. Bilgisayar programlama dersi ise Sağlık Bilgi Sistemleri Teknikerliği ve BY programı müfredatında yer almaktadır. Görüşmelere katılan öğrenciler BY programında eğitim gören öğrencilerden oluşmaktadır. BY programında Programlamaya Giriş, Programlama Dili I ve Programlama Dili II olmak üzere 3 dönem boyunca programlama dili dersi müfredatta yer almaktadır. Programlamaya Giriş dersinde temel algoritma ve programlama teknikleri C# programlama diliyle anlatılmaktadır. Programlama Dili I ve II derslerinde ise görsel programlamaya geçilmekte ve aynı dil kullanılarak form uygulamaları işlenmektedir.

Programlama dersi alan öğrencilerin araştırma kapsamında kullanılması düşünüldüğünden seçkisiz olmayan örnekleme yöntemlerinden amaçsal örnekleme yöntemi tercih edilmiştir. Amaçsal örnekleme yöntemi, seçkisiz ve olasılı olmayan bir çeşit örnekleme yöntemidir. Bu örnekleme yöntemi çok fazla bilgi içeren durumların seçilerek araştırmanın derinlemesine yapılmasını sağlar. Araştırmacı seçilen durumları anlamaya ve arasındaki ilişkileri keşfedip açıklamaya çalışır (Büyüköztürk, Çakmak, Akgün, Karadeniz ve Demirel, 2018). Bu yöntem ile programlama dersini en az bir kere almış olan öğrenciler tespit edilmiş ve bu öğrencilere bilgilendirmeler yapılarak

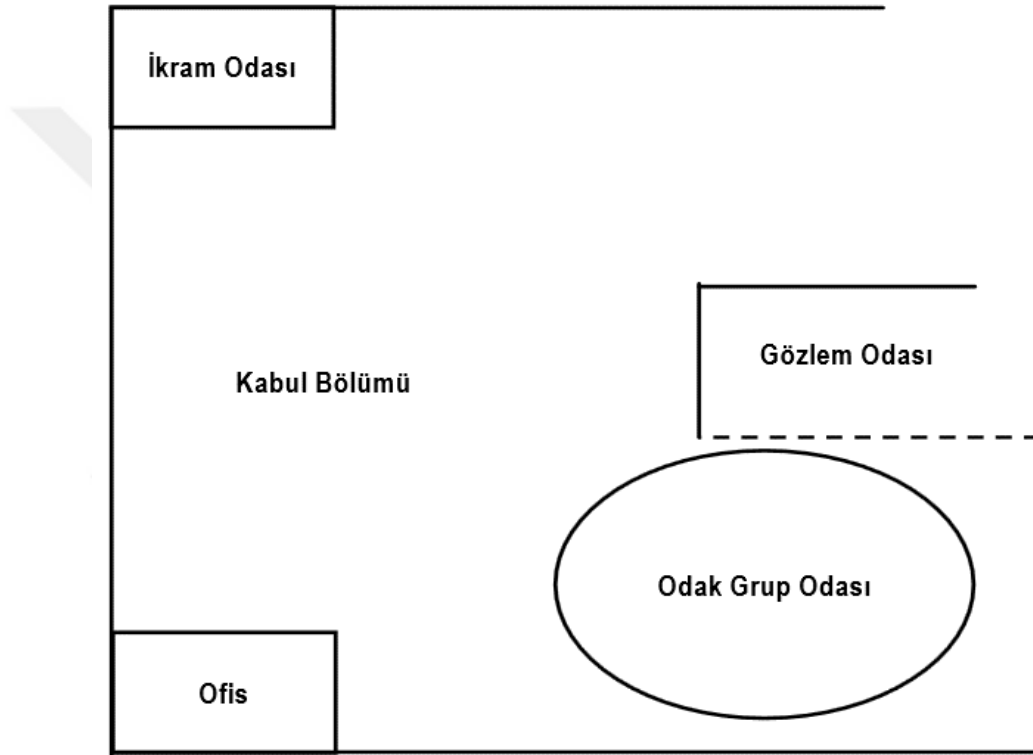
araştırmaya katılmak isteyenler belirlenmiştir. Araştırmaya katılmak isteyen öğrencilerin bilgileri daha sonra irtibata geçebilmek için kaydedilmiştir. Öğrenciler seçilirken lise mezuniyet alanları ve cinsiyetleri dikkate alınarak benzer yoğunlukta olması sağlanmıştır. 6 öğrenciden oluşan 3 farklı grup ile mülakatlar yapılması düşünülmüştür. Bir öğrenci çağrılmasına rağmen mülakata gelememiştir. Bundan dolayı ilk etapta 3 grupta yer alan 17 öğrenci ile görüşmeler yapılmış ancak yeni bilgilerin halen görüşmelerden elde edildiği görüldüğünden dolayı 1 grup ile daha görüşmelerin yapılması uygun görülmüştür.

Araştırmanın amacına ulaşmak için HTMYO Bilgi Yönetimi Programı'nda okuyan 16 öğrenci ve EGTBMYO Bilgi Yönetimi Programı'nda okuyan 6 öğrenci ile odak grup görüşmeleri yapılmıştır. Görüşmeler 5 ve 6 kişilik ikişer grup olmak üzere dört farklı grupta gerçekleştirilmiştir. Soru ve yanıt dayalı olarak yapılan çalışmada genelden özele beş adet soru yöneltilmiştir. Odak grup görüşmeleri önceden hazırlanmış yarı yapılandırılmış çerçeve sorular ile yürütülmüş ve görüşme kayıtları ses kayıt cihazı aracılığıyla kaydedilmiştir. Görüşme esnasında öğrencilerin söylediklerinden yola çıkarak farklı sorular da sorulmuştur. Tutulan ses kayıtları daha sonra transkript edilip betimsel analiz yapılarak belli temalar altında yorumlanmıştır.

3.3. Veri Toplama Süreci

Odak grup görüşmeleri yapabilmek için gereken izin sözlü olarak okul idarelerinden alınmıştır. Odak grup görüşmesi için programlama dili dersini daha önceden almış olan sınıflarda bilgilendirmeler yapılmış ve gönüllülük esasına göre öğrenciler belirlenmiştir. Belirlenen öğrencilerin listesi oluşturulmuş ve bilgileri alınmıştır. Belirlenen öğrencilerle ön görüşme yapılmış ve görüşme hakkında bilgiler verilmiştir. Ön görüşmede ayrıca öğrencilere yer ve zaman bilgisi de verilmiştir. Görüşmelerin video veya ses kaydı ile yapılacağı bildirilmiş ve öğrencilerin isteği üzerine sadece ses kaydı yapılması kararlaştırılmıştır. Video ile kayıt yapılması durumunda kendilerini rahat hissedemeyeceklerini belirtmişlerdir. Odak grup görüşme tarihinden bir gün önce katılımcılara telefon ile ulaşılarak görüşme saati ve yer bilgisi tekrar hatırlatılmıştır. Araştırma için gerçekleştirilen odak grup görüşmeleri 2017-2018 eğitim öğretim yılının bahar döneminde araştırmacının kendisi tarafından yapılmıştır.

Nakip ve Yarař (2017) odak grup grřmeleri yapılacak olan mekanın konforlu ve grřmeye uygun řekilde tasarlanmış olması gerektiğini belirtmiştir (řekil 11). Bu tasarıma uygun olarak grřmeler bilgisayar laboratuvarında gerekleřtirilmiştir. Katılımcıların kendilerini rahat hissetmelerini saėlayacak řekilde ve grřmelerin sesli olarak kaydedilmesi planlandığından dolayı ortam buna uygun olarak hazırlanmıştır. U dzeninde ėrenciler oturmuş ve her ėrenciye kaėıt kalem verilmiştir. Grřmelerin rahat gemesi iin ėrencilere yiyecek iecek ikramı yapılmıştır. Grřme bařlamadan nce katılımcılara grřme hakkında bilgilendirme yapılmıştır.



řekil 11. Odak grup grřme mekanı tasarımı (Nakip ve Yarař, 2017)

Tablo 6’da ėrenciler ile yapılan odak grup grřme takvimi yer almaktadır. ėrenciler ile yapılan odak grup grřmeleri yaklaşık 40 dakika ile 60 dakika arasında srmřtr. Grřmelerde ara verme ihtiyaı olmamıştır. Grřmeler esnasında moderatr ve raportr grevini arařtırmacı stlenmiştir. Odak grup grřmesi iin arařtırmacı ve katılımcılar, belirlenen tarih ve saatte belirlenen yerde toplanmışlardır.  farklı oturum gerekleřtirilmiştir. Farklı bilgiler gelmeye devam ettiėi iin drdnc bir grřme daha yapılması planlanmıştır. Elde edilen verileri netleřtirmek iin daha sonra bir oturum daha yapılması uygun grlmřtr.

Tablo 6.Odak grup görüşme takvimi

	Öğrenciler	Tarih	Süre
Birinci Odak Grup Görüşmesi	K1 - K2 - E1 E2 - E3	09.05.2018	50 dakika
İkinci Odak Grup Görüşmesi	E4 - E5- E6 K3 - K4 - E7	11.05.2018	60 dakika
Üçüncü Odak Grup Görüşmesi	E8 - K5 - K6 K7 - K8 - E9	15.05.2018	45 dakika
Dördüncü Odak Grup Görüşmesi	E10 - K9 - K10 E11 - E12	04.06.2018	40 dakika

3.4. Veri Toplama Aracı

Programlama dili öğretiminde yaşanan zorlukların öğrencilerin bakış açısından belirlenmesi için gerçekleştirilen bu araştırmanın veri toplama aracı olarak odak grup görüşme formu kullanılmıştır. Araştırmanın kuramsal çerçevesi olan programlama dili öğreniminde yaşanan zorluklar dikkate alınarak odak grup görüşmesi soruları hazırlanmıştır. Hazırlanan sorular ilgili alanda uzman kişiler (2 doktor öğretim üyesi ve 1 öğretim görevlisi) tarafından inceledikten sonra onların da görüşü alınarak son şekli verilmiştir. Soruların anlaşılabilirliğini belirlemek için beş kişilik bir grupta pilot uygulama yapılmıştır. Bu sorular yardımıyla yarı yapılandırılmış odak grup görüşmeleri yapılmıştır. Yarı-yapılandırılmış görüşmeler, önceden belirlenmiş temalara görüşme süresince uymak fakat soruların ve soruların sıralarının alınan cevaplara göre görüşme anında değiştirildiği (Merriam, 2013) ve çoğu zaman katılımcıların konuştuğu araştırmacıların dinlediği bir görüşme türüdür (Mertkan, 2015). Yarı yapılandırılmış nitel görüşmeler bireyler ve grupların değerler, varsayımlar, problemler gibi okul kültürünün somut olmayan yönleri hakkında veri toplamak için kullanılmaktadır (Cohen, Manion ve Morrison, 2013).

Odak grup tartışması, sosyal konuların derinlemesine anlaşılması için nitel bir yaklaşım olarak sıklıkla kullanılır (Nyumba, Wilson, Derrick ve Mukherjee, 2018). Cohen vd. (2013) odak grup görüşmelerinin eğitim alanında kullanımının arttığını söylemiştir. Belirli bir konuyu tartışmak için özel olarak seçilmiş katılımcı grubunu bir araya getirerek

düzenlenen odak gruplarının verimliliği grup içerisindeki etkileşime bağlı olmaktadır (Gibbs, 1997). Katılımcılar araştırmacı ile değil de birbirleriyle etkileşime girdikleri için farklı görüşler ortaya çıkabilmektedir. Odak grupların etkileşimli tartışmaları sayesinde karmaşık durumlar hakkında detaylı bilgiler elde edilmekte ve eylemlerin, inançların, tutumların ve algıların arkasındaki nedenleri açıklanabilmektedir (Koskela, Sandström, Mäkinen ve Liira, 2016). Kısa sürede fazla veriye ulaşılabildiğini ancak bire bir görüşmelerden daha kısıtlı veriye ulaşıldığı sonucuna varmıştır (Cohen vd., 2013). Gruplara katılanların sayısının kısıtlı olması, verilerin analiz edilmesinin uzun zaman alması, araştırılacak konuların sınırlı olması ve bazı katılımcıların baskın olmasının diğer katılımcıları etkilemesi olumsuz tarafları olarak görülmektedir.

Odak grup görüşmesinin kaç kişi ile yapılacağı hakkında farklı görüşler vardır. Odak grup görüşmeleri için uygun katılımcı sayısını MacIntosh (1981) 6 ile 10 kişi, Byers ve Wileox (1988) 8 ile 12 kişi, Kitzinger(1995) 4 ile 9 kişi, Goss ve Leinbach (1996) 15 kişi, Morgan (1997) ve Gibbs (1997) 6 ile 12 kişi, Edmunds (2000) ise 8 ile 10 kişiden oluşabileceğini belirtmiştir (Çokluk vd., 2011). Kruger ve Casey (2014) ise odak grup görüşmesinin 5 veya 8 kişiden oluştuğunu ancak en az 4 en fazla 12 kişi ile de yapılabileceğini ifade etmiştir. Buradan da anlaşılacağı üzere alanyazında üzerinde uzlaşmış kesin bir katılımcı sayısı yoktur. Katılımcı sayısını belirleyecek olan araştırmacıdır. Bir araştırmacı, yapacağı araştırmanın niteliğine ve hedef kitlenin durumuna göre katılımcı sayısını ayarlamalıdır.

Odak grup görüşmesi bir yöneticinin (moderatör) kontrolü altında, 8 ila 12 arasındaki katılımcıdan oluşan grubun, önceden belirlenmiş bir amaç doğrultusunda yaptığı karşılıklı görüşmelerdir (Balci, 2016). Kruger ve Casey (2014) odak grup görüşmesinde kabul görülen kuralın üç veya dört gruba görüşme planlamak olduğunu belirtmiştir. Üç veya dört grup ile görüşmeleri yaptıktan sonra doygunluğa ulaşıp ulaşılamadığı belirlenmeli, halen yeni bilgiler alınıyorsa farklı gruplarla görüşmeler yapmaya devam edilmesi gerektiğini söylemiştir.

Nakip ve Yaraş (2017) tarafında belirlenen odak grup görüşmelerinin planlanması aşamaları Şekil 12’de görülmektedir. Görüşme düzenlenmeden önce problemin ve toplantının amacının doğru tespit edilmesini ve önceden belirlenmiş aşamalara göre her şeyin bir plan dahilinde yürütülmesinin gerekmektedir.



Şekil 12. Odak grup görüşmesi planlaması (Nakip ve Yaraş, 2017)

Odak grup görüşmesi, bireysel deneyimden ziyade sistematik bir şekilde bir konu üzerinde odaklaşılarak grupla yürütülen görüşmedir (Ekiz, 2009). Katılımcıların birbirinden etkileşimi sayesinde daha fazla katılımcıyla yapılacak mülakattan fazla bakış açısını ortaya çıkarmaktadır (Mertens, 2014). Ayrıca odak grup görüşmesi insanların bir konu hakkındaki görüşlerini ortaya çıkarmada etkili bir araçtır (Balcı, 2016). Temel amacı, araştırma konusuyla ilgili çeşitli görüşlerin belirlenmesi ve konuların katılımcıların bakış açılarından anlaşılmasıdır. Grup bağlamı, tek seansta birebir görüşmelerden daha geniş kapsamlı bilgi toplamaya yöneliktir. Grup üyeleri, başkalarının fikirlerine ve yorumlarına yanıt vererek birbirlerini etkilemektedir (Hennink, 2007). Kamberelis ve Dimitriadis (2018) odak grup görüşmelerinin diğer veri toplama stratejilere kıyasla bazı farklı özelliklerinin olduğunu belirttiği araştırmasında bu farklılıkları sorunu parçalara ayırmak ve yeniden ifade etmek olarak açıklamıştır. Odak grup görüşmesi diğer yöntemlerden elde edilen sonuçların içerisindeki boşlukları doldurmada etkili olduğunu belirtmiştir.

Odak grup görüşmelerinde önceden belirlenmiş çerçeve sorular aşağıda yer almaktadır:

- Programlama dili size neyi ifade ediyor?
- Kod yazarken karşılaştığınız zorluklar neler?
- Hata ile karşılaşınca neler yapıyorsunuz?
- Kodlama yapmanın zevkli tarafları neler?
- Programlama dili sizce daha kolay nasıl öğretilbilir?

Yukarıda belirtilen sorular çerçevesinde görüşmeler gerçekleştirilmiştir. Yarı yapılandırılmış şekilde gerçekleştirilen görüşmelerde, görüşmenin gidişatından ve öğrencilerin vermiş oldukları cevaplardan yola çıkarak farklı sorular da sorulmuştur. Bu sorulardan bazıları aşağıda yer almaktadır.

- Bilgisayar başında geçirdiğiniz sürenin ne kadarını programlamaya ayırıyorsunuz?
- Programlama yaparken sıkılınca ne yapıyorsunuz?
- Lisede programlama dersini fazla görmeyişinizin üniversitede nasıl bir etkisi oldu?

3.5. Verilerin Analizi

Katılımcılardan alınan görüşme kayıtları bir depolama birimine kaydedilmiş ve analize hazır hale getirilmiştir. Ses kayıtları ayrı dosyalar halinde yazılı metinlere dönüştürülmüştür. Yazıya dönüştürme ve analiz işlemleri araştırmacı tarafından yapılmıştır. Odak grup görüşmelerinde elde edilen mülakatlar kendi içlerinde araştırmacı tarafından analiz edilmiştir. Çalışmada betimsel analiz yöntemi kullanılarak mülakatlar incelenmiş ve yorumlanmıştır. Kemp, Hort ve Hollowood (2018) betimsel analizi duyusal özelliklerin doğasını ve büyüklüğünü nesnel olarak tanımlamak için kullanılan bir yöntem, Erdem (2011) ise elde edilen verilerin özetlenmesi, organize edilmesi ve verilerin bütününe özümsemenin bulunduğu teknik olarak ifade etmiştir. Betimsel analiz esnek ve kişiselleştirilebilir olduğu için kullanım kolaylığı sağlamakta ve bu sayede kullanım devamlılığı sağlamaktadır (Kemp vd., 2018). Temel amacı, verilerde açıklanan kavram ve ilişkilere ulaşmak olan betimsel analiz ile araştırmacı mülakatları karşılaştırmış, karşıt ve benzer görüşleri tespit etmeye çalışmıştır. Belirlenen görüşler

sıklık sayılarına göre frekans dağılımları ve yüzdelik oranları çıkarılarak araştırma bulgularında verilmiştir.



4. BULGULAR

4.1. Katılımcıların Demografik Özellikleri

Görüşmelerden önce katılımcılara katılımcı bilgi formu dağıtılmıştır. Bu formda katılımcıların demografik özellikleri; cinsiyet, sınıf düzeyi, mezun oldukları lise bilgileri, önceki programlama deneyimleri ve kişisel bilgisayara sahip olup olmadığını belirlemek için sorular sorulmuştur.

Katılımcıların demografik özelliklerine ait frekans analizi sonuçları Tablo 7’de sunulmuştur. Buna göre katılımcıların 10’u (%45,5) kadın, 12’si (%54,5) erkektir. Katılımcıların 6’sı (%27,2) 1.sınıfa giderken, 16’sı (%72,8) 2.sınıfa gitmektedir. Katılımcıların lisede öğrenim gördükleri okul türüne göre dağılımları incelendiğinde 13’ü (%59,1) mesleki-teknik lisede, 9’u (%40,9) ise genel (Anadolu Lisesi, İmam Hatip Lisesi, Açık Lise) liseye gittikleri görülmektedir.

Tablo 7. Katılımcıların cinsiyet, sınıf düzeyi ve lise türlerine göre dağılımı

Özellik	Düzy	f	%
Cinsiyet	Kadın	10	45,5
	Erkek	12	54,5
Sınıf	1. Sınıf	6	27,2
	2. Sınıf	16	72,8
Lise mezuniyeti	Mesleki-teknik	13	59,1
	Genel	9	40,9

Katılımcıların daha önceden programlama dersi alıp almadıklarını gösteren programlama deneyimlerine ait frekans analizi sonuçları Tablo 8’de görülmektedir. Öğrencilerin 13’ü (%59) önceden programlama dersi aldıklarını, 9’u (%41) ise önceden programlama dersini hiç almadıklarını ve ilk kez üniversitede gördüklerini belirtmişlerdir.

Tablo 8. Önceki programlama deneyimleri

Geçmiş programlama deneyimi	Düzy	f	%
	Var	13	59,1
	Yok	9	40,9

Tablo 9’da öğrencilerin kişisel bilgisayara sahip olup olmadığının frekans analiz sonuçları yer almaktadır. Katılımcıların 14’ünün (%64) kişisel bilgisayarı bulunmakta iken 8’inin (%36) ise kişisel bilgisayarı bulunmamaktadır. Kişisel bilgisayarı olmayan öğrenciler bütün işlemlerini cep telefonu ile yaptıklarından dolayı ihtiyaç duymadıklarını belirtmiştir.

Tablo 9. Öğrencilerin kişisel bilgisayara sahip olma durumları

Kişisel bilgisayar sahipliği	Düzey	f	%
	Var	14	63,7
	Yok	8	36,3

4.2. Katılımcıların Programlama Dili Öğreniminde Yaşadığı Problemler

Katılımcıların yapılan odak grup görüşmeleri sonunda programlama dili öğrenmede yaşadığı problemlere ilişkin görüşleri; programlama diline ilişkin görüşleri, zevkli taraflarına ilişkin görüşleri, programlama dilinin daha kolay nasıl öğretilebileceğine dair görüşleri, lise eğitiminin programlamaya katkısına yönelik görüşler ve hata ile karşılaşınca yapılanlar şeklinde incelenmiştir.

Katılımcıların programlama diline ilişkin görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 10’da listelenmiştir. Katılımcıların 10’u (%45,5) programlama dilini sayılar ve kodlar olarak gördüğünü, 8’i (%36,4) gelecekte çok önemli olacağını ve 7’si (%31,8) gelecekte programlama dili bilmeyen insan kalmayacağını belirtmiştir. En çok frekansa sahip kategori “sayılar ve kodlardan oluşması” olarak görülmüştür. Ayrıca katılımcıların 1’i (%4,5) etraftaki her şeyin programlanmış olduğunu belirtmiştir.

Tablo 10. Programlama diline ilişkin görüşleri

Bulgular	f	%
Sayılar ve kodlardan oluşması	10	45,5
Gelecekte konuşma dili kadar önemli olacak	8	36,4
Gelecekte programlama dili bilmeyen insan kalmayacak	7	31,8
Sınırsız hayal gücü olması	4	18,2
Kodlarla sanal ortamda birçok şeyi yönetme	3	13,6
Bilgisayara ne yapması gerektiğini anlatmak	2	9,1
Etraftaki her şey programlanmış	1	4,5

Odak grup görüşmelerinde yer alan katılımcıların programlama diline ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

“Gelecekte konuşma dili kadar gerekli bir dil olacağını düşünüyorum” (E4)

“Kendi hayal gücünü kullanarak istediğin programı oluşturup hayata geçirebiliyorsun” (E5)

“Gelecekte çok önemli olacak ve programlama dili bilmeyen kalmayacak” (E7)

“Hayata geçirmek istediğimiz bir projeyi kodlarla bilgisayara dökmek” (K7)

“Geleceğin en kazançlı mesleği olacağını düşünüyorum” (E9)

“İş hayatına atıldığım zaman çok güzel yerlere getirebilir beni” (K4)

“Programlama dersi görüp mezun olduğumuz için diğer bölümlere göre öncelikli olacağımızı düşünüyorum” (K5)

Katılımcıların karşılaştıkları zorluklara ilişkin görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 11’de listelenmiştir. Katılımcıların 11’i (%50) en ufak eksiklikte hata vermesini, 10’u (%45,5) kodların İngilizce olmasını, 5’i (%22,7) ise uygulamayı yapamamasını karşılaştıkları zorluklar olarak belirtmiştir. En yüksek frekansa sahip kategori olarak “en ufak eksiklikte hata vermesi” görülmüştür. Katılımcıların 2’si (%9,1) ise konu başlangıcını kaçırdığında zorluk yaşadığını belirtmiştir.

Tablo 11. Programlamada karşılaşılan zorluklar

Bulgular	f	%
En ufak eksiklikte hata vermesi	11	50
Kodların İngilizce olması	10	45,5
Uygulamayı yapamama	5	22,7
Konu başlangıcını kaçırdığında	2	9,1

Katılımcıların programlama dili yazımında karşılaştıkları zorluklara ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

“Konunun ilk başını kaçırdığımda çok zorlanıyorum” (E4)

“Programı çalıştırırken en ufak bir eksiklikte hata vermesi ve hatayı bulamamak beni çok zorluyor” (E7)

“Küçük bir hatada çalışmaması programın” (E6)

“Düz liseden geldiğimden zor anlıyorum, hoca hızlı geçince yetişemiyorum ve çok zorlanıyorum” (K4)

- “Yabancı kelimeler birbirine çok benziyor ve karışıyor” (K3)*
- “İngilizce olduğundan zorlanıyoruz” (E12)*
- “Uzun kodlar yazıyoruz, bir virgül bile unutunca hata veriyor” (E8)*
- “Noktalama işareti eksik oluyor ve onu bulana kadar başım ağrıyor” (K4)*
- “Uygulamayı yapamadığımız zaman, hatayı bulamayınca sinirleniyorum.” (K1)*
- “Kodlar İngilizce” (K2)*
- “Ders harici tekrar etmediğimden unutuyorum” (K4)*
- “Kodlar Türkçe olmasa bile hatalar Türkçe olsa” (E2)*

Katılımcıların programlama dilinin zevkli taraflarına ilişkin görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 12’de listelenmiştir. Katılımcıların 14’ü (%63,6) yazdığı programın sorunsuz şekilde çalışmasını yani başarmayı, 7’si (%31,8) yazılım alanında kendini geliştirirse ileride kazanç sağlayacağını düşündüğünü ve 2’si (%9,1) verilen uygulamayı daha önce yapabilmeyi yani rekabeti zevkli tarafları olarak belirtmiştir. En yüksek frekansa sahip kategori olarak “başarmak” görülmüştür. Katılımcıların 1’i (%4,5) ise programlamayı oyun olarak gördüğünü belirtmiştir.

Tablo 12. Programlamanın zevkli yanları

Bulgular	f	%
Başarmak	14	63,6
Kazanç	7	31,8
Rekabet	2	9,1
Oyun	1	4,5

Katılımcıların programlama dilinin zevkli taraflarına ilişkin görüşlerinin yer aldığı ifadeleri şu şekildedir:

- “Başarma hissi, bir şeyler yapabildiğini görmek zevk veriyor” (E5)*
- “Kodları yazdıktan sonra ekrana gelen görsel tasarım ve bunun çalışması mutlu ediyor” (E7)*
- “Başardığım zaman ciddi anlamda mutlu oluyorum” (E6)*
- “Çoğu zaman yapamasam da bazen yapabiliyorum ve bu bana güven veriyor neden yapamayayım diyorum kendime” (K4)*
- “Hatalar yapıp bunları düzelttikten sonra programın çalıştığını görmek en zevkli yanı bence” (K3)*

“Yapabildiğim zaman ve rekabet olunca zevkli” (K1)

“Rekabet olunca zevkli” (E1)

“Kodları yazdıktan sonra çalışınca moral oluyor” (K7)

“Ufak işlerle büyük paralar kazanabilme ihtimali” (E9)

“Yazınca bir şeyler çalışıyor oyun gibi geliyor bana ve maddi olarak iyi kazanç sağlayacak olması” (K4)

“Sanal alemde gördüğümüz bir çok şeyin nasıl oluşturulduğunu görüyoruz ve mezun olunca iş imkanı olması” (K5)

“Yeni icat gibi geliyor bana yeni bir şeyler yapmak hoşuma gidiyor” (K8)

“Çalıştığında arkadaşlara hava atıyorum” (K6)

“Tatlı bir rekabet var ilk yapan övünüyor” (E11)

Katılımcıların programlama dilinin nasıl öğretilmesine yönelik görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 13’te listelenmiştir. Katılımcıların 7’si (%31,8) teoriden ziyade uygulama ile daha kolay öğrenileceğini, 4’ü (%18,2) kolay öğrenmenin gerçekleşebilmesi için öğrencinin istekli olması gerektiğini ve 4’ü (%18,2) ise birebir eğitim ile yapılabileceğini söylemiştir. En yüksek frekansa sahip kategori olarak “uygulama” görülmüştür. Katılımcıların 1’i (%4,5) kısa anlatımlar yaparak öğretilbileceğini belirtmiştir.

Tablo 13. Programlamanın nasıl öğretilmesine yönelik görüşler

Bulgular	f	%
Uygulama	7	31,8
İstekli öğrenci	4	18,2
Birebir eğitim	4	18,2
Türkçe eğitim	3	13,6
Kademeli ilerleme	3	13,6
Eğlenceli hale getirilerek	2	9,1
Ezber yapmamak	2	9,1
Kısa anlatım	1	4,5

Katılımcıların programlama dilinin nasıl öğretilmesine yönelik görüşlerinin yer aldığı ifadeleri şu şekildedir:

“Birebir anlatılırsa daha kolay öğrenileceğini düşünüyorum” (E8)

- “Uzun anlatım kafa yoruyor, aralıklı ve basit öğretmek daha iyi” (K7)*
- “Uygulamalı eğitim daha öğretici olur” (E9)*
- “Ezber yapılmadan mantığı öğretilmeli” (K7)*
- “Önce kodları verir diğer uygulamada vermeden yapmalarını isterim” (K6)*
- “Türkçe olarak kod yazdırırdım” (K8)*
- “Benzer örnekler yapıp öğrencilerin kodlamasını bekleyerek ve sonraki konuda öncekine benzer örneklerle ilerleyerek” (E11)*
- “Eğlenceli hale getirilerek” (K2)*
- “Öğrencinin istekli olması gerekmektedir yoksa boş” (E5)*

Katılımcıların lisede aldıkları eğitimin katkısına yönelik görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 14’te listelenmiştir. Katılımcıların 11’i (%50) farkın olduğunu ama zamanla bu farkın kapandığını, 8’i (%36,4) daha önceden dersi almış olmanın öğrenciler arasında bir fark oluşturmadığını, 3’ü (%13,6) ise dersi önceden alan öğrencilerin daha başarılı olduğunu söylemiştir. En yüksek frekansa sahip kategori olarak “fark var ama zamanla kapanıyor” olarak görülmüştür. Lise eğitiminde programlama dersi alan katılımcılar, başlangıçta çok kolaylık yaşadıklarını ama ilerleyen süreçte zorlanmaya başladıklarını belirtmiştir. İlk kez programlama alan katılımcılar ise ilk başta çok zorlandıklarını ama sonradan aradaki farkı kapattıklarını belirtmiştir.

Tablo 14. Lise eğitiminin programlamaya katkısına yönelik görüşler

Bulgular	f	%
Fark var ama zamanla kapanıyor	11	50
Fark yok	8	36,4
Fark var	3	13,6

Katılımcılardan lisede programlama eğitimi alan öğrencilerin aldıkları programlama eğitiminin etkisine yönelik görüşlerinin yer aldığı ifadeleri şu şekildedir:

- “İlk dönem başında fark vardı bizde alt yapı olduğu için sonra farkın kapandığını düşünüyorum” (E8)*
- “Biz anlıyorduk ama lisede görmeyen arkadaş temeli olmadığı için anlamıyordu ” (K7)*
- “Lisede yüzeysel gördüğümüzden ilk başta zorlanmadık ama sonradan fark kapandı” (K5)*

“Temelleri olmadığı için zorlanıyorlar” (K1)

“Temeli olmasa bile bazı arkadaşlar daha iyi yapabiliyor” (K2)

“Bence fark etmiyor burada da başlangıçtan başlıyoruz” (E1)

“Dersi önceden görmemek anlamamak için bir sebep değil, dersi sevmeyeyle alakalı bence önceden görse de sevmiyorsa başarılı olamaz” (E4)

“Kulak aşinalığı oluyor diğer arkadaşlardan kolay anlıyoruz buda kolaylık sağlıyor” (K3)

“İlk haftalarda anlamakta zorlandı arkadaşlar ama bizde lisede ilk zamanlar zorlandık sonra fark kapaniyor, kendini geliştirmek isteyen geliştiriyor, derse kendini vermeyenler hala anlayamıyor” (E5)

“Lisede çok çalışıyorduk ama şimdi çalışma isteği yok” (K1)

“Lisede programlama dersini çok fazla gördük ve zorlandık bundan dolayı programlamadan soğudum” (E1)

Katılımcılardan lisede programlama eğitimi almayan öğrencilerin görüşlerinin yer aldığı ifadeleri şu şekildedir:

“Lisede görmedim ama fark kapandı ikinci dönem” (K7)

“Programlama hiç görmedim ama meslek lisesinden gelenlerden iyi yaptığımı düşünüyorum” (E8)

“Temeli olan arkadaşlar hemen anlıyordu ama olmayanlar zorlanıyordu ama zamanla bizimde temelimiz oluştu” (K4)

“Onlar daha bilgili söylediğinizi hemen yapıyorlar biz zorlanıyoruz” (K4)

“İlk zamanlar çok fark vardı ama zamanla kapandı bu fark” (E7)

Katılımcıların kod yazarken hata ile karşılaşınca ne yaptıklarına yönelik görüşlerinin kodlanması sonucunda oluşan kategoriler Tablo 15’te listelenmiştir. Katılımcıların 8’i (%36,4) hata ile karşılaşınca çözmeye çalıştığını, 6’sı (%27,3) ara verip daha sonra tekrar hatayı arayarak çözmeye yoluna gittiğini ve 5’i (%22,7) ise uzmana ya da bilen bir arkadaşına danıştığını belirtmiştir. En yüksek frekansa sahip kategori olarak “hatayı çözmeye çalıştıkları” görülmüştür. Katılımcıların 1’i (%4,5) ise tekrar yazmayı denediğini belirtmiştir.

Tablo 15. Programlamada hata ile karşılaşınca yapılanlar

Bulgular	f	%
Çözmeye çalışıyorum	8	36,4
Ara veriyorum	6	27,3
Uzmana danışıyorum	5	22,7
Hata penceresine bakıyorum	4	18,2
İnternette araştırıyorum	2	9,1
Tekrar yazıyorum	1	4,5

Katılımcıların kod yazarken hata ile karşılaşınca neler yaptıklarına yönelik görüşlerinin yer aldığı ifadeleri şu şekildedir:

“Deneyerek sorunu çözmeye çalışıyorum, çözemeyince size danışıyorum” (E8)

“Hata yapınca üstüne gitmiyorum daha kötü oluyor” (K7)

“Biraz ara verdikten sonra kafamı toparlayıp hatayı bulmaya çalışıyorum” (E9)

“Üstüne gidince hatayı bulamıyorum, ara verince bulması daha kolay oluyor” (K7)

“Size sorarız hocam” (E10)

“Önce hata kodlarına bakıyorum, sonra ders notlarına yine yapamazsam size soruyorum hocam” (K8)

“Hata penceresine bakıyorum ayrıca çizgi oluyor hata olan yerde oraya odaklanıyorum” (E12)

“Kendimi zorlayarak çözmeye çalışıyorum çözemezsem size soruyorum” (K2)

“Error list bölümünden hatanın nerede olduğunu ve ne yapmam gerektiğini bakıp yapmaya çalışıyorum” (E4)

“Yapamadıkça daha çok üstüne gidiyorum, internette araştırıp bulmaya çalışıyorum” (E7)

“Error bölümüne bakıyorum, bulamayınca sizden yardım alıyorum” (E6)

“Hatayı bulmaya çalışıyorum bulamazsam yanımdaki bilen arkadaştan yardım istiyorum, yine bulamazsak sizden yardım istiyorum” (K4)

“Kitaptan araştırıyorum, error list bakıyorum ama her hatayı göstermiyor, tekrar baştan yazmayı deniyorum” (K3)

Katılımcıların programlama diline ne kadar süre ayırıyorsunuz sorusuna ilişkin cevaplarının sonucunda oluşan kategoriler Tablo 16’da listelenmiştir. Görüşmeler

esnasında öğrenciler tarafından gelen yanıtlar doğrultusunda sorulan farklı soruların analizleri diğer görüşler teması olarak Tablo 16’da verilmiştir. Katılımcıların 10’u (%45,5) sadece sınav zamanlarında, 8’i (%36,3) yalnızca ders saatinde, 4’ü (%18,1) günde birkaç saat vakit ayırabildiğini belirtmiştir. En yüksek frekansa sahip değer “sınav zamanlarında” olarak görülmüştür.

Tablo 16. Programlamaya ayrılan vakit

Bulgular	f	%
Sınav zamanlarında	10	45,5
Ders saatinde	8	36,3
Günde 1-2 saat	4	18,1

Katılımcıların programlama diline ayırdıkları sürelerle ilişkin görüşlerinin yer aldığı ifadeler şu şekildedir:

“Doğrusu sadece sınav dönemlerinde zaman ayırıyorum” (K3)

“Bilgisayarım yok sadece derslerde” (E5)

“Sınav haftaları kodları hatırlamak amacıyla bakıyorum” (E9)

“Lisede programlamaya çok zaman harcıyorduk şimdi sadece derslerde” (E1)

“Ders saatleri dışında en fazla bir saat” (K9)

“Okulda birkaç saat” (E11)

5. TARTIŞMA VE SONUÇ

“Ön Lisans Öğrencilerinin Programlama Dili Öğrenimine Yönelik Genel Görüşleri, Karşılaştıkları Zorluklar ve Önerilerinin İncelenmesi” konulu tez çalışması ile öğrencilerin programlama diline bakış açıları, karşılaştıkları zorluklar, geçmiş deneyimlerinin ne yönde etkilere sahip olduğu tespit edilmeye çalışılmıştır. Araştırmada ulaşılan bulgular incelendiğinde öğrencilerin programlama öğretimi sürecinde karşılaştıkları öncelikli zorluklar; çok fazla kod içeriyor olması, hata ayıklamada sıkıntı yaşanması ve programlama dilinin Türkçe olmaması olarak öne çıkmıştır.

Çalışmada yer alan katılımcıların nesne tabanlı programlama diline ilişkin genel görüşleri; %45,5'i programlama dilini sayılar ve kodlar olarak gördüğünü, %16,4'ü gelecekte konuşma dili kadar önemli olacağını belirtmiştir. Gomes ve Mendez (2007) programlamanın karmaşık sözdiziminin programlama eğitiminde yaşanan zorluklardan birisi olduğu sonucuna ulaşmıştır. Nesnel programlama dili kavramlarının soyut ve karmaşık olması öğrenmeyi zorlaştırmaktadır. Programlamaya yeni başlayanlar için Alice, Scratch ve Blockly gibi blok programlama dilleri ile eğitim verilmesi teklif edilmektedir (Stolee ve Fristoe, 2011; Wu, Chang ve He, 2010;).

Katılımcıların nesnel programlama dili öğrenmede sıklıkla karşılaşılan sorunlar; %50'si en ufak bir eksiklikte hata almalarını, %45,5'i kodların İngilizce olmasını karşılaştıkları zorluklar olarak ifade etmişlerdir. Kinnunen ve Malmi (2008), programlama dersinde 340 üniversite öğrencisi üzerinde gerçekleştirdiği çalışmada en çok zorlanılan konuları, hata bulma ve kod yazma olarak ifade etmektedir. Bosse ve Gerosa (2017) yaptığı çalışmada öğrencilerin program yazarken hata ile karşılaşınca hata mesajlarını anlamadıklarını ve kafalarının karıştığı sonucuna ulaşmıştır. Arabacıoğlu, Bülbül ve Filiz (2007) programlama yaparken yabancı dil kullanımının zor bir süreç olduğu sonucuna varmışlardır.

Çalışmada yer alan katılımcıların programlama dili öğrenmenin zevkli tarafları; %63,6'sı başardığında mutlu olduğunu, %31,8'i ise kazanç getirecek bir meslek olduğunu söylemiştir. Öğrencilerin ifadelerini dikkate aldığımızda programlama öğretimi üzerinde öğrencilerin başarı hissi ve tatmin duygusu ile gelecekte ortaya çıkması umulan maddi beklenti ve kazançların daha etkili olduğu anlaşılmaktadır. Esteves ve Mendes (2004)

öğrencilerin nesne yönelimli programlamanın temel kavramlarını öğrenmelerine ve programlama yeteneklerini geliştirmelerine yardımcı olmak için yaptıkları çalışmada öğrencilerin çalışan bir çözüme ulaştıklarında, deneyimleri ve güvenleri artacağından dolayı öğrenmenin kolaylaşacağını savunmaktadır.

Çalışmada yer alan öğrencilerin programlama dili öğretimi nasıl olmalıdır sorusuna yönelik görüş olarak %31,8'i uygulama ağırlıklı olması gerektiğini, %18,2'si birebir eğitim verilerek daha etkili öğretim yapılabileceğini belirtmiştir. Öğrenciler, yapılan mülakatlar neticesinde programlama öğretiminde teorik bilgiden daha çok uygulamalı eğitimin daha verimli olduğu, programlama dilinin kolaydan zora doğru aşama aşama ve uzman kontrolünde bire bir verildiğinde başarı düzeyinin arttığına yönelik fikir beyan etmişlerdir. Saracaloğlu ve Çelik (2018), “Web Tasarımı ve Programlama Dersi Öğretiminde Proje Tabanlı Öğrenme Yaklaşımı Kullanımının Öğrenci Görüşlerine Göre Değerlendirilmesi” adlı çalışmalarında öğrencilerin istenilen hedeflere ulaşmasında ve dersi anlamalarında uygulama ağırlıklı olmasını en önemli etken olarak gözlemlemişlerdir.

Çalışmada yer alan öğrencilerin lise döneminde programlama dersi almanın programlama dili öğrenmeye katkısı var mıdır sorusuna; %50'si geçmiş deneyimlerin programlama öğretiminde başlangıç seviyesinde bir etkisi olsa da zamanla bu farkın kapandığını, %36,4'ü ise hiç fark olmadığını belirtmektedir. Öğrencilerin sadece %13,6'sı geçmiş deneyimlerinin programlama öğreniminde farka neden olduğunu ifade etmiştir. Konu üzerine gerçekleştirilmiş olan daha önceki araştırmalarda tam anlamı ile bir fikir birlikteliği söz konusu değildir. Kimi çalışmalarda benzer sonuçlara ulaşıldığı görülmekte iken (Aldağ ve Tekdal, 2015; Atabaş, 2018; Erol ve Kurt, 2017; Pillay ve Jugoo 2005; Yağcı, 2016), kimi çalışmalarda ise bunun tersi sonuçlara ulaşıldığı görülmüştür (Altun ve Mazman, 2012). Örneğin, Altun ve Mazman (2012) programlama geçmiş deneyimleri ve öz yeterlilikleri arasında anlamlı bir ilişkiye ulaşmıştır. Erol ve Kurt (2017) ise lise mezuniyet türünün programlamaya karşı tutumlarına bir etkisi olmadığı sonucuna ulaşmıştır. Atabaş (2018) ise yaptığı çalışmada geçmiş programlama deneyiminin düşük seviyede katkı sağladığını iddia etmekle birlikte programlamayla ilgilenilen süre arttıkça başarının da arttığını belirtmiştir.

Çalışmaya katılan öğrencilerin kodlama esnasındaki karşılaştıkları zorluklardaki davranışları nasıldır sorusuna; %36,4'ü çözmeye çalıştığını, %27,3'ü ara verip daha sonra tekrar denediğini belirtmiştir. Programlama öğretimi alan öğrencinin program yazma aşamasında kendisinden istenen görevlerin yerine getirilmesinde ortaya çıkan problemlerle bizzat kendisinin yüzleşmesi, çözememesi durumunda ara vererek tekrar denemesi, problemin ortadan kalkmaması halinde ise öğretici ile bire bir iletişime geçerek uzman yardımından faydalanmasının etkili olacağı görülmüştür.

Çalışmaya katılan öğrencilerin programlamaya ayırdıkları vakit olarak; %45,5'i sadece sınav zamanlarında, %36,3'ü ders saatinde cevaplarını vermiştir. Lise eğitiminde bilgisayar bölümünden gelen öğrenciler lisede çok fazla bilgisayar ile ilgilendikleri için artık sıkıldıklarını ve ilgilenmediklerini belirtmiştir. Bednarik ve Tukiainen'e (2004) göre, programlamayı seven öğrencilerin kendini geliştirmek ve dahi iyi uygulamalar yazabilmek için bilgisayar başında fazla zaman ayırdıkları; programlamaya önyargılı olan öğrencilerin programlamaya ekstra zaman ayırmadıkları görülmüştür.

Literatür taramasından ve çalışmadan elde edilen bulgulardan anlaşıldığı üzere esas dikkat edilmesi gereken noktanın öğrencilerin daha iyi nasıl öğrendiği ve onlara nasıl programlama öğretimi yapıldığıdır. Bu kapsamda öğrencilerin programlama dilinin yabancı olması, hata ayıklama sürecinin zorluğu ve öğrenci sayısının fazla olmasından kaynaklanan öğrenci başına düşen sürenin az olması programlama dili öğretiminin zorlukları olarak karşımıza çıkmaktadır. Öğrencilerin konuya ilişkin ilgilerinin yüksekliği, başarı hissi ve kazanç beklentisinin öğretim sürecini olumlu yönde etkileyeceği düşünülebilir. Öğretim süreci esnasında öğrencinin konuyu basitten zora doğru yaparak öğrenmesine katkı sağlayacak şekilde yönlendirilmesinin programlama öğretiminde başarıya katkı sağlayacağı değerlendirilmektedir. Ayrıca kimi çalışmalarda da belirtildiği üzere programlama öğretiminde karşılaşılan zorlukları aşmak için görsel programlama dilleri kullanımının öğrenmede etkili olacağı düşünülmektedir (Adams, 2014; Herbert, 2010).

Çalışma sonucunda elde edilen bulgulara dayanılarak programlama öğretimi ve öğreniminde aşağıdaki hususların dikkate alınmasının uygun olacağı değerlendirilmektedir.

1. Programlama öğretiminde başlangıç seviyesinde çok fazla kod içermeyen programlama dilleri ile eğitim verilebilir.
2. Başlangıç aşamasında nesnel programlama yerine görsel programlamaya ağırlık verilmesi faydalı olabilir.
3. Türkçe desteği olan programlama dilleri kullanılabilir.
4. Karşılaşılması muhtemel hatalara ve hata ayıklamaya derslerde daha fazla yer ayrılması uygun olabilir.
5. Geçmiş deneyimleri olmayan öğrencilerin başlangıç aşamasında ek dersler ile desteklenmesi sağlanabilir.
6. Programlama eğitimi verilecek öğrencilerin üst düzey bilişsel becerilere sahip olması gerekliliği dikkate alınabilir.
7. Programlama derslerinde öğrenci sayıları sınırlı tutulabilir.

KAYNAKÇA

- Açık Akademi. (2018). *Microsoft Türkiye açık akademi nedir?* <https://www.acikakademi.com/portal/default.aspx> adresinden 13 Eylül 2018 tarihinde edinilmiştir.
- Adams, J. (2014). *Alice 3 in action: Computing through animation*. Australia: Cengage Learning.
- Afyon Kodluyor. (2018). *Afyon kodluyor projesi nedir?* <http://www.afyonkodluyor.org/afyon-kodlama-robotik-egitimi.aspx> adresinden 14 Eylül 2018 tarihinde edinilmiştir.
- Akçay, A. ve Çoklar, A. N. (2016). Bilişsel becerilerin gelişimine yönelik bir öneri: Programlama eğitimi. A. İşman, H. F. Odabaşı ve B. Akkoyunlu (Ed.), *Eğitim Teknolojileri Okumaları* (ss. 121-139). Ankara: TOJET.
- Akyüz, A. O. (2018). Çocuklara programlama nasıl öğretilbilir?. *Prof. Dr. Emine AKYÜZ'e armağan*. Ankara: Pegem Akademi.
- Aldağ, H. ve Tekdal, M. (2015). Bilgisayar kullanımı ve programlama öğretiminde cinsiyet farklılıkları. G. Ağrıdağ (Ed.), *1. Uluslararası Çukurova Kadın Çalışmaları Kongresi* (ss. 236-243). Adana: Çukurova Üniversitesi.
- Algan, S. (2010). *Her yönüyle C#* (12. baskı). İstanbul: Pusula Yayıncılık.
- Alican, F. (2006). *Dünyada ve Türkiye'de yazılım sektörü*. İstanbul: İletişim Yayınları.
- Allsop, Y. (2016). A reflective study into children's cognition when making computer games. *British Journal of Educational Technology*, 47(4), 665-679.
- Altun, A. ve Mazman, S. G. (2012). Programlamaya ilişkin öz yeterlilik algısı ölçeğinin Türkçe formunun geçerlilik ve güvenirlik çalışması. *Eğitimde ve Psikolojide Ölçme ve Değerlendirme Dergisi*, 3(2), 297-308.
- Anastasiadou, S. D., & 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.
- Arabacıoğlu, T., Bülbül, H. İ. ve Filiz, A. (2007). *Bilgisayar programlama öğretiminde yeni bir yaklaşım*. Akademik Bilişim 2007 sempozyumunda sunuldu, Dumlupınar Üniversitesi, Kütahya.
- Arkan, Ş. (2017). *C programlama dili* (8. baskı). Ankara: Seçkin Yayıncılık.
- Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From scratch to "real" programming. *ACM Transactions on Computing Education (TOCE)*, 14(4), 25.

- Aspinal, B. (2015). *Kodlama öğrenmek için 10 neden* <http://brianaspinall.com/10-reasons-to-teach-coding-sketchnote-by-sylviaduckworth/> adresinden 04 Kasım 2018 tarihinde edinilmiştir.
- Atabaş, S. (2018). *Programlama başarısını etkileyen bazı faktörlerin incelenmesi*. Yayınlanmamış yüksek lisans tezi, Ondokuz Mayıs Üniversitesi, Samsun, Türkiye.
- Aytekin, A., Sönmez Çakır, F., Yücel, Y. ve Kulaöz, İ. (2018). Geleceğe yön veren kodlama bilimi ve kodlama öğrenmede kullanılabilecek bazı yöntemler. *Avrasya Sosyal ve Ekonomi Araştırmaları Dergisi*, 5(5), 24-41.
- Baist, A., & Pamungkas, A. S. (2017). Analysis of student difficulties in computer programming. *VOLT: Jurnal Ilmiah Pendidikan Teknik Elektro*, 2(2), 81-92.
- Bal, H. Ç. (2009). *Bilgisayar ve internet kullanımı XP* (16. baskı). Trabzon: Murathan Yayınevi.
- Balcı, A. (2016). *Açıklamalı eğitim yönetimi terimleri sözlüğü* (3. Baskı). Ankara: Pegem Akademi.
- Bartlett, J., & Bruno Jr, D. (2003). *Programming from the ground up*. Orange Grove Books.
- Baş T., Çamır, M. ve Özmalda, B. (2008). *Nitel araştırma yöntemleri*. Ankara: Seçkin Yayınevi.
- Baz, F. Ç. (2018). Çocuklar için kodlama yazılımları üzerine karşılaştırmalı bir inceleme. *Current Research in Education*, 4(1), 36-47.
- Bednarik, R., & Tukiainen, M. (2004). Visual attention and representation switching in java program debugging: A study using eye movement tracking. *In Proceedings of 16th Annual Psychology of Programming Interest Group Workshop (PPIG'04), Institute of Technology Carlow, Ireland, April 5-7, pp. 159-169.*
- Benzer, A. İ. ve Erümit, A. K. (2017). The analysis of the graduate theses related to programming instruction. *Öğretim Teknolojileri & Öğretmen Eğitimi Dergisi*, 6(3), 99-110.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, 72, 145-157.
- Bilim, Sanayi ve Teknoloji Bakanlığı. (2017). *Türkiye yazılım sektörü stratejisi ve eylem planı (2017-2019)*. <https://www.sanayi.gov.tr/DokumanGetHandler.ashx?dokumanId=663ce026-f803-4802-9267-73e84dbef61c> adresinden 12 Kasım 2017 tarihinde edinilmiştir.

- Bilişim Garaj Akademisi*. (2014). <https://www.bilisingarajakademisi.com/Bizi-Kesfedin/BGA-Dongusu/Nedir> adresinden 09 Ağustos 2018 tarihinde edinilmiştir.
- Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., & Koller, D. (2014). Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming. *Journal of the Learning Sciences*, 23(4), 561-599.
- Bosse, Y., & Gerosa, M. A. (2017). Why is programming so difficult to learn?: Patterns of difficulties related to programming learning Mid-Stage. *ACM SIGSOFT Software Engineering Notes*, 41(6), 1-6.
- Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., & Miller, P. (1997). Mini-languages: A way to learn programming principles. *Education and Information Technologies*, 2(1), 65-83.
- Büyüköztürk, Ş., Çakmak, E. K., Akgün, Ö. E., Karadeniz, Ş. ve Demirel, F. (2018). *Bilimsel araştırma yöntemleri* (21. baskı). Ankara: Pegem Akademi.
- Cevahir, H. ve Özdemir, M. (2007). Programlama öğretiminde karşılaşılan zorluklara yönelik öğretmen görüşleri ve çözüm önerileri. *1. Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumu* (ss. 317-332). Malatya: İnönü Üniversitesi.
- Chao, P. Y. (2016). Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. *Computers & Education*, 95, 202-215.
- Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education*, 109, 162-175.
- Code Club (2014). *About code club*. <https://www.codeclub.org.uk/about> adresinden 09 Ağustos 2018 tarihinde edinilmiştir.
- Cohen, L., Manion, L., & Morrison, K. (2013). *Research methods in education* (7th ed.). London: Routledge.
- Costa, C. J., & Aparicio, M. (2014). Evaluating success of a programming learning tool. In *Proceedings of the International Conference on Information Systems and Design of Communication* (pp. 73-78). ACM.
- Çetin, İ. (2013). Visualization: A tool for enhancing students' concept images of basic object-oriented concepts. *Computer Science Education*, 23(1), 1-23.
- Çokluk, Ö., Yılmaz, K. ve Oğuz, E. (2011). Nitel bir görüşme yöntemi: Odak grup görüşmesi. *Kuramsal Eğitimbilim Dergisi*, 4(1), 95-107.
- Deitel, P. J., & Deitel, H. M. (2008). *C++ how to program*. Pearson: Prentice Hall.

- Demir, F. (2015). *Programlama öğretiminde eğitsel programlama dilinin farklı kullanımlarının programlama başarısı ve kaygısına etkisi*. Yayınlanmamış doktora tezi, Atatürk Üniversitesi, Erzurum, Türkiye.
- Demirbilek, M. (2014). The ‘digital natives’ debate: An investigation of the digital propensities of university students. *Eurasia Journal of Mathematics, Science & Technology Education*, 10(2), 115-123.
- Demirer, V. ve Sak, N. (2016). Dünyada ve Türkiye’de programlama eğitimi ve yeni yaklaşımlar. *Eğitimde Kuram ve Uygulama*, 12(3), 521-546.
- Derus, S. R. M., & Ali, A. Z. M. (2012). Difficulties in learning programming: Views of students. *1st International Conference on Current Issues in Education* (pp. 74-79). Indonesia: Yogyakarta State University.
- Dikici, M. (2017). *C++ programlama dili* (3. baskı). Ankara: Seçkin Yayıncılık.
- Ekiz, D. (2009). *Bilimsel araştırma yöntemleri*. Ankara: Anı Yayıncılık.
- Erdem, D. (2011). Türkiye’de 2005-2006 yılları arasında yayımlanan eğitim bilimleri dergilerindeki makalelerin bazı özellikler açısından incelenmesi: Betimsel bir analiz. *Eğitimde ve Psikolojide Ölçme ve Değerlendirme Dergisi*, 2(1), 140-147.
- Erdem, O. (2017). *Google’den yazılım eğitimi desteği*. <http://www.meb.gov.tr/googledan-yazilim-egitimi-destegi/haber/14587/tr> adresinden 01 Kasım 2017 tarihinde edinilmiştir.
- Erol, O. ve Kurt, A. A. (2017). BÖTE bölümü öğrencilerinin programlamaya karşı tutumlarının incelenmesi. *Mehmet Akif Ersoy Üniversitesi Eğitim Fakültesi Dergisi*, 1(41), 314-325.
- Ersoy, H., Madran, R. O. ve Gülbahar, Y. (2011). Programlama dilleri öğretimine bir model önerisi: Robot programlama. *Akademik Bilişim 2011 Konferansı* (ss. 785-790). Malatya: İnönü Üniversitesi.
- Eryılmaz, S. (2003). *Algoritma tasarlama ve programlamaya giriş*. Ankara: Detay Yayıncılık.
- Esteves, M., & Mendes, A. J. (2004). A simulation tool to help learning of object oriented programming basics. In *34th Annual Frontiers in Education, 2004. FIE 2004*. (pp. F4C-7). IEEE.
- Exworthy, M., Peckham, S., & Powell, M. (Eds.). (2012). *Shaping health policy: case study methods and analysis*. Great Britain: Bristol.
- Gibbs, A. (1997). Focus groups. *Social Research Update*, 19(8), 1-8.

- Github, (2017). *Türkiye programlama dili istatistikleri*.
https://alpcanaydin.github.io/github-stats-forturkey/?utm_source=elektroyazilim#/
adresinden 02 Ağustos 2018 tarihinde edinilmiştir.
- Gomes, A., & Mendes, A. J. (2007). Learning to program difficulties and solutions. *In International Conference on Engineering Education*. Portekiz: University of Coimbra.
- Govender, I., & Grayson, D. J. (2008). Pre-service and in-service teachers' experiences of learning to program in an object-oriented language. *Computers & Education*, 51(2), 874-885.
- Gowrishankar, S., & Veena, A. (2018). *Introduction to python programming*. Boca Raton: CRC Press.
- Gundurao, H. K., Manjunath, N. S., & Nachappa, M. N. (2010). *Computer technology and computer programming*. Mumbai: Global Media.
- Gülbahar, Y. ve Kalelioğlu, F. (2018). Bilişim teknolojileri ve bilgisayar bilimi: Öğretim programı güncelleme süreci. *Milli Eğitim Dergisi*, 47(217), 5-23.
- Gültepe, A. (2018). Kodlama öğretimi yapan bilişim teknolojileri öğretmenleri gözüyle öğrenciler kodluyor. *Uluslararası Liderlik Eğitimi Dergisi*. 2(2), 50-60.
- Hacker Can. (2018). *Milli Kodlama Eğitimi Platformu*. <http://www.hackercan.com/tr/>
adresinden 09 Ağustos 2018 tarihinde edinilmiştir.
- Han, B., Bae, Y., & Park, J. (2016). The effect of mathematics achievement variables on scratch programming activities of elementary school students. *International Journal of Software Engineering and Its Applications*, 10(12), 21-30.
- Hancock, D. R., & Algozzine, B. (2016). *Doing case study research: A practical guide for beginning researchers*. Teachers College Press.
- Heintz, F., Mannila, L., Farnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. *In Frontiers in Education Conference (FIE)*, 2016 IEEE (pp. 1-9). IEEE.
- Hendrix, R., & Weeks, M. (2018). First programming language for high school students. *In Society for Information Technology & Teacher Education International Conference* (pp. 1896-1903). Association for the Advancement of Computing in Education (AACE).
- Hennink, M. M. (2007). *International focus group research: A handbook for the health and social sciences*. Cambridge University Press.

- Herbert, C. W. (2010). *An introduction to programming using Alice 2.2*. Boston: Course Technology Press.
- HourOfCode (2018). <https://hourofcode.com/tr> adresinden 15 Kasım 2018 tarihinde edinilmiştir.
- Kamberelis, G., & Dimitriadis, G. (2018). Focus group research. In *The Oxford Handbook of Qualitative Research*.
- Karaçay, T. ve Karaçay A. (2016). *Hiç bilmeyenler için C# ile programlamaya giriş* (2. baskı). Ankara: Seçkin Yayıncılık.
- Kemp S. E., Hort J. & Hollowood T. (Eds.). (2018). *Descriptive analysis in sensory evaluation*. New Jersey: John Wiley & Sons.
- Kinnunen, P., & Malmi, L. (2008). CS minors in a CS1 course. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 79-90). ACM.
- Kitzinger, J. (1995). Qualitative research: Introducing focus groups. *British Medical Journal*, 311, 299-302.
- Koç, M. (2017). Learning analytics of student participation and achievement in online distance education: A structural equation modeling. *Educational Sciences: Theory and Practice*, 17(6), 1893-1910.
- Koçak, G. H. (2006). *Bilgisayar teknolojisi ve programlama eğitim programlarının sorunları*. Yüksek Lisans Tezi, Çukurova Üniversitesi, Adana, Türkiye.
- Kod Adı 2023. (2018). *Kod adı 2023 proje özeti*. <https://kodadi2023.gsb.gov.tr/> adresinden 09 Ağustos 2018 tarihinde edinilmiştir.
- Kodlama Festivali (2018). <http://www.kodlamafestivali.com/> adresinden 08 Ağustos 2018 tarihinde edinilmiştir.
- Kodla(Ma)nisa, (2015). *Kodla(Ma)nisa gelecek kodla gelecek*. <http://www.kodlamanisa.gov.tr/> adresinden 8 Ağustos 2018 tarihinde edinilmiştir.
- Konuk, M. ve Öztürk, A. (2010). Üniversite - sanayi işbirliği ve teknokentlere bakış. *Cumhuriyetimizin 100. Yılına Doğru Üniversite Vizyonumuz* (ss. 123-128), Ankara: Türk Eğitim-Sen.
- Kordaki, M. (2010). A drawing and multi-representational computer environment for beginners' learning of programming using C: Design and pilot formative evaluation. *Computers & Education*, 54(1), 69-87.
- Koskela, T., Sandström, S., Mäkinen, J., & Liira, H. (2016). User perspectives on an electronic decision-support tool performing comprehensive medication reviews-a

- focus group study with physicians and nurses. *BMC Medical Informatics and Decision Making*, 16(1), 6.
- Kukul, V. ve Gökçearslan, Ş. (2014). Scratch ile programlama eğitimi alan öğrencilerin problem çözme becerilerinin incelenmesi. 8. *Bilgisayar ve Öğretim Teknolojileri Sempozyumu* (ss. 58-63). Edirne: Trakya Üniversitesi.
- Köse, U. ve Tüfekçi, A. (2015). Algoritma ve akış şeması kavramlarının öğretiminde akıllı bir yazılım sistemi kullanımı. *Pegem Eğitim ve Öğretim Dergisi*, 5(5), 569-586.
- Krueger, R. A. (2014). *Focus groups: A practical guide for applied research*. California: Sage publications.
- Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *ACM*, 37(3), 14-18.
- Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K-8 curriculum. *Acm Inroads*, 5(4), 64-71.
- Lewis, C. M., & Shah, N. (2012). Building upon and enriching grade four mathematics standards with programming curriculum. *43rd ACM Technical Symposium on Computer Science Education* (pp. 57-62). ACM.
- Lye, S. Y., & Koh, J. H. L. (2018). Case studies of elementary children's engagement in computational thinking through scratch programming. In *Computational Thinking in the STEM Disciplines* (pp. 227-251). Springer, Cham.
- Mazman, S. G. (2013) *Programlama performansını etkileyen faktörlerin bilişsel tabanlı bireysel farklılıklar temelinde modellenmesi*. Yayınlanmamış doktora tezi, Hacettepe Üniversitesi, Ankara, Türkiye.
- Merriam, S. B. (2013). *Nitel araştırma: Desen ve uygulama için bir rehber*. (Çev. Sellahattin Turan). Ankara: Nobel Yayın Dağıtım.
- Mertens, D. M. (2014). *Research and evaluation in education and psychology: Integrating diversity with quantitative, qualitative, and mixed methods*. CA: Sage publications.
- Mertkan, Ş. (2015). *Karma araştırma tasarımı*. Ankara: Pegem Akademi.
- Milli Eğitim Bakanlığı, (2017). *Google'dan yazılım eğitimi desteği*. <http://www.meb.gov.tr/googledan-yazilim-egitimi-destegi/haber/14587/tr> adresinden 05 Kasım 2017 tarihinde edinilmiştir.

- Milli Eğitim Bakanlığı. (2018). *Bilişim teknolojileri ve yazılım dersi öğretim programı (Ortaokul 5 ve 6. Sınıflar)* <http://mufredat.meb.gov.tr/ProgramDetay.aspx?PID=37> adresinden 25 Kasım 2017 tarihinde edinilmiştir.
- Milli Eğitim Bakanlığı, (2018). *Bilgisayar bilimi dersi öğretim planı*. <http://mufredat.meb.gov.tr/Dosyalar/2018120203611364-BILGISAYAR%20BILIMI%20DERSI%20OGRETIM%20PROGRAMI.pdf> adresinden 30 Ekim 2018 tarihinde edinilmiştir.
- Nakip, M. ve Yaraş, E. (2017). *Pazarlamada araştırma teknikleri* (4. baskı). Ankara: Seçkin Yayıncılık.
- Nyumba, T. O., Wilson, K., Derrick, C. J., & Mukherjee, N. (2018). The use of focus group discussion methodology: Insights from two decades of application in conservation. *Methods in Ecology and Evolution*, 9(1), 20-32.
- Ozan Leylum, Ş., Odabaşı, H. F. ve Kabakçı Yurdakul, I. (2017). Eğitim ortamlarında durum çalışmasının önemi. *Eğitimde Nitel Araştırmalar Dergisi*, 5(3), 369-385.
- Özdiñç, F. ve Altun, A. (2014). Bilişim teknolojileri öğretmeni adaylarının programlama sürecini etkileyen faktörler. *İlköğretim Online*, 13(4), 1531-1541.
- Özmen, B. & Altun, A. (2014). Undergraduate students' experiences in programming: Difficulties and obstacles. *Turkish Online Journal of Qualitative Inquiry*, 5(3), 1-27.
- Özyurt, H. ve Özyurt, Ö. (2017). Görsel programlama dersinde proje tabanlı öğrenme deneyimine ilişkin öğrenci görüşlerinin incelenmesi. *Türk Bilgisayar ve Matematik Eğitim Dergisi*, 8(2), 244-260.
- Pillay, N. & Jugoo, V. R. (2005). An investigation into student characteristics affecting novice programming performance. *ACM SIGCSE*, 37(4), 107-110.
- Rahmat, M., Shahrani, S., Latih, R., Yatim, N. F. M., Zainal, N. F. A., & Ab Rahman, R. (2012). Major problems in basic programming that influence student performance. *Procedia-Social and Behavioral Sciences*, 59, 287-296.
- Renumol, V., Jayaprakash, S., & Janakiram, D. (2009). Classification of cognitive difficulties of students to learn computer programming. *Indian Institute of Technology, India*, 12.
- Robincod.org. (2014). *Neden Robincod*. <http://www.robincod.org/neden-robincod/> adresinden 07 Aralık 2018 tarihinde edinilmiştir.
- Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137-172.

- Robinson, J. D. (1995). *The effects of two approaches to basic programming on the achievement and field dependent students*. Unpublished doctoral thesis, Texas Tech University, Texas, USA.
- Saracaloğlu, A. S. ve Çelik, B. (2018). Web tasarımı ve programlama dersi öğretiminde proje tabanlı öğrenme yaklaşımı kullanımının öğrenci görüşlerine göre değerlendirilmesi. *Kastamonu Eğitim Dergisi*, 26(2), 323-333.
- Saygıner, Ş. ve Tüzün, H. (2017). Programlama eğitiminde yaşanan zorluklar ve çözüm önerileri. *19. Akademik Bilişim Konferansı*. Aksaray: Aksaray Üniversitesi.
- Sayın, Z. ve Seferoğlu, S. (2016). Kodlama eğitimi ve kodlamanın eğitim politikalarına etkisi. *18. Akademik Bilişim Konferansı*. Aydın: Adnan Menderes Üniversitesi.
- Sebesta, R. W. (2016). *Concepts of programming languages* (11th ed.). New Jersey: Pearson Education.
- Seferoğlu, S. S. (2007). İlköğretim bilgisayar dersi öğretim programı: Eleştirel bir bakış ve uygulamada yaşanan sorunlar. *Eurasian Journal of Educational Research*, (29), 99-111.
- Shin, S., Park, P., & Bae, Y. (2013). The effects of an information-technology gifted program on friendship using scratch programming language and clutter. *International Journal of Computer and Communication Engineering*, 2(3), 246.
- Subaşı, M. ve Okumuş, K. (2017). Bir araştırma yöntemi olarak durum çalışması. *Atatürk Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, 21(2), 419-426.
- Stolee, K. T., & Fristoe, T. (2011). Expressing computer science concepts through Kodu game lab. *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, (pp. 1-26). ACM.
- Tekdal, M. (2016). *Temel programlama*. Ankara: Seçkin Yayıncılık.
- TIOBE, (2018). *En popüler programlama dilleri*. <https://www.tiobe.com/tiobe-index/> adresinden 03 Ağustos 2018 tarihinde edinilmiştir.
- Toal, R., Rivera, R., Schneider, A., & Choe, E. (2016). *Programming language explorations*. New York: Chapman and Hall/CRC.
- Thomas, G. (2015). *How to do your case study*. California: Sage publications.
- Tungut, H. (2013). *Algoritma ve programlama mantığı* (2. baskı). İstanbul: Kodlab Yayınevi.

- Türk Bilişim Derneği, (2018). *TBD, yerli yazılım sanayi ve ambargo*. <http://www.keyofchange.com/tr/2054/TBD%2C%20Yerli%20Yaz%C4%B1%C4%B1m%20Sanayi%20ve%20Ambargo/> adresinden 03 Ağustos 2018 tarihinde edinilmiştir.
- Wang, W. (2008). *Beginning programing all-in-one desk reference for dummies*. Indiana: Wiley Publishing.
- Weinberg, G. M. (2011). *The psychology of computer programming: Silver anniversary eBook edition*. New York: Smashwords.
- Winslow, L. E. (1996). Programming pedagogy - Psychological overview. *SIGCSE Bulletin*, 28.
- Wu, W. Y., Chang, C. K., & He, Y. Y. (2010). Using scratch as game-based learning tool to reduce learning anxiety in programming course. *In Proceedings of Global Learn Asia Pacific*, (pp. 1845-1852).
- Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies*, 21(3), 559-588.
- Yağcı, M. (2016). Bilişim teknolojileri öğretmen adaylarının ve bilgisayar programcılığı öğrencilerinin programlamaya karşı tutumlarının programlama öz yeterlik algılarına etkisi. *İnsan Bilimleri Dergisi*, 13(1), 1418-1432.
- Yang, J., Wong, G. K., & Dawes, C. (2018). An exploratory study on learning attitude in computer programming for the twenty-first century. *In New Media for Educational Change* (pp. 59-70). Springer, Singapore.
- Yang, T. C., Hwang, G. J., Yang, S. J., & Hwang, G. H. (2015). A two-tier test-based approach to improving students' computer-programming skills in a web-based learning environment. *Journal of Educational Technology & Society*, 18(1), 198-210.
- Yaşar, E. (2012). *Algoritma ve programlamaya giriş* (4. baskı). Trabzon: Murathan Yayıncılık.
- Yecan, E., Özçınar, H. ve Tanyeri, T. (2017). Bilişim teknolojileri öğretmenlerinin görsel programlama öğretimi deneyimleri. *İlköğretim Online*, 16(1), 377-393.
- Yıldırım, A. ve Şimşek, H. (2013). *Sosyal bilimlerde nitel araştırma yöntemleri* (9. baskı). Ankara: Seçkin Yayınevi.
- Yılmaz, F. (2013). *Meslek yüksekokulu öğrencilerinin programlama başarısını etkileyen faktörlerin incelenmesi*. Yayınlanmamış yüksek lisans tezi, Gazi Üniversitesi, Ankara, Türkiye.

- Yilmaz, R., Topu, F., Goktas, Y., & Coban, M. (2013). Social presence and motivation in a three-dimensional virtual world: An explanatory study. *Australasian Journal of Educational Technology*, 29(6).
- Yin, R. K. (2017). *Case study research and applications: Design and methods*. California: Sage publications.
- Yumuřak, N. ve Adak, F. (2018) *Programlama dillerinin prensipleri*, Ankara: Seękin Yayıncılık.
- Yükseköğretim Kurulu, (2018). *Gençlere ücretsiz kodlama eğitimi verilecek*. <http://www.yok.gov.tr/web/guest/genclere-ucretsiz-kodlama-egitimi> adresinden 14 Ağustos 2018 tarihinde edinilmiştir.
- Yükseltürk, E. ve Altıok, S. (2018). Blok tabanlı programlama. Y. Gülbahar (Ed.), *Bilgi işlemsel düşünmeden programlamaya* (ss. 241-263). Ankara: Pegem Akademi.
- Zhong, B., Wang, Q., Chen, J., & Li, Y. (2017). Investigating the period of switching roles in pair programming in a primary school. *Journal of Educational Technology & Society*, 20(3), 220-233.



Ek A. Odak Grup Görüşmesi Soruları

Çerçeve Sorular

1. Programlama dili size neyi ifade ediyor?
2. Kod yazarken karşılaştığınız zorluklar neler?
3. Hata ile karşılaşınca neler yapıyorsunuz?
4. Kodlama yapmanın zevkli tarafları neler?
5. Programlama dili sizce daha kolay nasıl öğretilbilir?

Diğer Sorular

- Bilgisayar başında geçirdiğiniz sürenin ne kadarını programlamaya ayırıyorsunuz?
- Programlama yaparken sıkılınca ne yapıyorsunuz?
- Lisede programlama dersini fazla görmemenizin üniversitede nasıl bir etkisi oldu?

ÖZGEÇMİŞ

Adı Soyadı: Abdurrahman ÖKTE

Doğum Yeri ve Yılı: Bucak, 1984

Medeni Hali: Evli

Yabancı Dili: İngilizce

Eğitim Durumu (Kurum ve Yıl)

Lise:

Bucak Anadolu Lisesi, 2000-2003

Lisans:

Dokuz Eylül Üniversitesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi, 2004-2008

Çalıştığı Kurum/Kurumlar ve Yıl

Selçuk Üniversitesi, Doğanhisar Meslek Yüksekokulu, 2011-2012

Mehmet Akif Ersoy Üniversitesi, Hikmet Tolunay Meslek Yüksekokulu, 2012-2018

Mehmet Akif Ersoy Üniversitesi, Emin Gülmez Teknik Bilimler Meslek
Yüksekokulu, 2018- devam ediyor