
GAZİ ÜNİVERSİTESİ
Gazi
1926

PROGRAMLAMA DİLİ ÖĞRETİMİNDE ALICE PROGRAMININ ÖĞRENCİLERİN AKADEMİK BAŞARILARI, PROBLEM ÇÖZME BECERİSİ ALGILARI, MOTİVASYONLARI VE PROGRAMLAMAYA HAZIR BULUNUŞLUK DÜZEYLERİ ÜZERİNE ETKİSİ

Ceren Baştemur Kaya

DOKTORA TEZİ

BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ

EĞİTİM BİLİMLERİ ENSTİTÜSÜ

KASIM, 2018

TELİF HAKKI VE TEZ FOTOKOPİ İZİN FORMU

YAZARIN

Adı : Ceren

Soyadı : Baştemur Kaya

Bölümü : Bilgisayar ve Öğretim Teknolojileri Eğitimi

İmza :

Teslim Tarihi :

TEZİN

Türkçe Adı : Programlama Dili Öğretiminde Alice Programının Öğrencilerin Akademik Başarıları, Problem Çözme Becerisi Algıları, Motivasyonları ve Programlamaya Hazır Bulunuşluk Düzeyleri Üzerine Etkisi

İngilizce Adı : Effect of Use of Alice Software on Students' Academic Achievement, Problem Solving Skill Perceptions, Motivations and Readiness Level to Programming in Computer Programming Teaching

ETİK İLKELERE UYGUNLUK BEYANI

Tez yazma sürecinde bilimsel ve etik ilkelere uyduğumu, yararlandığım tüm kaynakları kaynak gösterme ilkelerine uygun olarak kaynakçada belirttiğimi ve bu bölümler dışındaki tüm ifadelerin şahsıma ait olduğunu beyan ederim.

Yazar Adı Soyadı : Ceren Baştemur Kaya

İmza :

JÜRİ ONAY SAYFASI

Ceren BAŞTEMUR KAYA tarafından hazırlanan "Programlama Dili Öğretiminde Alice Programının Öğrencilerin Akademik Başarıları, Problem Çözme Becerisi Algıları, Motivasyonları ve Programlamaya Hazır Bulunuşluk Düzeyleri Üzerine Etkisi" adlı tez çalışması aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Gazi Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı'nda Doktora tezi olarak kabul edilmiştir.

Danışman : Doç. Dr. Hasan ÇAKIR

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Gazi Üniversitesi ……

Başkan : Prof. Dr. Süleyman Sadi SEFEROĞLU

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Hacettepe Üniversitesi ……

Üye : Prof. Dr. Ebru KILIÇ ÇAKMAK

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Gazi Üniversitesi ……

Üye : Prof. Dr. Tolga GÜYER

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Gazi Üniversitesi ……

Üye: Doç. Dr. Yasemin DEMİRASLAN ÇEVİK

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Hacettepe Üniversitesi ……

Tez Savunma Tarihi : 02/11/2018

Bu tezin Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı'nda Doktora tezi olması için şartları yerine getirdiğini onaylıyorum.

Prof. Dr. Selma YEL
Eğitim Bilimleri Enstitüsü Müdürü ……

Sevgili Oğlum Kerem'e

TEŞEKKÜR

“Programlama Dili Öğretiminde Alice Programının Öğrencilerin Akademik Başarıları, Problem Çözme Becerisi Algıları, Motivasyonları ve Programlamaya Hazır Bulunuşluk Düzeyleri Üzerine Etkisi” başlıklı tezin başlamasına ve yürütülmesine vesile olan, çalışmalar sırasında sabır ve desteğini esirgemeyen, eleştirileri ile tezime yön veren tez danışmanım Doç. Dr. Hasan ÇAKIR'a ve tez izleme komitemde yer alan değerli hocalarım Prof. Dr. Ebru KILIÇ ÇAKMAK ve Doç. Dr. Yasemin DEMİRASLAN ÇEVİK'e, ayrıca tez savunma jürisi üyeleri hocalarım Prof. Dr. Süleyman Sadi SEFEROĞLU ve Prof. Dr. Tolga GÜYER'e yorumları, önerileri ve destekleri için teşekkürlerimi sunarım.

Çalışmalarım sırasında yaşadığım tüm zorluklarda desteğini daima hissettiğim eşim Ebubekir KAYA’ya ve tezimin her aşamasında motivasyon kaynağım olan oğlum Kerem'e, dualarını hiçbir zaman eksik etmeyen anneme, babama ve kardeşime çok teşekkür ederim.

PROGRAMLAMA DİLİ ÖĞRETİMİNDE ALICE PROGRAMININ ÖĞRENCİLERİN AKADEMİK BAŞARILARI, PROBLEM ÇÖZME BECERİSİ ALGILARI, MOTİVASYONLARI VE PROGRAMLAMAYA HAZIR BULUNUŞLUK DÜZEYLERİ ÜZERİNE ETKİSİ

(Doktora Tezi)

Ceren Baştemur Kaya

GAZİ ÜNİVERSİTESİ

EĞİTİM BİLİMLERİ ENSTİTÜSÜ

KASIM 2018

ÖZ

Bu çalışmanın amacı Alice programı ile programlama öğretiminin; öğrencilerin akademik başarısına, problem çözme becerisi algısına, motivasyonuna ve programlamaya hazır bulunuşluk düzeyine etkisini, Alice programı ile ilgili öğrenci değerlendirmelerini belirlemektir. Araştırmada karma yöntemlerden sıralı açıklayıcı desen kullanılmıştır. Araştırmanın nicel boyutunda öntest-sontest kontrol gruplu yarı deneysel model kullanılmış, nitel boyutunda deney grubu öğrencileriyle Alice programı ile programlama öğretimine yönelik odak gruplu görüşmeler yapılmıştır. Araştırma, 2015-2016 eğitim-öğretim yılı Nevşehir Hacı Bektaş Veli Üniversitesi Meslek Yüksekokulu Bilgisayar Teknolojileri Bölümü'nde okuyan ve Nesne Tabanlı Programlama I dersini alan 63 birinci sınıf öğrencisi ile 8 hafta boyunca sürmüştür. Normal öğretim öğrencileri deney grubu,

ikinci öğretim öğrencileri karşılaştırma grubu olarak, yansız atama yoluyla atanmıştır. Hem deney hem de karşılaştırma grubunda Gagné'nin dokuz aşamalı öğretim modeli göz önünde bulundurularak öğretim tasarlanmıştır. Deney grubundaki öğrenciler Alice ve NetBeans programları ile karşılaştırma grubundaki öğrenciler NetBeans programı ile Java programlama dilini öğrenmişler, ilgili programlarla örnek ve uygulamalar yapmışlardır. Araştırmanın nicel verileri akademik başarı testi, problem çözme becerisi algısı ölçeği, motivasyon ölçeği, programlamaya hazır bulunuşluk düzeyi belirleme testi; nitel verileri yarı yapılandırılmış görüşme formu aracılığı ile toplanmıştır. Araştırma sonucunda nicel veriler analiz edildiğinde; akademik başarı, problem çözme becerisi algısı ve motivasyon bakımından deney ve karşılaştırma grubu arasında anlamlı bir farklılık bulunmamıştır. İki grupta yer alan öğrenciler Java programlama dillini öğrenmeye odaklanmışlardır. İlgili derste başarılı olabilmek için çaba sarf etmişlerdir. Ders kapsamında problem çözme becerisi gerektiren örnek ve uygulamalar yapmışlardır. Bu nedenler göz önünde bulundurulduğunda ilgili değişkenlerde anlamlı bir fark çıkmamış olabilir. Programlama öğretiminde Alice programı kullanımı ile programlamaya hazır bulunuşluk düzeyi arasında deney grubu lehine olumlu yönde anlamlı bir ilişki tespit edilmiştir. Programlama dillerinde kullanılan temel kavramlar benzerdir. Her programlama dili öğretiminde temel kavramlar yeniden öğrencilere sunulmaktadır. Alice programının programlamaya hazır bulunuşluk düzeyini anlamlı olarak olumlu yönde etkilediği sonucu, öğretim ortamlarında önceden programlama dersi alınsa dahi her programlama dersinin başında programlama kavramlarının ilk kez öğretiliyormuş gibi tekrar öğretilmesiyle harcanan zamanı en aza indirip programlama becerilerini artırmaya yönelik çalışmaların yapılmasına yoğunlaşılmasını sağlayabilir. Nitel veriler analiz edildiğinde öğrencilerin Alice programının temel kod kavramlarının öğrenilmesini ve programlama mantığını anlamayı kolaylaştırdığını, programlamayı öğrenme isteğini arttırdığını ifade ettikleri görülmüştür. Ancak Alice programında kod yazılamaması nedeniyle kod yazma becerisinin geliştirilememesini ve Türkçe dil desteğinin olmamasını olumsuz yönde eleştirdikleri; Alice programını, ileri seviye programlama için yetersiz gördükleri belirlenmiştir. Ayrıca öğrencilerin çoğu, ileride programlama dili öğretmeleri gerekirse Alice programını öğrenme ortamı olarak kullanacaklarını belirtmiştir. Araştırma sonuçları göz önünde bulundurulduğunda Alice programı ile programlama öğretimi, öğrenme ortamlarını zenginleştirip programlama öğrenimini kolaylaştırabilir. Öğrencileri derste aktif kılıp programlama temeli oluşturmalarında yarar sağlayabilir. Öğrencilerin programlama becerileri arttırılabilir. Özellikle meslek yüksekokullarında teorik ağırlıklı eğitim verilmesinin yanı sıra uygulamaya dönük çalışmaların kalitesi bu şekilde arttırılabilir.

Anahtar Kelimeler : Alice programı, meslek yüksekokulu, programlama ortamları, programlama öğrenimi

Sayfa Adeti : xviii + 193

Danışman : Doç. Dr. Hasan ÇAKIR

EFFECT OF USE OF ALICE SOFTWARE ON STUDENTS' ACADEMIC ACHIEVEMENT, PROBLEM SOLVING SKILL PERCEPTIONS, MOTIVATIONS AND READINESS LEVEL TO PROGRAMMING IN COMPUTER PROGRAMMING TEACHING

(Ph.D Thesis)

Ceren BAŞTEMUR KAYA

GAZI UNIVERSITY

GRADUATE SCHOOL OF EDUCATIONAL SCIENCES

NOVEMBER 2018

ABSTRACT

The purpose of this study is to determine the effect of programming teaching via Alice software on student's academic achievement, problem solving skill perception, motivation and readiness level to programming, opinions about Alice software. The sequential descriptive design, which is a type of the mixed method was used in the research. The pretest posttest control group quasi experimental design model was utilized in the quantitative part of the study, focus group interviews related to teaching of programming with Alice software were conducted with the experimental group students in the qualitative part of study. The study lasted for 8 weeks with 63 freshman students studying at Nevşehir Hacı Bektaş Veli University, Vocational School, Computer Technology Department, taking the Object Based Programming I course in 2015-2016 academic year. Day education students were assigned as experimental group and evening education students were assigned as comparison group by neutral assignment. In both groups, the teaching

was designed considering Gagne's nine events of instruction model. Students in the experimental group learned Java programming language by using Alice and NetBeans programs, the comparison group students learned Java programming language by using NetBeans program, both groups performed the examples and the applications with related programs. The quantitative data of this study were collected through academic achievement test, problem solving skill perception scale, motivation scale, readiness academic achievement test to programming and on other hand the qualitative data were collected through the semi-structured interview form. When quantitative data were analyzed as a result of the research, there was no significant difference between the experimental group and the comparative group in terms of academic achievement, problem solving skill perception and motivation. The students in both groups focused on learning the Java programming language. The students made efforts to be successful in the related course. Within the scope of the course, the students performed examples and applications that require problem solving skills. Considering these reasons, there may be no significant difference in the related variables. A significant relationship in favor of the experimental group was determined between programming instruction based on Alice software and the readiness level to programming. The basic concepts used in programming languages are similar. The basic concepts in each programming language learning are presented to the students again. The fact that the Alice software has a significant positive effect on the readiness level to programming, can help to minimize the time spent re-teaching as if taught for the first time for learning the programming concepts in teaching environments, even if pre-programming courses are taken and the students can be concentrated to making efforts to increase the programming skills. When qualitative data were analyzed, students expressed that Alice software facilitates to learn the basic code concept and programming logic, and increases the desire to learn programming. But students criticized that the practicality of code writing cannot be improved due to not being able to write code and lack of Turkish language support in Alice program. It has been determined that students saw Alice software for advanced programming inadequately. In addition, many of the students have indicated that they will use Alice software as a learning environment if they will have to teach programming languages in the future. Considering the results of the research, programming teaching with Alice software can enrich learning environments and facilitate programming learning. This program can be useful for building students' programming skills and make students active during the course. It can enhance the programming skills of students. Besides the theoretical weighted education given especially in vocational schools, the quality of application-based studies can be increased in this way.

Key Word : Alice program, vocational school, programming environments, programming learning

Page Number : xviii + 193

Supervisor : Assoc. Prof. Dr. Hasan ÇAKIR

İÇİNDEKİLER

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

SİMGELER VE KISALTMALAR LİSTESİ

n	Veri sayısı
f	Verilerin tekrarlama sıklığı
p	Anlamlılık düzeyi
%	Yüzde
SS	Standart Sapma
$\bar{X}$	Ortalama
sd	Serbestlik derecesi
t	t testi puanı
KR-20	Kuder Richardson-20
K-S	Kolmogorov-Smirnov testi
KT	Kareler toplamı
KO	Kareler ortalaması

BÖLÜM I

GİRİŞ

Bu bölümde araştırmanın problem durumu, amacı, önemi, varsayımları ile araştırmada kullanılan tanımlar ve kısaltmalara yer verilmiştir.

Problem Durumu

Günümüz bilgi çağında teknolojinin gelişmesiyle birlikte bireylerden beklenen özellikler değişmekte, yenilikçi bir bakış açısına sahip olmaları ve rekabet ortamlarında ön sıralarda yer almaları beklenmektedir. Özellikle eleştirel düşünme, problem çözme, iletişim, öz düzenleme, takım çalışması gibi becerilere sahip olunması gereklilik haline gelmiştir. Bireyin sahip olduğu bilgi ve becerilerini yeni durumlara uyarlayabilmesi ve farklı ürünler ortaya çıkarabilmesi önem kazanmıştır (Partnership for 21st Century Skills [P21], 2009).

Yenilikçi ve fark yaratan ürünler ortaya çıkarılabilmesinin özellikle bilişim sektöründe önemli bir yeri bulunmaktadır. Bilgiyi üretmede, saklamada, işlemede ve paylaşmada sunduğu imkânlarla; yeni pazarlara erişim konusunda sağladığı avantajlarla bilgi ve iletişim teknolojileri rekabet gücünün şekillenmesinde çok kritik bir rol oynamaktadır. Bu rekabet ortamında önemli olan bilginin nasıl işlendiğidir. Yeni ürünler geliştirilerek, farklı teknolojiler ortaya çıkarılarak bilişim sektöründe yer bulunabilir. Bu durumda programlamanın önemi ön plana çıkmakta ve programlamaya olan ilgi artmaktadır. Programlama öğrenimi, günümüz toplumunda var olan önemli sorunların çözümü ve yeni projelerin tasarlanması için önemli stratejiler geliştirmeyi desteklemektedir (Resnick, 2013).

Günümüzde bireyler bilgisayarla ilgili bir bölümde öğrenim görmeseler bile kendilerini bu alanda geliştirerek kendi programlarını oluşturmak istemektedir. Ayrıca bilişim sektörünün önde gelen firmaları da bireyleri kendi programlarını, kendi oyunlarını geliştirmeleri yönünde teşvik etmektedir. Dikkatlerin programlamaya çevrilmesi özellikle eğitim alanında yapılan birçok çalışmanın programlama eğitimi ve öğretimine yoğunlaşmasını sağlamıştır (Scott, 2005).

Programlama öğrenim sürecinde özellikle bilişsel etkinlikler ön plana çıkmaktadır. Birey program tasarımında zihinsel betimlemeler yapar, program akışını tasarlayarak program kodlarını yazar, hata ayıklar, kavramsal bilgileri yapılandırır. Temel kavramların (döngüler, koşul cümleleri vb.) kullanımlarını geliştirerek program yazma becerisini geliştirir. Program yazma becerileri soyutlama, problem çözme, genelleme, transfer gibi başlıklar altında ifade edilmektedir (Gomes & Mendes, 2007; Gundurao, Manjunath & Nachappa, 2010; Rogalski & Samurçay, 1991).

Programlama dili öğrenim sürecinde gerekli beceriler arasında problem çözme becerisi önemli bir yer almaktadır. Çünkü bir programın geliştirilmesi sürecinde sadece programlama dilinin kurallarını bilmek yetmemekte, başarılı bir programlama için problemin nasıl çözülmesi gerektiğinin iyi bir şekilde analiz edilmesi gerekmektedir. Bilgisayar programlarının nasıl yazıldığının öğrenilmesi için öğrencilerin problem çözme becerileri geliştirilmelidir (Gundurao vd., 2010). Problem çözme doğuştan itibaren öğrenilen ve okul yıllarında geliştirilen bir beceridir (Miller & Nunn, 2001). Programlama bir problem çözme sürecidir. Bu yüzden problem çözme becerisi programlama başarısını etkileyen en önemli ve en çok etkisi incelenen faktörlerden bir tanesidir. Özellikle öğrencilerin problem çözme becerisi algısı bir programı geliştirebilmelerinde önemli yer tutmaktadır. Yapılan çalışmalarda problem çözme ve analitik düşünme becerisinin programlama başarısını doğrudan etkilediği görülmüştür (Grant, 2003; Henderson, 1986; Hung, 2008; Ismail, Ngah & Umar, 2010; Linn & Dalbey, 1985; Pillay & Jugoo, 2005).

İlgili alanyazın incelendiğinde, programlama sürecinin zor olarak kabul edilebildiği görülmektedir ve bu zorluk eğitim ortamlarında öğrencilerin programlama öğrenmeye karşı isteklerini olumsuz etkileyebilmektedir. Eğitim sırasında bazı öğrencilerin ilgili derse, konuya ya da karşılaşılan probleme çözüm üretmekte istekli oldukları gözlenirken, bazı öğrencilerin derslerde isteksiz oldukları, karşılaştıkları problemlerle mücadele etmek

yerine daha çok kaçmayı tercih ettikleri gözlemlenmektedir (Akbaba, 2006). Motivasyon, kişinin istekli hale gelerek, harekete geçmesini sağlar (Deci, Koestner & Ryan, 1999). Yapılan araştırmalar göstermektedir ki motivasyon programlama başarısını arttıran bir faktördür (Jiau, Chen & Ssu, 2009; Matthíasdóttir, 2006).

Her programlama dilinin kendine özel kodlama sistemi bulunsa da, hepsinde kullanılan ortak temel kavramlar mevcuttur (Sureau, 2012). Bu yüzden programcılarda aranan temel özelliklerden biri genel programlama bilgisidir (Eryılmaz, 2003). Programcılık eğitimi verilirken yalnızca kullanılan programlama diline ait söz diziminin öğretilmesi, öğrencilerin o dilin yapısını öğrenmesini sağlar. Ancak başarılı bir programcı olabilmeleri için temel kavramları çok iyi öğrenmeleri gerekir (Kak, 2009). Temel kavramlar öğrencilerin programlamaya hazır bulunuşluk düzeyi ile ilişkilidir. Yapılan araştırmalar öğrencilerin programlamaya hazır bulunuşluk düzeyleri arttıkça programla başarılarının arttığını göstermektedir (Byrne & Lyons, 2001; Goold & Rimmer, 2000; Holden & Weeden, 2003; Konvalina, Stephens & Wileman, 1983).

Programlama eğitimi Türkiye'de ve dünyada farklı bölümlerde eğitim alan öğrencilere verilmektedir. Ancak alanyazında programlama öğreniminin zor olduğu ile ilgili tartışmalar bulunmaktadır (Akbaba, 2006; Resnick, Flanagan, vd., 2009) ve öğrencilerin programlama derslerindeki akademik başarılarının yüksek olduğu çalışmalar bu düşünceleri destekler nitelikte sınırlı sayıdadır (Robins, Rountree & Rountree, 2003; Solmaz, 2014). Özellikle ilk programlama dili öğrenimi öğrencileri zorlayabilirken, programlama becerisini geliştirmek de karmaşık bir süreç olarak görülebilmektedir (Ford Jr, 2015; Gomes & Mendes, 2007; Rogalski & Samurçay, 1991). Programlama eğitimi zor bir süreç olarak görülebildiği için öğrencilerin motivasyonu her zaman üst seviyede olmamaktadır ve karşılarına çıkan zor problemler öğrencilerin motivasyon algılarını azaltarak kendilerini yetersiz hissetmelerine neden olabilmektedir. Öğrenciler bir programı geliştirmeye başlarken programın geneline odaklanamamakta, program içerisindeki kod bloklarında kaybolabilmektedir (Berge, Borge, Fjuk, Kaasbøll & Samuelsen, 2003; Bucci, Long & Weide, 2001). Özellikle soyut program yapılarını, somut problemlerin çözümünde kullanırken zorluk çekebilmektedirler (Gomes & Mendes, 2007). Bir programın bütününden ziyade, program içerisindeki modüllere takılıp kalabilmektedirler (Berge vd., 2003; Bucci vd., 2001). Ayrıca öğrenciler bir problemin çözümü için algoritma

oluştururken ve çözümü planlarken zorlanabilmektedir. Kodların hangi sırayla yazılması gerektiği, probleme nereden başlanacağı ve hangi kod parçalarıyla çözüme ulaşılabileceği yönünde sorunlar yaşayabilmektedirler (Cooper, Dann & Pausch, 2000b). Bu problemlerin yanı sıra programlama sürecinde yaşanan en büyük sorunlardan biri de öğrencilerin önceden programlama dersi almalarına rağmen yeni bir programlama dili öğrenirken zorlandıklarının görülebilmesidir. Halbuki öğrenilen programlama dilleri farklı olsa da programlama temel kavramları ve programlama yapıları benzerdir. Her defasında programlamanın temel yapılarının öğrenilmesi için zaman harcanılması, programlama becerisinin geliştirilmesine ve yeni programların ortaya çıkarılmasına odaklanılmasının önüne geçebilmektedir. Bu durum öğrencilerin programlamaya hazır bulunuşluk düzeylerinin düşük olduğunu gösterebilir.

Yukarıda açıklandığı gibi programlama karmaşık bir süreçtir ve programlama başarısını etkileyen problem çözme becerisi algısı, motivasyon, programlamaya hazır bulunuşluk düzeyi gibi temel yapılar bulunmaktadır. Bunun yanında yıllar geçtikçe programlama dillerinin öğrenilmesinin kolaylaştırılması amacıyla dillerin yapısının değiştirilmesine rağmen, programlama öğretiminde pek fazla değişiklikler olmadığı da görülmektedir. Genellikle öğretmenler, yeni kontrol ve veri yapılarını öğrencilere anlatır, bir kaç örnek gösterir ve onlardan karşılaştıkları problemleri çözmelerini bekler. Ancak öğrencilerin bu konuda belirttikleri sıkıntı, öğretmen anlatırken anladıkları fakat kendi problemlerine çözüm üretemedikleri yönündedir (Garner, 2003).

Programlama derslerindeki bu problemler göz önünde bulundurularak programlama dersi verilen bölümlerden biri olan, 2017-2018 öğretim yılı Yükseköğretim Bilgi Yönetim Sistemi verilerine göre %92'lik bir oranla Türkiye'nin bilişim ve iletişim teknolojileri sektörünün yetişmiş insan gücünü karşılayan meslek yüksekokullarında bulunan bilgisayar teknolojileri bölümü öğrencilerinden 130 öğrenciye mesleki bilgilerini belirlemek amacıyla, geçerliği ve güvenirliği sağlanmış dört temel ders kapsamında dört akademik başarı testi uygulanmıştır. Bu derslerden biri Programlama Temelleri dersidir. 20 sorudan oluşan bu testten öğrencilerin başarıları %36 olarak bulunmuştur (Baştemur Kaya & Çakır, 2015). Yine öğrencilerin geçmiş programlama sınavı başarı notları incelenmiş ve araştırmayı destekler nitelikte düşük olduğu görülmüştür. Bilgisayar teknolojileri bölümünün temel amacı bilişim teknolojileri alanında tüm kamu ve özel sektör

kuruluşlarının ihtiyaçlarını karşılayabilen bireyler yetiştirmektir. Ancak öğrencilerin öğrendiklerini yeni ve farklı durumlarda kullanamamaları nedeniyle sektör bu öğrencilerden beklentilerini karşılayamamaktadır. Bu yüzden mezun öğrencilerin istihdamı her geçen yıl azalmaktadır.

Bu araştırmalar ve problemler göz önünde bulundurulduğunda programlama eğitiminde bilişsel süreçleri destekleyen eğitim ortamlarının hazırlanması, öğrencilerin programlama mantığını daha kolay bir şekilde kavramalarını ve soyutlamayı daha iyi bir şekilde yapmalarını sağlayacak bir öğretim aracının kullanılması gerekliliği ortaya çıkmaktadır. Alanyazın incelendiğinde özellikle öğrencilerin yazdıkları kodların görsel olarak ne işe yaradığını görmelerini sağlayan ve kodlarla görsellerin eşgüdümlü çalışmasına imkân veren görselleştirme araçlarının programlama derslerinde kullanıldığı görülmektedir (Cervesato, 2008; Cliburn, 2008; Cooper, Dann & Pausch, 2000a; Cooper vd., 2000b; Doherty & Kumar, 2009; Gomes & Mendes, 2007; Howard, Evans, Courte & Bishop-Clark, 2006; Karsten, Kaparthi & Roth, 2005; Lin & Kuo, 2010; Maloney vd., 2004; Radošević, Orehovački & Lovrenčić, 2009; Wang, Mei, Lin, Chiu & Lin, 2009). Bu görselleştirme araçlarından biri üç boyutlu öğrenme ortamı olan Alice programıdır.

Alice programı üç boyutlu ve etkileşimli bir görsel programlama öğrenme ortamıdır. Alice programındaki yönergeler, nesne yönelimli programlama dillerindeki kod sözdizimi ile uyumludur (Hayat, Al-Shukaili & Sultan, 2017). Kullanıcılar Alice programında kod yazmak yerine ilgili kodları sürükleyip çalışma ekranına bırakarak programlarını oluşturabilmektedir (Cooper vd., 2000a). Böylece kod hataları ile uğraşmak yerine kodların ne işe yaradığına odaklanmaktadırlar (Cliburn, 2008).

Alice programı ile ilgili yapılan araştırmalar incelendiğinde; öğrencilerin kodların işlevlerini görerek programlama dilinin özellikle temel kavramlarını anlamalarına yardımcı olduğu, öğrencilerin algoritmik düşünme becerilerini geliştirdiği, öğrencilerin derse olan ilgi ve katılımlarını arttırmasıyla motivasyonlarını olumlu yönde etkilediği belirlenmiştir (Cliburn, 2008; Cooper vd., 2000b; Wang vd., 2009; Zhang, De Pablos & Zhang, 2012; Zhang, de Pablos & Zhu, 2012).

Bu çalışmada Alice programı kullanılarak, Nesne Tabanlı Programlama I dersi kapsamında Java programlama dili öğretiminde yardımcı araç olarak Alice programı kullanımının

öğrencilerin akademik başarıları, motivasyonları, problem çözme becerisi algıları ve programlamaya hazır bulunuşluk düzeyleri üzerindeki etkisi araştırılmıştır. Ayrıca Alice programı kullanımında engelleyici ve kolaylaştırıcı faktörleri tespit etmek amacıyla, olumlu ve olumsuz öğrenci değerlendirmeleri belirlenmiştir.

Araştırmanın Amacı

Bu çalışma Alice programının öğrencilerin programlama dersindeki akademik başarılarına, motivasyonlarına, problem çözme becerisi algılarına, programlamaya hazır bulunuşluk düzeylerine etkisini, Alice programı kullanımında engelleyici ve kolaylaştırıcı faktörleri belirlemek amacıyla yapılmıştır. Bu genel amaç çerçevesinde aşağıdaki sorulara cevap aranmıştır:

1. Alice programını kullanan öğrenciler ile kullanmayan öğrencilerin;
    a. Akademik başarı puanları arasında anlamlı bir fark var mıdır?
    b. Problem çözme becerisi algıları arasında anlamlı bir fark var mıdır?
    c. Motivasyonları arasında anlamlı bir fark var mıdır?
    d. Programlamaya hazır bulunuşluk düzeyleri arasında anlamlı bir fark var mıdır?
2. Alice programı kullanımında engelleyici ve kolaylaştırıcı faktörler nelerdir?

Araştırmanın Önemi

Bilişim sektöründe var olabilmek ve değişen dünyada söz sahibi olabilmek için programlama önemli bir alandır. Ancak programlama dersleri öğrencileri zorlayabilmekte ve başarı oranları düşük olabilmektedir (Baştemur Kaya & Çakır, 2015; Robins vd., 2003; Solmaz, 2014). Özellikle programlama bilgisi edinmeyi gerektiren bölümlerde bu durum önemli bir sorundur. Öğrencilerin başarılarını ve programlama dersine olan ilgilerini arttırmak amacıyla farklı eğitim yaklaşımlarıyla ortamlar düzenlenebilir, farklı yöntem ve teknikler kullanılabilir (Norvell & Bruce-Lockhart, 2004). Bu çalışmada programlama öğrenme sürecinde yaşanan zorlukları kolaylaştırmak amacıyla farklı bir öğrenme

ortamının kullanılması ve yaşanan zorluklara çözüm önerileri sunulması, çalışmanın önemini ortaya koymaktadır.

Araştırmanın çalışma grubunu, meslek yüksekokulu bilgisayar teknolojileri bölümü öğrencileri oluşturmaktadır. Meslek yüksekokulu bilgisayar teknolojileri bölümü, %92'lik bir oranla yazılım sektörünün yetişmiş insan gücünü karşılamaktadır (Yükseköğretim Bilgi Yönetim Sistemi [YBYS], 2018). Ancak günümüzde meslek yüksekokullarında öğrencileri derste aktif hale getirecek stratejiler kullanılmaması, öğrencilerin bu derslerde edinmeleri gereken becerileri kazanamamalarına neden olabilmektedir (Binici & Necdet, 2004; Şahin & Fındık, 2008; Ünal & Çakır, 2016). Öğrencilerin öğrendiklerini yeni ve farklı durumlarda kullanamamaları nedeniyle sektörün ihtiyaçları karşılanamamakta ve mezun öğrencilerin istihdamı her geçen yıl azalmaktadır. Ayrıca Türkiye'de üniversite düzeyindeki öğrencilerle programlama eğitiminde görsel programların kullanıldığı çalışmalar sınırlı, meslek yüksekokulu öğrencileriyle yapılan çalışmalar yok denecek kadar azdır. Yazılım sektörünün yetişmiş insan gücünün büyük bir bölümünü karşılaması beklenen meslek yüksekokulu öğrencileriyle programlama öğretiminde üç boyutlu görsel programlama araçlarından biri olan Alice programı ile çalışılması çalışmayı özgün hale getirmektedir.

Çalışma kapsamında Alice programının öğrencilerin programlama dersindeki akademik başarısı, motivasyonu, problem çözme becerisi algısı ve programlamaya hazır bulunuşluk düzeyine etkisi incelenmektedir. Programlama dili öğreniminde öğrencilerin özellikle temel kavramları öğrenme düzeyinde kaldıkları ve bu yüzden programlamada ilerleyemedikleri (Berge vd., 2003; Bucci vd., 2001), sözdizimi hatalarıyla zaman kaybettikleri ve bu yüzden programlamaya bütünsel olarak bakamadıkları (Berge vd., 2003; Bucci vd., 2001), soyutlamayı yeterince yapamadıkları ve bu yüzden somut problemleri yeterince çözemedikleri (Gomes & Mendes, 2007) görülebilmektedir. Bu bağlamda etkisi incelenen değişkenlerin programlama öğreniminde önemi ortaya çıkmaktadır.

Çalışmada Alice programı ile çalışılarak öğrencilerin kod hatalarıyla uğraşmak yerine temel kavramları daha iyi anlamaları sağlanıp, soyutlaştırma becerilerini geliştirmek hedeflenmiştir. Soyutlamayı ileri düzeyde yapabilen ve temel kavramları daha iyi anlayıp yenilikçi programların ortaya çıkmasına odaklanan öğrencilerin yetişmesinin amaçlanması

çalışmayı önemli kılmaktadır. Ayrıca problem çözme becerisi algısı, programlamanın alt becerisi olmasının yanı sıra 21. yüzyıl becerileri arasında yer almaktadır. Çalışmanın bu yönü de araştırmanın önemini ortaya çıkarmaktadır.

Varsayımlar

1. Uygulamaya katılan deney ve karşılaştırma grubundaki öğrencilerin kontrol dışı değişkenlerden, araştırmanın sonucunu etkileyecek şekilde etkilenmediği varsayılmıştır.

Tanımlar

a. Meslek Yüksekokulu: Sektörlere ara eleman yetiştirme misyonunu üstlenmiş, uygulamaya dönük eğitim kurumlarıdır (Yükseköğretim Kurulu Başkanlığı [YÖK], 2007).

b. Programlama Becerisi: Her biri farklı bilişsel süreçleri ve üst düzey düşünme becerilerini kapsayan alt görevlerden oluşan, günümüz teknoloji okuryazarlığının önemli bir parçasını oluşturan beceridir (Ambrosio, Costa, Almeida, Franco & Macedo, 2011; Lau & Yuen, 2011).

c. Alice Programı: Nesne yönelimli programlama dili ile ilgili temel bilgilerin elde edilebileceği şekilde tasarlanmış, serbestçe kullanılabilir bir 3D programlama ortamıdır (Alice, 2018).

d. Geleneksel Öğrenme Ortamı: Bilgiyi aktarmaya ağırlık veren eğitim anlayışı, ders kitaplarına aşırı bağımlılık, öğretmenin mutlak egemenliği, öğrencileri araştırmaya yöneltmeyip yalnızca dinleyen/izleyen konumda tutarak onların zihinsel açıdan edilgenleştirilmelerine sebep olan düzenlemeler, kişisel görüşleri açıklamaya izin vermeyen sınıf iklimi, sunulan bilgileri anlamaya ve farklı yorumlar yapmaya olanak tanımayan öğretim yöntemlerinin kullanıldığı öğrenme ortamıdır (Arslan, 2009; Deryakulu, 2000).

BÖLÜM II

KAVRAMSAL ÇERÇEVE İLE İLGİLİ ARAŞTIRMALAR

Bu bölümde araştırmanın kavramsal çerçevesini oluşturan programlamaya; programlama dili öğretimine; programlama başarısını etkileyen faktörler ile problem çözme becerisi algısı, motivasyon ve programlamaya hazır bulunuşluk düzeyine; programlama eğitiminde kullanılan görsel programlama ortamlarına; konuyla ilgili Türkiye'de ve dünyada yapılan çalışmalara yer verilmektedir.

Programlama

Programlama, bilgisayara ne yapacağını söyleyen talimatlar yazma sanatıdır ve programlama dilleri aracılığıyla yazılan talimatlara program adı verilmektedir (Resnick, Maloney, vd., 2009). Programlama ile gerçek durumlar, bilgisayar ortamında modellenir (Çölkesen, 2002). Programlama süreci aşamalı bir yapıdadır ve her bir aşama ciddi bir çalışma gerektirmektedir. Programlamada öncelikle problemin çözümüne yönelik adımlar tasarlanmalı ve akış diyagramı oluşturulmalıdır (Çölkesen, 2002; Eryılmaz, 2003). Sonrasında kullanılacak programlama dili belirlenmelidir. Programların geliştirilebilmesi için farklı yapılarda programlama dilleri bulunmaktadır.

İlk bilgisayarlarda bilgisayarla kullanıcı arasındaki iletişim, makine dili aracılığıyla sağlanmaktaydı. Makine dilinde kodlar herhangi bir derleyici kullanılmadan yazılmaktaydı. Kodların bilgisayar işlemcisine aracı program olmadan gitmesi, hız açısından avantaj sağlasa da programları denetimli bir şekilde çalıştırmak zor olabilmekteydi. Bu durum kullanımı kolay yapıya sahip farklı programlama dillerinin

ortaya çıkmasını sağlamıştır. Özellikle programın yazıldığı bilgisayardan bağımsız farklı bilgisayarlarda çalışabilen, İngilizce komutlar içeren ve karmaşık yapıdaki makine dillerinden daha anlaşılır olan Fortran ve Algol dillerinin geliştirilmesinden sonra Cobol, Basic, Pascal, C gibi programlama dilleri oluşturulmuştur. Ancak oluşturulan bu programlama dillerinde geliştirilen programların yüksek oranda kod içermesi, programlarda oluşan hataların kontrolünü zorlaştırabilmektedir. Günümüzde daha az kod kullanmaya olanak sağlayan, daha kullanışlı, gelişmiş ve nesneye yönelik programlamayı destekleyen C++, Java, C# gibi üst düzey programlama dilleri geliştirilmiştir.

Programlama Dili Öğretimi

Programlama dilleri, içinde barındırdığı komut ve işlevlerle ilgili kurallardan oluşan bir yapıdır (Ford Jr, 2015). Kullanıcı istediği programı yazmak için belirlediği programlama dili ile kodlar yazmakta, bilgisayarlar ise bu kodları yorumlamaktadır (Kingsley-Hughes & Kingsley-Hughes, 2005).

Programlama dillerinin öğretimi ve öğrenimi üst düzey becerileri gerektirmektedir (Gültekin, 2006). Özellikle bilişsel beceriler ön plana çıkmaktadır (Helminen & Malmi, 2010; Storey, 2005). Çünkü bir program yazarken programcı, zihinsel bir model oluşturarak programın işlem adımlarını planlar ve oluşturduğu bu zihinsel model ile farklı çözüm yolları üretir, programı geliştirir (Helminen & Malmi, 2010).

Programlama sürecini bilişsel bir süreç olarak gören Shneiderman ve Mayer (1979) programcı davranışlarını beş temel adımda aşağıdaki şekilde ele almıştır:

1. Kompozisyon : Bir programın yazılması.
2. Anlama : Verilen bir problemin anlaşılması.
3. Hata ayıklama : Verilen bir programdaki hataların bulunması.
4. Değiştirme : Yeni bir göreve uygun hale getirmek için verilen programın değiştirilmesi.
5. Öğrenme : Yeni programlama bilgi ve becerilerinin kazınılması.

Bu adımlar sonucunda programcılar kendi bilişsel yapılarını oluşturmakta, bilgilerini bilişsel süreçlerine eklemekte ve gerektiğinde kullanmaktadır (Shneiderman & Mayer, 1979).

Bayman ve Mayer (1988) programlama öğrenimde programcıların yazılımsal, kavramsal ve stratejik bilgilere sahip olması gerektiğini belirtmişlerdir. Yazılımsal bilgi ile programlama dilinin kurallarının, kavramsal bilgi ile programlama dilinin kavramlarının, stratejik bilgi ile problem çözme becerilerinin kazanıldığını vurgulamaktadırlar.

Kesici ve Kocabaş (2001) bir programın hazırlanmasındaki beş temel adımı şu şekilde ele almaktadır :

1. Problemin tanımlanması : Problemin ne derece anlaşıldığını gösterir.
2. Çözüm yolu belirlenmesi : Çözüm adımlarının yer aldığı algoritma hazırlanır.
3. Kodlama : Seçilen programlama dili kodları yazılır.
4. Kod derlenmesi : Bilgisayarın anlayacağı şekle kodlar çevrilir.
5. Hata kontrol : Programdaki yazım veya mantık hataları belirlenerek giderilir.

Gundurao vd. (2010) programlamayı üç temel adım altında toplamış ve her adımı kendi içerisinde basamaklara ayırmıştır. Adımlar ve basamakları şu şekildedir:

1. Problem analizi : Problemin tüm yönleriyle tanımlanması.
2. Problem çözümü : Algoritma ve akış diyagramının hazırlanması.
3. Problemin bilgisayar çözümü : Programlama dili kodlarının yazılması, test edilmesi, hataların belirlenip giderilmesi.

Farklı yıllarda tanımlanan programlama adımları incelendiğinde görülmektedir ki program oluşturma süreci aşamalı bir yapıdır. Programcıların sadece programa ait bir algoritma tasarlaması yetmemekte, çözüme dair işlem adımlarını belirledikleri bir programlama diline de uyarlamaları gerekmektedir (Cooper, Dann & Pausch, 2003). Programcı kullandığı programlama dilinin kavramsal düzenlemesini yaparak programı yazmalı, test etmeli ve hata varsa bulup düzeltmelidir (Helminen & Malmi, 2010). Programlama süreci programcıların, problem çözme ve tasarlama algılarını desteklemektedir.

Günümüzde programlama iyi bir kariyer vaat eden bir alan olarak görülmekte ve bireylerden hazır programları kullanmaları yerine kendi programlarını geliştirmeleri beklenmektedir (Resnick, Flanagan vd., 2009; Scott, 2005). Ancak programlama karmaşık bir süreçtir ve öğrenilmesi kolay değildir (Resnick, Flanagan, vd., 2009). İlgili alanyazın incelendiğinde programlama dersleri zor olarak kabul edildiği ve öğrencilerin başarı oranlarının yüksek olduğu çalışmaların sınırlı sayıda olduğu görülmektedir (Baştemur Kaya & Çakır, 2015; Robins vd., 2003; Solmaz, 2014). Yıllar geçtikçe programlama dillerinin öğrenilmesinin kolaylaştırılması amacıyla dillerin yapısının değiştirilmesine rağmen programlama öğretiminde pek fazla değişiklikler olmamıştır. Genellikle öğretmenler, yeni kontrol ve veri yapılarını öğrencilere anlatır, bir kaç örnek gösterir ve onlardan karşılaştıkları problemleri çözmelerini bekler. Ancak öğrencilerin bu konuda belirttikleri sıkıntı, öğretmen anlatırken anladıkları fakat kendi problemlerine çözüm üretemedikleri yönündedir (Garner, 2003). Araştırmalar göstermektedir ki öğrencilerin çoğunun programlamayı kendi kendilerine öğrenmeleri zorlu bir süreç olabilmektedir ve öğrencilerin başarılı olabilmeleri için öğretmenler çeşitli stratejiler geliştirmeli, eğitsel araçlar kullanmalıdır (Norvell & Bruce-Lockhart, 2004).

Programlama Başarısını Etkileyen Faktörler

Programlama öğrenmede pek çok öğrenci zorluk yaşayabilmektedir (Gomes & Mendes, 2007; Resnick, Flanagan, vd., 2009). Programlama öğrenilmesinde ve öğretilmesinde dikkat edilmesi ve çaba gösterilmesi gereken faktörler bulunmaktadır. Programlama öğreniminde öğretim yöntemleri, öğrencilerin çalışma yöntemleri, öğrencilerin yetenekleri ve tutumları, programlamanın yapısı ile psikolojik etkenler programlamayı etkileyen temel faktörler arasındadır (Gomes & Mendes, 2007).

Öğretim Yöntemleri

Programlama öğretiminde geleneksel öğretim yöntemleri, birçok öğrencinin öğretimden beklentileri doğrultusunda yeterli görülmemektedir (Gomes & Mendes, 2007; Norvell & Bruce-Lockhart, 2004). Daha kişiselleştirilmiş öğrenci gözetimi, problem çözme sırasında anında geri bildirim ve daha az anlaşılmış yönlerin detaylı açıklanması öğrencilere

yardımcı olabilmektedir. Ancak sınıf ortamında belirli bir ders saatinde belirli bir müfredatın yetiştirilmesi gerekmektedir ve bu tür bir desteğin verilmesine imkân bulunamayabilir. Öğretmen öğrencinin gelişim evrelerini takip etmeli, şüphelerini netleştirmeli, farklı problem durumları ve aktivitelerle öğrenmeyi zenginleştirmelidir (Driscoll, 1994; Gomes & Mendes, 2007). Öğrencinin motivasyonunu azaltan durumları önlemeli, süreci iyi yönetmeli ve öğrencileri motive etmelidir (Gomes & Mendes, 2007).

Geleneksel eğitimde, tüm öğrenciler aynı ritimde ve öğretmenin öğrenme stiline dayanan stratejiler doğrultusunda öğrenmektedirler (Deryakulu, 2000; Gomes & Mendes, 2007). Öğretmenlerin kullandıkları stratejiler, ilgili konunun anlatımını ya da bütün öğrencilerin öğrenme stillerini desteklemeyebilir. Kimi öğrenciler öğrenmeyi sadece bir süreç olarak görürken, kimi öğrenciler daha dinamik öğrenme ortamlarını tercih edebilir (Gomes & Mendes, 2007). Öğrencilerin kendilerine en uygun öğrenme yaklaşımını benimsemelerine yardımcı olunmalıdır (Haşlaman & Aşkar, 2007). Öğrenme çevresinde her öğrencinin tercih ettiği öğrenme stillerini destekleyebilecek aktiviteler kullanılmalıdır (Pintrich & De Groot, 1990). Aktiviteler, farklı sunum formatları ve etkileşimli stratejilerle öğrenci özelliklerine uyarlanmalıdır (Gomes & Mendes, 2007). Aktivitelerin sunumunda, öğrencilerin algıda seçicilik süreçleri dikkate alınmalı ve içeriğin temel ayırt edici özellikleri vurgulanmalıdır (Driscoll, 1994).

Programlama öğretiminde genellikle statik materyallerle, dinamik kavramların eğitimi verilmekte ve bu statik materyaller; yansıtılan sunumları, sözlü açıklamaları, metinleri içermektedir (Gomes & Mendes, 2007). Statik materyallerin yanı sıra esnek teknoloji tabanlı sistemlere geçiş yapılsa da (çevrimiçi ders, sınav vb.) genellikle ders kitapları temel kaynak olarak sunulmaktadır (Garner, 2003). Bazı öğrenciler için bu tür statik materyallerle programlamanın dinamik yapısını anlamak sorundur. Programlama öğretimi yapılacak öğrenme çevresinin, programlama kavramlarını temsil eden ve farklı öğrenme stillerine hitap eden dinamik materyaller içermesi, programlama öğrenimi açısından yararlı olmaktadır (Gomes & Mendes, 2007).

Programlama eğitiminde öğretmenler, genellikle programlama dili kullanarak problem çözmeyi teşvik etmek yerine programlama dilinin ve sözdiziminin ayrıntılarını öğretmeye daha fazla konsantre olmaktadır (Gomes & Mendes, 2007). Programlama öğretilirken programlama ile problem arasında ilişkinin nasıl kurulacağı da verilmelidir (Kak, 2009).

Gomes ve Mendes (2007)'e göre farklı türdeki problemler aracılığıyla ilerleme aşamalı ve ilerici olmalıdır. İlk aşamadaki problemler basit, öğrencilerin ilgisini çeken, öğrencileri programlamaya teşvik eden özellikte olmalıdır. Yavaş yavaş sorunlar, tipik programlama problemlerine yönelik daha spesifik alanlara ilerlemelidir. Öğrencilere sunulan her yeni problem daha ayrıntılı ve farklı prosedürleri içeren çözümleri içermelidir. Burada amaç, öğrencinin ilerleyişini teşvik edici ve ilgi çekici bir ders akışı oluşturmaktır (Gomes & Mendes, 2007).

Öğrencilerin Çalışma Yöntemleri

Programlama sürecinde öğrencilerin öğrenmelerini yönlendirme becerilerini kazanmaları önemli yer tutmaktadır (Haşlanaman & Aşkar, 2007). Öğrenme süreci yönlendirme becerilerinin kazandırılması aşamasında; öğrenme ortamlarında bir etkinlik gerçekleştirildikten sonra, ek sorular veya faaliyetler sağlanarak öğrencilerin çözümlerini nasıl geliştirebilecekleri hakkında düşünmeleri sağlanmalıdır (Gomes & Mendes, 2007). Böylece öğrencilerin kendilerine uygun olan öğrenme stratejisini seçmelerine, hedefleri doğrultusunda seçtikleri stratejiyi uygulamalarına ve bireysel gelişimlerini izleyip değerlendirme yapmalarına imkân tanınmış olunur (Haşlanaman & Aşkar, 2007). Öğrencilere verilen görevlerde, gösterdikleri çabanın kontrolünü ve yönetimini sağlayan stratejilerin benimsenmesi önemlidir (Pintrich & De Groot, 1990). Özellikle sunulan materyalleri öğrenme ve hatırlamayı içeren yineleme, ayrıntılandırma, düzenleme gibi bilişsel stratejilerin (Zimmerman & Martinez-Pons, 1986, 1988) kullanımı sağlanmalıdır.

Öğrencilerin Yetenekleri ve Tutumları

Öğrencilerin çoğu problemi tamamen anlamadan çözmeye çalışmakta ve dikkatlice düşünmeden önce bir cevap yazma eğiliminde olmaktadır (Gomes & Mendes, 2007). Öğrenciler, bir problemi öğretmenin rehberliğinde anlayabilecek olsa da, kendi başlarına çözmenin zor olduğunu düşünebilirler (Garner, 2003). Öğrencilerin problemi anlama yeteneklerinin geliştirilebilmesi amacıyla öğrencilere ne sorulduğunu anladıklarının doğrulanması için; sorunun ne olduğu, problem çıktılarının ne olabileceği veya problem

durumunda bazı değişiklikler yapılarak yeni çıktının ne olması gerektiği soruları yöneltilebilir (Gomes & Mendes, 2007).

Birçok öğrenci geçmiş problemlerle doğru analojiler oluşturamamakta, geçmiş bilgileri yeni sorunlara aktaramamakta ve genellikle aynı kurallarla benzer yüzeysel özelliklere sahip problemleri gruplandırma eğiliminde olmaktadırlar (Gomes & Mendes, 2007). Öğrencilerde bilginin ilişkilendirilmesi yeteneğinin geliştirilmesi için, yeni bilgiler öğretilirken öğrencilere eski bilgileri hatırlatılmalı ve ilişki kurulmalıdır (Merrill, 2002). Önceki problemlerle benzerlik gösteren problemler sorulmalı, problemlerin zorluk seviyeleri arttırılmalı, yeni problemler önceki problemlerdeki bazı alt görevleri çözmek için gereken benzer kavramları içermeli, öğrencinin mevcut problem durumunu çözemediği tespit edilirse; problemi bölmek ve bazı parçaları daha önce çözülen problemlerle ilişkilendirmek gibi ipuçları verilmelidir (Gomes & Mendes, 2007).

Öğrencilerin çözüme kendilerinin ulaşmalarını sağlamak amacıyla bilginin kalıcılığını arttırmak için yeterli geri bildirim verme ve öğrencilerle etkileşim içinde olma yolu izlenmeli, problemi çözmeleri için öğrenciler motive ve teşvik edilmelidir (Gomes & Mendes, 2007). Gerekli yerlerde öğrencilere ipuçları verilmeli ve sonuca ulaşmaları desteklenmelidir (Driscoll, 1994). Öğrencileri motive ve teşvik etmek amacıyla; uzun bir süre problem çözümüne ulaşılmazsa çözüm verilip sorular yöneltilebilir, öğrencilerden gelen çözümlerdeki hataların bulunması istenebilir, eksik çözümlerin nasıl tamamlanacağı sorulabilir (Gomes & Mendes, 2007).

Programlamanın Yapısı

Program yazmak bilişsel bir süreçtir (Apiola & Tedre, 2012; Zapusek & Rugelj, 2013). İyi bir programcı olmak için programlama dilinin sözdizimi bilinmelidir. Ancak sözdizimini bilmenin ötesinde bir dizi becerinin de kazanılması gerekmektedir (Gomes & Mendes, 2007). Programlama için bireyler soyutlama, problem çözme, genelleme, transfer, algoritmik düşünme ve eleştirel düşünme gibi becerilere sahip olmalıdır (Gomes & Mendes, 2007; Gundurao vd., 2010).

Soyutlama becerisini etkileyen matematiksel ve mantıksal düşünme becerileri programlama için önemli bir yer tutmaktadır ve programlama öğrenme ortamları; örtülü ya

da açık bir şekilde, tipik programlama problemlerini çözmeye yardımcı olabilecek farklı zorluk seviyelerinde olan matematiksel kavramları içerecek şekilde düzenlenmelidir (Gomes & Mendes, 2007). Programlama problemlerinin çözümünü kolaylaştıran, algoritma tasarlama ile geliştirmeye yardımcı olan kaynaklardan ve görsel araçlardan yararlanılmalıdır (Garner, 2003).

Programlama dillerinin temel yapıları birbirine benzerdir. Temel kavramların bilinmesi farklı dillerin öğrenilmesi için kolaylık sağlamaktadır. Ayrıca programlamada, programlamaya özgü yeteneklerin geliştirilmesinin yanı sıra geçmiş programlama bilgi ve yeteneklerinin geliştirilmesi, eski bilgilerle yeni bilgilerin inşa edilmesi çok önemlidir (Gomes & Mendes, 2007). Öğrenciler, programlamaya hazır bulunuşluk düzeyleri arttıkça programlamanın yapısını daha iyi bir şekilde anlayabilir.

Programlama basamakları incelendiğinde, programlamanın temelinde bir problemi çözmek olduğu görülmektedir. Programlamada bir problemi çözmek amacıyla, programlama dilleri kullanılarak programlar geliştirilir. Ancak öğrencilerin problem çözme becerilerindeki düşüklük, özellikle başlangıç seviyesinde olan programlama derslerini olumsuz yönde etkileyen faktörler arasında görülmektedir (Gomes & Mendes, 2007).

Bu bağlamda, programlama becerisine etki eden önemli faktörlerden olan programlamaya hazır bulunuşluk düzeyi ve problem çözme becerisi algısı aşağıda ayrıntılı olarak açıklanmaktadır.

Programlamaya Hazır Bulunuşluk Düzeyi

Her programlama dilinin kendine özel kodlama sistemi bulunsa da, hepsinde kullanılan ortak temel kavramlar mevcuttur (Sureau, 2012). Bu yüzden programcılarda aranan temel özellik, genel programlama bilgisidir (Eryılmaz, 2003). Programcılık eğitimi verilirken yalnızca kullanılan programlama diline ait söz diziminin öğretilmesi, öğrencilerin o dilin yapısını öğrenmesini sağlar. Ancak başarılı bir programcı olabilmeleri için temel kavramları çok iyi öğrenmeleri gerekir (Kak, 2009). Temel kavramlar öğrencilerin programlamaya hazır bulunuşluk düzeyi ile ilişkilidir.

Byrne ve Lyons (2001) çalışmalarında akademik performans, bilgisayar deneyimi, cinsiyet, öğrenme stilleri ve geçmiş programlama deneyimi arasındaki ilişkiyi, anket ve akademik kayıtlar aracılığıyla incelemişlerdir. Araştırma sonucunda geçmiş programlama deneyimine sahip olan öğrencilerin programlama performanslarının daha yüksek olduğu belirlenmiştir.

Holden ve Weeden (2003) çalışmalarında programlama alanındaki deneyimlerin programlama dersi sırasındaki etkisini inceleyen bir anket çalışması yapmışlardır. Araştırma sonucunda öğrencilerin programlama deneyimlerinin ilk derslerde programlama dili öğrenimi performansını olumlu yönde etkilediği belirlenmiştir.

Goold ve Rimmer (2000) çalışmalarında cinsiyet, akademik beceri, öğrenme stili, problem çözme becerisi ve motivasyon arasındaki ilişkiyi anket aracılığıyla incelemişlerdir. Bilgisayar deneyimleri ve lise not ortalamaları bilgileri de toplanmıştır. İki dönem süren araştırmada öğrencilerin bilgisayar deneyimlerinin akademik becerilerini ve programlamaya olan ilgilerini olumlu etkilediği görülmüştür. Ayrıca problem çözme becerisi algısı ve motivasyonu yüksek olan öğrencilerin daha başarılı olduğu, öğrenme stilinin bir etkisinin olmadığı belirtilmiştir.

Yapılan araştırmalar öğrencilerin programlamaya hazır bulunuşluk düzeyi arttıkça programla başarılarının arttığını göstermektedir (Byrne & Lyons, 2001; Goold & Rimmer, 2000; Holden & Weeden, 2003). Ancak mevcut öğretim sisteminde öğrencilerin önceden programlama dersi almalarına rağmen yeni bir programlama dili öğrenirken zorlanabildikleri görülmektedir. Bu durum programlamaya hazır bulunuşluk düzeylerinin düşük olduğunu ve temel kavramları yeterince öğrenemediklerini göstermektedir. Öğrencilere programlamanın temel yapısını öğreten yeni araçlara ve öğretim yöntemlerine ihtiyaç duyulmaktadır.

Problem Çözme Becerisi Algısı

Bir problemin geliştirilmesi sürecinde sadece programlama dilinin kurallarını bilmek yetmemekte, başarılı bir programlama için problem çözme becerileri de gerekmektedir. Bilgisayar programlarının nasıl yazıldığının öğrenilmesi için öğrencilerin problem çözme

becerileri geliştirilmelidir (Gundurao vd., 2010). Problem çözme doğuştan itibaren öğrenilen ve okul yıllarında geliştirilen bir beceridir (Miller & Nunn, 2001).

Problem çözme becerisine sahip bireyler yenilikçi, aktif, sorumluluk duygusuna sahip, alternatif fikir üretebilen, dikkatli, yöntemlere bağlı, yaratıcı ve olaylara eleştirel yaklaşabilen özellikleriyle ön plana çıkmaktadır (Koberg, 1981). Ayrıca problem çözme becerisine dayalı olarak tasarlanan eğitim ortamları öğrencilerin motivasyonlarını arttırır, öğrendiklerini gerçek hayatla ilişkilendirerek daha kalıcı davranış sergilemelerini sağlar, işbirlikli çalışmayı destekler (Çakmak & Tertemiz, 2002; Stepien & Gallagher, 1993).

Programlama bir problem çözme sürecidir. Problem çözme becerisinin düşük olması özellikle programlama sürecinin başlangıç düzeyinde başarısızlığa sebep olan faktörlerden biridir (Gomes & Mendes, 2007). Programlama sürecinde programcının yapması gereken adımlar incelendiğinde problemin anlaşılması ve çözüm yollarının belirlenmesi adımlarının, program geliştirmenin temelini oluşturduğu görülmektedir (Gundurao vd., 2010; Kesici & Kocabaş, 2001; Shneiderman & Mayer, 1979). Bu yüzden problem çözme becerisi, programlama başarısını etkileyen en önemli ve etkisi en çok incelenen faktörler arasında yer almaktadır.

Henderson (1986), Special Interest Group on Computer Science Education (SIGCSE)'nin tutanaklarını belirttiği çalışmasında analitik düşünme ve problem çözme becerisinin programlama alanında önemli bir yer tuttuğunu ve özellikle öğrencilerin bu becerilerde zorlandığını belirtmiştir.

Pillay ve Jugoo (2005) çalışmalarında programlama eğitiminde problem çözme becerisi, cinsiyet, öğrenme stili, ilk programlama dili öğrenimi ve geçmiş bilgisayar deneyimleri arasındaki ilişkiyi belirlemek amacıyla, Java programlama dilinin öğretildiği bir derste, öğrenenlere bir anket uygulamışlardır. Çalışma sonucunda problem çözme becerisi ve ilk programlama dili öğreniminin, programlama başarısını en çok etkileyen faktörler olduğu tespit edilmiştir.

Hung (2008) çalışmasında problem çözme beceri ile programlama başarısı arasındaki ilişkiyi incelemiştir. Deneysel olarak yaptığı çalışmada programlama başarısı için başarı testleri, problem çözme becerisi için görüşme ve çeşitli etkinlikler uygulamıştır. Çalışma sonucunda problem çözme becerisinin programlama becerisini arttırdığı belirtilmiştir.

Ismail vd. (2010) çalışmalarında programlama eğitimindeki ihtiyaçları ve var olan durumu belirlemek amacıyla programlama eğitimi veren öğretim elemanlarıyla görüşmeler yapmışlardır. Araştırma sonucunda programlamanın zor ve karmaşık bir süreç olduğu programlama becerisi ve analitik düşünmenin bu süreçte önemli bir rol oynadığı ancak öğrencilerin bu becerilere tam anlamıysa sahip olmadıkları bu yüzden programlama sürecinde zorluk yaşandığı belirtilmiştir.

Kwon, Yoon ve Lee (2011) çalışmalarında hibrit programlama ortamının öğrencilerin problem çözme ve programlama algısı üzerindeki etkisini belirlemeyi amaçlamışlardır. Veriler anket yardımıyla toplanmıştır. Deneysel olarak yaptıkları çalışma sonucunda ortamın, problem çözme becerisini ve programlama algısını arttırdığı görülmüştür.

Linn ve Dalbey (1985) bilişsel süreçlerle tasarlan bir programlama dersinin öğrencilerin öğretim şekillerini, bilgisayara erişimlerini, programlama sürecinde öğrencileri etkileyen becerileri belirlemek üzere deneysel bir çalışma yapmışlardır. Araştırmada problem çözme becerisinin programlama dili öğretiminde gerekli becerileri arttırdığı belirtilmiştir. Öğrenciler programlama sürecinde geribildirimin kendilerine bilgisayar tarafından verilmesinin, problem çözme becerilerini geliştirdiğini ifade etmişlerdir.

Yapılan çalışmalarda problem çözme ve analitik düşünme becerisinin programlama başarısını doğrudan etkilediği görülmüştür (Goold & Rimmer, 2000; Henderson, 1986; Hung, 2008; Ismail vd., 2010; Kwon vd., 2011; Linn & Dalbey, 1985; Pillay & Jugoo, 2005).

Psikolojik Etkenler

Programlama öğretiminde öğrenci motivasyonunun önemli bir yeri bulunmaktadır. Ancak öğrenciden öğrenciye geçen olumsuz çağrışımlar, öğrencilerin programlamayı incelemek için yeterli motivasyona sahip olmalarının önüne geçmektedir. Ayrıca programlama zor ve uğraş gerektiren bir süreç olarak görülebildiği için öğrenciler bu süreç içinde sıkılabilir ve ilgileri dağılabilir (Ersoy, Madran & Gülbahar, 2011). Programlama öğretiminde çeşitli problem çözme aktiviteleri içeren görsel programların ve çoklu ortamların kullanılması öğrencilerin ilgilerini ve motivasyonlarını arttırabilmektedir (Zhang vd., 2014). Böylece öğretim daha eğlenceli bir yapıya dönüşebilir ve öğrenci temel problemlerin çözümüne

odaklanabilir. Programlama öğretiminde mümkün olduğunca gerçek hayat problemleri kullanılarak, öğrencilere programlamanın insan yaşamını kolaylaştırmak amacıyla yararlı bir araç olduğunu göstermek gerekmektedir(Gomes & Mendes, 2007).

Bu bağlamda, programlama becerisine etki eden önemli faktörlerden biri olan motivasyon algısı aşağıda ayrıntılı olarak açıklanmaktadır.

Motivasyon Algısı

Eğitim sırasında bazı öğrencilerin ilgili derse, konuya ya da karşılaşılan probleme çözüm üretmekte istekli oldukları gözlenirken, bazı öğrencilerin derslerde isteksiz oldukları, karşılaştıkları problemlerle mücadele etmek yerine daha çok kaçmayı tercih ettikleri gözlemlenmektedir (Akbaba, 2006). Motivasyon, kişinin istekli hale gelerek, harekete geçmesini sağlar (Deci vd., 1999).

Alanyazında birçok çalışmada programlama eğitimi zor bir süreç olarak görülmektedir ve bu zorluk nedeniyle öğrencilerin motivasyonu her zaman üst seviyede olamamaktadır. Karşılarına çıkan zor problemler öğrencilerin motivasyon algılarını azaltarak kendilerini yetersiz hissetmelerine neden olabilmektedir. Bu da öğrencilerin başarılarını etkilemektedir.

Matthíasdóttir (2006) çalışmasında öğretmenlerin ders konularını ses kaydı yaptığı, online test uyguladığı ve birebir öğrenme yöntemlerinin kullanıldığı bir programlama dersi tasarlayarak ilk defa programla dersi alan katılımcılara ders vermiş ve anket yoluyla verileri toplamıştır. İlgili araştırma bir proje kapsamında yapılmıştır. Programlama dersinin zor olduğuna dikkat çekilerek öğrenenlerin ders kapsamında öğretmenin anlattıklarını dinlemek yerine öğretmenin çektiği ders kayıtlarını sisteme eklemesi ile öğrenenlerin gerekli durumlarda evden ya da okuldan kayıtları dinlemesi, okulda birbirleriyle iletişim kurarak verilen görevleri yerine getirmesi sayesinde ilgilerinin daha çok programlamaya yöneldiği belirtilmiştir. Çalışma sonunda yapılan anket ile öğrenenlerin motivasyonlarının artmasıyla programlama dili öğrenim isteklerinin de arttığı belirlenmiştir.

Jiau vd. (2009) çalışmalarında programlama dersinde öğrencilerin sadece verilen kodlarla ders işlemek zorunda kaldığını ve bu durumun öğrencilerin motivasyonlarını olumsuz

etkilediğini belirtmişler, programlama dersine yönelik eğitsel bir oyun geliştirerek öğrencilerin derste aktif olmasını amaçlamışlardır. Görüşme ve anket yoluyla toplanan verilerde öğrencilerin motivasyonları arttığında programlama eğitiminde daha başarılı olduğu belirlenmiştir.

Yapılan araştırmalar göstermektedir ki motivasyon programlama başarısını arttıran bir faktördür (Goold & Rimmer, 2000; Jiau vd., 2009; Matthíasdóttir, 2006). Mevcut öğretim sisteminde öğrenciler öğrendiklerini farklı problemlerde kullanamamaktan yakınmaktadır. Bu durum öğrencilerin ilgilerinin dağılmasına ve programlama öğrenimine karşı isteksiz olmasına neden olabilir. Bu yüzden programlama öğretiminde öğrencilerin ilgisini çekecek yeni araçlar ve öğrencilerin derse aktif katılımını sağlayacak yeni eğitim ortamları tasarlanmalıdır.

Programlama başarısını etkileyen faktörler bir bütün olarak incelendiğinde, programlama öğretiminde; öğrenme ortamında öğrencilerin ilgilerini ve motivasyonlarını kaybetmemelerini sağlayan, öğretim sırasında öğrencileri programlamaya teşvik eden, gerekli durumlarda geribildirim ve ipuçları verilen, eski bilgilerle yeni durumların ilişkilendirilmesi sağlanan, farklı düzeyde problemlerin sunumu ve bu problemlerin çözümü için gerekli yardımın verilmesine olanak tanıyan öğretim modellerinin kullanılması; görsel programlar gibi dinamik araçlarla programlama öğrenimini kolaylaştıracak desteğin verilmesi gerektiği görülmektedir. Bu bağlamda, çalışmada Gagné'nin dokuz aşamalı öğretim modeli uygulanmış ve öğretim ortamı bahsedilen problemleri gidermeye yönelik düzenlenmiştir. Görsel programlama ortamlarından Alice programı kullanılarak dinamik araçlarla; öğrencilerin temel kavramları daha iyi anlamaları, derse olan ilgilerinin arttırılması ve öğrencileri programlama yapmaya teşvik etmek amaçlanmıştır. Gagné'nin dokuz aşamalı öğretim modeli kullanılarak düzenlenen öğrenme ortamı ve Alice programı ile ilgili ayrıntılı bilgi Bölüm III'te, Uygulama Süreci başlığı altında detaylı bir şekilde verilmiştir

Programlama Eğitimi ile İlgili Yapılan Çalışmalar

Programlama ile ilgili alanyazın incelendiğinde karmaşık bir süreç olarak nitelendirildiği ve öğrenci başarısının yüksek olduğu çalışmaların sınırlı sayıda olduğu görülmektedir

(Baştemur Kaya & Çakır, 2015; Robins vd., 2003; Solmaz, 2014). Yıllar geçtikçe programlama dillerinin yapısında kod ve sözdizimi yükünün azaltılmasına rağmen, programlama eğitiminde, öğrencilerin daha iyi anlamasını ve başarılarının artmasını sağlayacak değişiklikler pek fazla olmamıştır. Öğretmen merkezli sınıf ortamlarında öğrenciler konuyu anladıklarını fakat yeni problemlere uygulayamadıklarını belirtmektedir (Garner, 2003). Özellikle son yıllarda programlamanın önem kazanmasının yanı sıra, programlamanın zor bir süreç olarak görülebilmesi ve programlamada başarı oranlarının düşük olabilmesinden dolayı eğitim alanında bir çok çalışma programlama alanında yapılmaktadır. Aşağıda programlama eğitiminde yapılan çalışmalara yer verilmektedir.

Feddon ve Charness (1999) çalışmalarında başlangıç, orta ve ileri düzey programlama bilgisi ile; programlamanın basamakları olan anlama, düzenleme, hata ayıklama ve derleme arasındaki ilişkiyi belirlemeyi amaçlamışlardır. Programlama düzeyi farklı olan 35 kişi ile çalışmışlar ve dört basamağı da kapsayan bir test uygulamışlardır. Araştırma sonucunda orta düzeyde programlama bilgisi ile düzenleme ve derleme basamakları arasında anlamlı ilişki olduğu görülmüştür.

Wilson ve Shrock (2001) Midwestern Üniversitesi'nde bilgisayara giriş dersi kapsamında 105 öğrenci ile yaptıkları çalışmalarında matematik bilgisi, başarı veya başarısızlık algısı, programlama deneyimi, cinsiyet, derse etki, dersteki rahatlık ve cesaretlendirme değişkenlerinin akademik başarıyı öngörebilme düzeyini incelemişlerdir. Araştırma sonucunda dersteki rahatlık, matematik altyapısı ve başarı veya başarısızlık algısı; akademik başarıyı öngören değişkenler olarak belirlenmiştir.

Vihtonen, Alaoutinen ve Kaarna (2001) çalışmalarında C programlama dili öğretimi için Viope adında bir ortam tasarlamışlardır. Deneysel olarak yaptıkları çalışmalarında deney grubuna tasarlanan Viope ile, kontrol grubuna geleneksel yöntemle eğitim verilmiş, çalışma sonunda anketle ve başarı sınavıyla veriler toplanmıştır. Çalışma sonunda deney grubunun başarılarının ve derse katılma isteklerinin daha yüksek olduğu belirlenmiştir.

Grant (2003) çalışmasında programlama dersinin öğrencilerin problem çözme becerisine ve bilişsel öğrenme stiline etkisini araştırmıştır. 41 öğrenci ile iki farklı programlama dersi için yapılan araştırmada anket ile veriler toplanmış, programlama dersinin problem çözme becerisi ve bilişsel öğrenme stiline etki etmediği görülmüştür.

Unuakhalu (2008) çalışmasında problem çözme ve programlama yeterliliğini belirlemek amacıyla Visual Basic programlama dili öğretimi için bir eğitim ortamı planlamıştır. Deneysel olarak tasarlanan çalışmada, deney grubu öğrencilerinin çeşitli görevleri yapması planlanmış, kontrol grubu öğrencilerine geleneksel yöntemlerle eğitim verilmiştir. Deneysel süreç başında ve sonunda ölçekler uygulanmıştır. Çalışma sonucunda problem çözme becerisinde gruplar arasında anlamlı bir farklılık gözlenmezken programlama yeterliliği deney grubu lehine anlamlı bir farklılık gözlenmiştir.

Hernandez vd. (2010) çalışmalarında programlama becerilerini geliştirmek amacıyla bir bilgisayar oyunu tasarlamışlardır. Çeşitli diller kullanılarak geliştirilen oyun öğrencilere uygulanmıştır. Bir önceki yıl bu dersi alan öğrenciler ile, oyun kullanarak dersi alan öğrenciler programlama becerileri açısından karşılaştırılmıştır. Oyun kullanan öğrencilerin programlama mantığını daha iyi geliştirdikleri ve mantıksal hataları görsel olarak gördükleri için daha iyi bulabildikleri belirlenmiştir.

Coşar (2013) çalışmasında probleme dayalı olarak bir programlama dersi tasarlamış; tasarladığı dersin akademik başarı, eleştirel düşünme ve bilgisayara yönelik tutum üzerine etkisini araştırmıştır. Deneysel olarak tasarlanan çalışmada 58 katılımcı ile çalışılmış, çalışma sonucunda tasarlanan ortamın akademik başarı, eleştirel düşünme ve bilgisayara yönelik tutumun deney grubu lehine anlamlı çıktığı görülmüştür.

Çalışmalar incelendiğinde programlama dersine yönelik öğrencilerin farklı eğitsel çıktılarının gelişimine yardımcı olunması ve soyut kavramların daha iyi anlaşılabilmesi amacıyla ya bilgisayar destekli ortamlar geliştirilerek derslerin verildiği ya da programlama dersinin farklı etkinlikler çerçevesinde tasarlandığı görülmektedir.

Programlama Eğitiminde Kullanılan Görsel Programlama Ortamları

Alanyazında programlama öğreniminde yaşanılabilen zorlukların giderilmesi amacıyla görsel programlama ortamlarının kullanıldığı görülmektedir. Bu bölümde Greenfoot, Stratch, StarLogo TNG, BlueJ ve Alice programları ile ilgili bilgi verilmektedir.

Scratch Programı

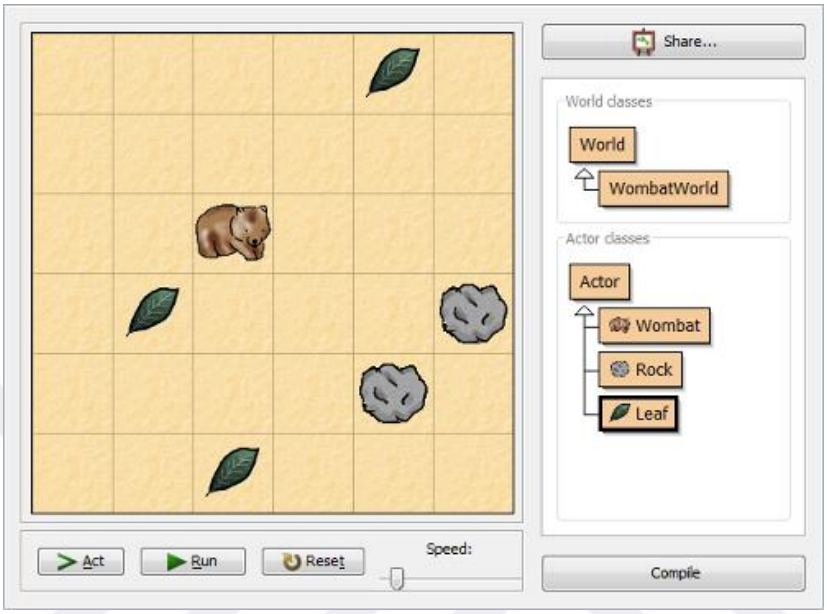
Scratch programı, Massachusetts Institute of Technology (MIT) Medya Lab'ında yer alan Lifelong Kindergarten grubunun bir projesi olarak, prosedürel programlama dillerinin yapısının anlaşılması amacıyla geliştirilmiştir (Scratch, 2018). Kullanıcıların etkileşimli projeler oluşturmasına olanak tanıyan görsel bir programlama ortamıdır (Maloney, Resnick, Rusk, Silverman & Eastmond, 2010).

Şekil 1. Scratch programının kullanıcı arayüzü. "*The scratch programming language and environment*", Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E., 2010, *ACM Transactions on Computing Education (TOCE), 10*(4), Article 16. http://doi.acm.org/10.1145/1868358.1868363 kaynağından alınmıştır.

Scratch programının kullanıcı arayüzüne ait görüntü Şekil 1'de verilmektedir. Tek pencereli kullanıcı arayüzü vardır ve görsel blok dilidir (Maloney vd., 2010). Sürükle-bırak özelliği ile kodlar eklenebilir, grafik ve seslerle sonuçlar görülebilir (Resnick, Maloney, vd., 2009).

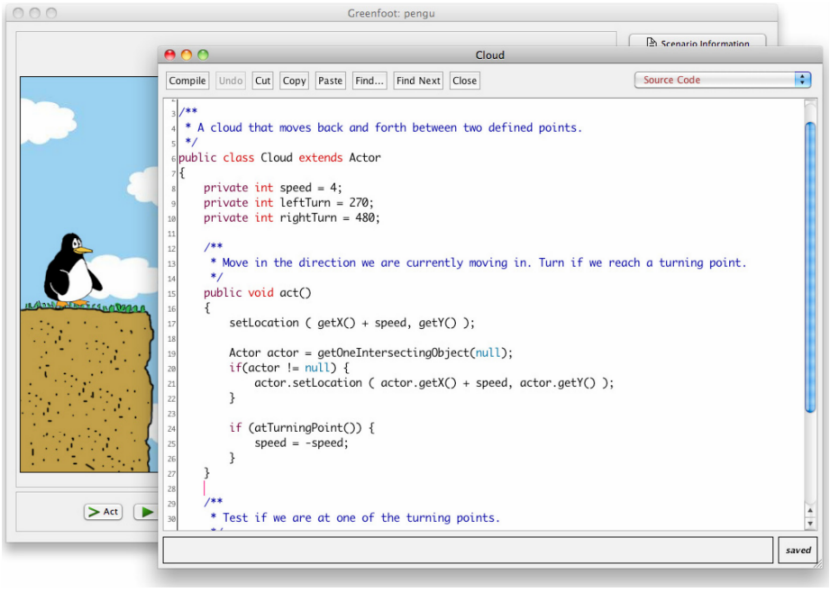
Greenfoot Programı

Greenfoot programı bir üniversite projesi olarak geliştirilmiştir ve temelinde Java programlama dili kullanılmıştır (Greenfoot, 2018). Greenfoot programı nesne yönelimli programlama kavramlarının görselleştirmesine olanak tanımaktadır (Kölling, 2010).



Şekil 2. Greenfoot programının kullanıcı arayüzü. Greenfoot, 2018, https://www.greenfoot.org/overview sayfasından erişilmiştir.

Greenfoot programında, etkileşimli bir şekilde nesnelerin davranışlarına yön verilebilmektedir (Greenfoot, 2018). Greenfoot programının ana görünümü Şekil 2'de verilmektedir. Pencerenin ana kısmı, programın gerçekleştirildiği Greenfoot dünyasını göstermektedir. Bu dünya, kullanıcı tanımlı boyuttadır ve uygulanacak senaryonun aktörlerini tutmaktadır. Şekil 2'de aynı zamanda, bu senaryoda kullanılan sınıfları ve kalıtım ilişkilerini görselleştiren bir sınıf diyagramı da görülmektedir. Burada görünen iki üst sınıf (Dünya ve Aktör) Greenfoot programının bir parçasıdır ve her zaman mevcuttur (Kölling, 2010).

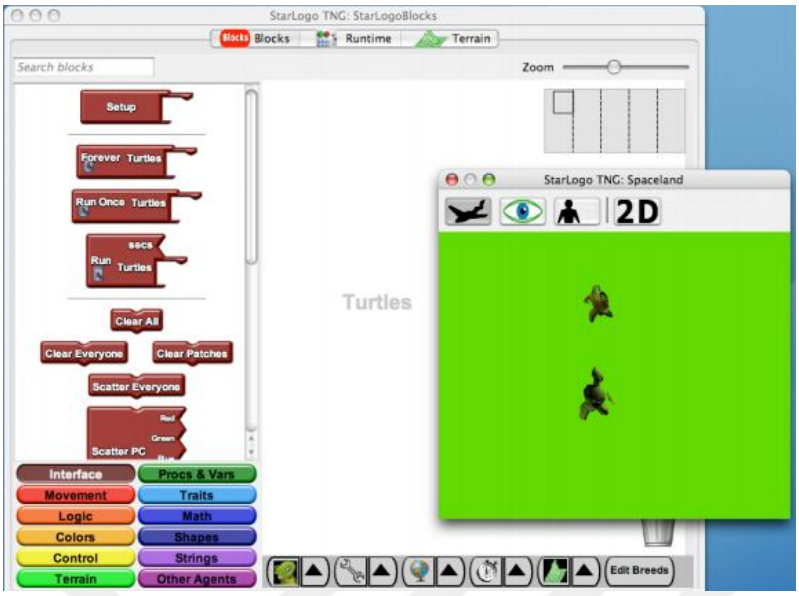


Şekil 3. Greenfoot programı kod editörünün görünümü. "*The Greenfoot programming environment*", Kölling, M., 2010, *ACM Transactions on Computing Education (TOCE), 10*(4), Article 14. http://doi.acm.org/10.1145/1868358.1868361 kaynağından alınmıştır.

Sınıf diyagramında bir sınıfa çift tıklatıldığında Şekil 3'te verildiği gibi sınıfın kaynak kodunu gösteren bir metin düzenleyicisi açılmaktadır. Bir sınıf için kodlar yazıldıktan sonra sınıf derlenebilir ve aktörler etkileşimli olarak oluşturulabilir. (Kölling, 2010).

StarLogo TNG Programı

StarLogo TNG programı kullanıcıların programlama mantığını anlamaları amacıyla geliştirilmiştir (Klopfer & Begel, 2003). Geleneksel programlama dillerine göre genellikle öğrenilmesi ve kullanılması daha basit olan grafik tabanlı uygulamalar yapmaya imkân sağlamaktadır (McCaffrey, 2006). Logo programlama dili yapısına uygun olarak geliştirilmiştir (Klopfer, Scheintaub, Huang, Wendel & Roque, 2009).

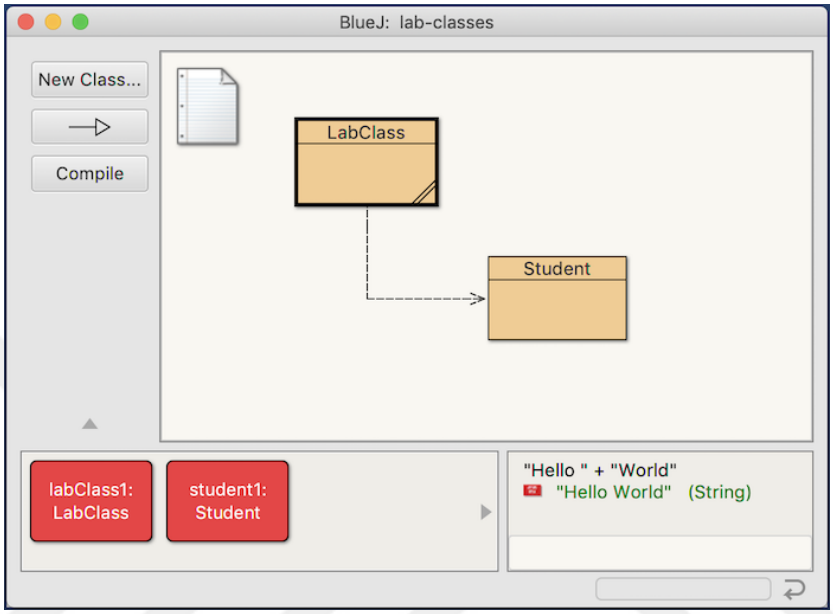


Şekil 4. StarLogo TNG programının kullanıcı arayüzü. "*StarLogo TNG: The convergence of graphical programming and text processing*", McCaffrey, C., 2016, https://dspace.mit.edu/bitstream/handle/1721.1/36904/80770344-MIT.pdf?sequence=2 kaynağından alınmıştır.

StarLogo TNG programında kullanıcıların kendi uygulamalarını oluşturabildiği bir grafik arayüzü bulunmaktadır. StarLogo TNG programının kullanıcı arayüzüne ait görüntü Şekil 4'te verilmektedir. Kullanıcılar uygulamalarını oluştururken, uygulama arayüzü ile ilgili talimatları veya değerleri belirleyebilirler. StarLogo TNG programında girdi, çıktı, yardım vb. işlemleri gerçekleştirebilmek amacıyla çeşitli kullanıcı arabirimi öğeleri vardır (Klopfer & Begel, 2003). Kullanıcı uygulamasını oluştururken butonlar ve kaydırma çubuğundan yararlanmaktadır. Kullanıcının uygulamasına eklediği komutların çalıştırılmasının ve yürütülmesinin kontrol edilebilmesi amacıyla butonlar kullanılmaktadır. Kaydırma çubuğu aracılığıyla, kullanıcının uygulamadaki global değişkenleri kontrol etmesine olanak sağlanır. Programlamanın temel kavramlarının öğrenilmesini amaçlayan StarLogo TNG programı ile kodlar bloklar halinde uygulamaya dahil edilmekte ve sonuçlar üç boyutlu olarak görülebilmektedir (McCaffrey, 2006).

BlueJ Programı

BlueJ programı, nesne yönelimli programlama dili kavramlarının öğrenilmesini kolaylaştırmak amacıyla Michael Kölling ve John Rosenberg tarafından geliştirilen görsel programlama ortamıdır (Kölling & Rosenberg, 1996).



Şekil 5. BlueJ programının kullanıcı arayüzü. BlueJ, 2018, https://www.bluej.org sayfasından erişilmiştir.

BlueJ programı kullanıcının nesne yönelimli programlama dilinin temel kavramlarını öğrenmesine kadar geçen süre içerisinde, herhangi bir kod yazmadan mevcut sınıflar ile etkileşimde bulunmasına imkân sağlamaktadır (Hagan & Markham, 2000). BlueJ programının kullanıcı arayüzüne ait görüntü Şekil 5'te verilmektedir. BlueJ programı kullanıcılara bir uygulamadaki sınıfların ve nesnelerin grafiksel bir resmini verir, kullanıcıların doğrudan verilen sınıf ve nesne resmiyle etkileşime girmesini sağlar, metot ve sınıfların test edilmesini kolaylaştırır (Van Haaster & Hagan, 2004). Fare tıklamaları ile kod oluşturulmasına olanak sağlar (Davis, 2017).

Alice Programı

Alice programı, bir üniversite projesi olarak geliştirilmiştir. Programlama kavramlarını tanıtmayı amaçlayan üç boyutlu, etkileşimli, grafiksel bir programdır. Programcılar Alice programını kullanarak hem genel programlama yapılarını öğrenmekte hem de kendi sanal dünyalarını tasarlamaktadır (Cooper vd., 2000a). Alice programı kullanıcıların animasyon geliştirirken bir problemi çözdürmeyi amaçlaması sebebiyle kullanıcıların problem çözme ve algoritmik düşünme becerilerini destekleyen bir araç olarak değerlendirilmektedir (Cooper vd., 2000b).

Ortamda kullanıcılar basit komutlar ile eklediği nesnelerin özelliklerini ve hareketlerini kontrol edebilmektedir (Cooper vd., 2000b; Wang vd., 2009). Kodlarla görsellerin eşgüdümlü olarak nasıl çalıştığını görebilmekte, kod-görsel arasında ilişki kurabilmektedir (Cooper vd., 2000b). Böylece kullanıcılar için programlama, soyut kavramları ve programlama yapısını hayal etmekten çıkar ve kodların görsel olarak ne işe yaradığını görmelerini sağlar (Berge vd., 2003). Ayrıca kullanıcılar temel kavramların nasıl kullanıldığını, ne işe yaradığını görür ve aktif olarak programlama sürecine katılarak farklı programlama dillerine ait yapıların (fonksiyonlar, döngüler, olaylar vb.) işlevlerini rahatlıkla anlayabilirler (Cooper vd., 2000b; Wang vd., 2009). Bu şekilde Alice programı farklı programlama dilleri için kullanıcıların programlamaya hazır bulunuşluk düzeyleri ile programlamaya karşı ilgi ve katılımlarını arttırmaktadır (Cooper vd., 2003; Howard vd., 2006).

Programda kod yazmak yerine sürekle-bırak yöntemi ile kodlar hazır olarak eklenir. Böylece söz dizimi hatalarının önüne geçilerek, kullanıcının programlama yapısına ve kavramlarına yönelmesi sağlanır. Çalışılan her bir nesneye özgü kodlar eklenebilir ve nesnelerin uzuv, boy, konum vb. özellikleri rahatlıkla değiştirilerek nesneler hareket ettirilebilir (Cooper vd., 2000b, 2003). Programın kod dilinde Alice ve Java seçenekleri bulunmaktadır. Java seçeneği seçilerek değişkenler ve bazı kodlar Java programlama diline uygun şekilde eklenip görülebilir. Ayıca kodların Java programlama dilindeki dizilim şekli de programda çalışırken eşgüdümlü olarak görülebilir. Programda hazırlanan animasyonlar NetBeans programlama dili derleyicisi ile de açılabilmektedir. Ancak NetBeans programında kod eklenmesi veya çıkarımı yapılmaz. Sadece kodların derleyicide görünümü ve derleme işlemi yapılabilir (Alice, 2018).

Görsel Programlama Ortamlarının Karşılaştırılması

Scratch, Greenfoot, StarLogo TNG, BlueJ ve Alice programları nesne yönelimli programlama diline uyumluluk, animasyon özelliği, iki ve üç boyutlu olma durumu, hikaye oluşturma, kullanılabilirlik, sürükle-bırak özelliği, yardım desteği, çalışmasında yaşanabilecek sorunlar, güncelleştirme, programlama dili derleyicilerinde çalıştırılabilirlik, eklenen kodların programlama diline uygun bir şekilde görünümü, örnek ve uygulamalara ulaşılabilirlik ile ilgili temel özellikler üzerinden karşılaştırılması bu bölümde verilmiştir.

Scratch programı, kendi davranışlarına sahip nesnelerden oluşmaktadır ve prosedürel programlama dili yapısıyla uyumludur (Maloney vd., 2010). Greenfoot programı, nesne yönelimli Java programlama dili alt yapısına sahiptir (Greenfoot, 2018). StarLogo TNG programı Logo programlama dili yani nesne yönelimli programlama dili yapısına uygundur (Klopfer vd., 2009). BlueJ programı nesne yönelimli programlama dili yapısının öğretilmesini amaçlamaktadır (Hagan & Markham, 2000). Alice 3 programı da temelinde nesne yönelimli programlama dillerini barındırmaktadır (Alice, 2018).

Alice programı, animasyon geçişlerini yoğun bir şekilde kullanmaktadır ve animasyonda yapılan her değişiklik (pozisyon hareketleri gibi) görülebilir (Cooper vd., 2000a). Buna karşın; Scratch programında sadece bir komut görsel olarak görülebilir, diğer animasyonlar döngüler kullanılarak oluşturulmak zorundadır (Utting vd., 2010). StarLogo TNG programında her kodun etkisi görsel bir şekilde görülebilir.

Üç boyut, bazı proje türleri (hikâyeler, oyunlar, sanal dünyalar vb.) için çok iyi bir özelliktir. Scratch ve Greenfoot programları, oyun ve hikâyeleri desteklemektedir fakat bu programlarda daha büyük sanal dünyalar oluşturmak için büyük çaba harcanması gerekmektedir (Utting vd., 2010). BlueJ programı üç boyut uygulama yapmaya olanak tanımaz. StarLogo TNG programında üç boyutlu uygulamalar yapılabilir. Alice programı da üç boyutlu görsel programlar arasındadır (Cooper vd., 2000a). Hikâyeler üç boyutta daha gerçekçidir (Utting vd., 2010).

Greenfoot programının çerçevesi daha reaktif bir sisteme yöneliktir (Utting vd., 2010). Başka bir ifadeyle, kullanıcı girişlerine tepki veren veya birbirleri ile etkileşimde olan nesnelere yöneliktir (Kölling, 2010). BlueJ programının yapısı da Greenfoot programıyla benzerlik göstermektedir. Hikaye oluşturma durumu, Scratch ve Alice programında daha

kolaydır ve iki programda öyküler ve öykü anlatımı açısından birbirine yakındır (Utting vd., 2010). StarLogo TNG programıyla da kolay uygulamalar yapılabilir.

Scratch, StarLogo TNG ve Alice programları kodların eklenebilirliği ve program arayüzünün kullanışlığı açısından kolay kullanılabilir programlardır. BlueJ programı kod yazma açısından geleneksel programlama diline göre kullanım kolaylığı sağlamaktadır.

Scratch, StarLogo TNG ve Alice programlarında sürükle-bırak özelliği ile kodlar eklenebilmektedir. Ancak Greenfoot ve BlueJ programlarında bu özellik bulunmamaktadır.

Tüm programlarda yardım menüsü desteği bulunmaktadır. Tüm programların güncel sürümleri sitelerinde mevcuttur. Bu durum bir versiyonda oluşan hataların, sonraki versiyonlarında düzeltilebilmesine imkân sağlamaktadır.

Diğer programlama ortamlarından farklı olarak Alice programı uygulamaları NetBeans derleyicisi programında açılabilmektedir. Bu şekilde Alice uygulamalarına eklenen kodlar ve işlevleri derleyicide kolay bir şekilde görülebilmektedir.

Greenfoot, BlueJ ve Alice programlarında uygulamalara karşılık gelen Java programlama dili sözdizimine uygun bir şekilde kodlar görülebilmektedir. Scratch, Greenfoot ve Alice programları alanyazında en çok kullanılan görsel programlama ortamlarındandır. Bu açıdan ilgili programların örnek uygulamalarına ulaşılabilirlik daha kolay olabilmektedir.

Görsel programlama ortamlarının; nesne yönelimli programlama diline uyumluluk, animasyon özelliği, iki ve üç boyutlu olma durumu, hikaye oluşturma, kullanılabilirlik, sürükle-bırak özelliği, yardım desteği, çalışmasında yaşanabilecek sorunlar, güncelleştirme, programlama dili derleyicilerinde çalıştırılabilirlik, eklenen kodların programlama diline uygun bir şekilde görünümü, örnek ve uygulamalara ulaşılabilirlik ile ilgili temel özellikler üzerinden karşılaştırılma bilgisi Tablo 1'de verilmektedir.

Tablo 1

Görsel Programlama Ortamlarının Karşılaştırılması

Özellik	**Program**				
	Scratch	**Greenfoot**	**StarLogo TNG**	**BlueJ**	**Alice**
Nesne yönelimli programlama diline uyumluluk	-	✓	✓	✓	✓
Her kodun işlevinin animasyon halinde görülebilmesi	-	-	✓	-	✓
Üç boyutlu olma	-	-	✓	-	✓
Kolay hikaye oluşturulabilme	✓	-	✓	-	✓
Kullanım kolaylığı	✓	-	✓	✓	✓
Sürükle-bırak özelliği ile kod eklenmesi	✓	-	✓	-	✓
Yardım menü desteği	✓	✓	✓	✓	✓
Hatalardan arındırılmış olma	✓	✓	✓	✓	✓
Güncelleştirme desteği	✓	✓	✓	✓	✓
Programlama dili derleyicilerinde çalıştırılabilirlik	-	-	-	-	✓
Programa eklenen kodların programlama dilinin söz dizimine uygun bir şekilde görünebilirliği	-	✓	-	✓	✓
Örnek uygulamalara ulaşılabilirlik	✓	✓	-	-	✓

Scratch, Greenfoot, StarLogo TNG, BlueJ ve Alice programları programlama öğretiminde kullanılan önemli görsel ortamlardır. Nesne yönelimli programlama öğretimi için uygunluk durumu, kaliteli animasyon oluşturabilme özelliği, üç boyutlu olması, farklı öykülerin kolayca tasarlanabilmesi, eklenen her kodun etkisinin animasyon olarak görülebilmesi,

sürükle-bırak özelliğinin olmasından dolayı programlama mantığının ve temel kavramların anlaşılmasına daha iyi odaklanılabilmesi, derleyici programlarda eklenen kodların görünebilmesi, örnek uygulamalara kolay bir şekilde ulaşılabilmesi gibi özellikler birlikte değerlendirildiğinde Alice programının daha avantajlı olduğu görülmektedir. Bu sebeple bu tez kapsamında öğrenme ortamı olarak Alice programı kullanılmış, Alice programının öğrencilerin programlama başarısına, problem çözme becerisi algısına, motivasyonuna ve programlamaya hazır bulunuşluk düzeylerine etkisi araştırılmıştır.

Alice Programı ile İlgili Yapılan Çalışmalar

Alice programının çeşitli eğitsel çıktılara olan etkisinin ve Alice programı ile ilgili görüşlerin belirlendiği çalışmalar incelenmiştir. Alice programının akademik başarıya olan etkisinin incelendiği çalışmalar aşağıda verilmektedir.

Moskal, Lurie ve Cooper (2004), Alice programının programlama öğretimindeki etkisini belirlemek amacıyla deneysel bir çalışma yapmışlardır. Araştırmanın başlangıcında Bilgisayar Bilimleri öğrencilerinden oluşan çalışma grubuna test uygulanarak programlamadaki durumları belirlenmiştir. Çalışmanın deney grubunda uygulanan test sonucuna göre programlama alanında zayıf olan öğrenciler Alice programıyla çalışmışlardır. İki kontrol grubunun birinde programlama alanında zayıf olan öğrenciler, diğer kontrol grubunda programlama alanında iyi olan öğrenciler yer almış ve geleneksel programlama ortamında çalışmışlardır. Süreç başında ve sonunda veri toplama aracı olarak ölçek ile başarı testi kullanılmış ayrıca ders kayıt listelerinden yararlanılmıştır. Alice programı ile çalışan programlama alanında zayıf olan öğrenciler, Alice programı ile çalışmayan programlama alanında iyi olan öğrenciler kadar başarılı olduğu belirtilmiştir. Öntest-sontest sonuçları incelendiğinde gruplar arasında anlamlı bir farklılık bulunmamıştır.

Sykes (2007) programlama eğitiminde Alice programı kullanımının etkisini belirlemek amacıyla karma yöntemin kullanıldığı bir çalışma yapmıştır. Lisans öğrencileri ile Bilgisayar Bilimleri 1 dersi kapsamında yapılan çalışmada iki karşılaştırma grubu bir deney grubu olmak üzere üç grupla çalışma yürütülmüştür. Birinci karşılaştırma grubu 2004 yaz dönemi, ikinci karşılaştırma grubu 2005 yaz dönemi, deney grubu 2005 güz

dönemi Bilgisayar Bilimleri 1 dersini alan öğrencilerden oluşmaktadır. Karşılaştırma gruplarında geleneksel yöntemle, deney grubunda Alice programı ile eğitim verilmiştir. Araştırmanın nitel boyutunda her hafta öğrencilerle bireysel görüşmeler yapılmış, öğrencilere anket uygulanmış ve gözlem yapılmıştır. Nicel boyutunda öntest-sontest olarak başarı testi uygulanmıştır. Araştırma sonucunda deney grubu lehine anlamlı bir faklılık olduğu tespit edilmiştir.

Johnsgard ve McDonald (2008) programlama öğreniminde Alice programı kullanımının akademik başarıya olan etkisini incelemişlerdir. Deneysel olarak yürüttükleri çalışmada Programlama 1 dersi kapsamında C++ programlama dili öğretilmiştir. Araştırma sonuçlarında Alice programının kullanımıyla başarı oranının %46,4'den %70,3'e çıktığı ve akademik başarının deney grubu lehine anlamlı olarak arttığı tespit edilmiştir.

Wang vd. (2009) yaptıkları deneysel çalışmada, lise öğrencileriyle çalışmış ve Alice programıyla C++ programlama dili öğrenimini karşılaştırmışlardır. Deney grubunda Alice programı ile, kontrol grubunda C++ programlama dili ile 8 hafta eğitim verilmiş ve eğitim sonunda temel akademik yeterlilik testi, başarı testi ve anket öğrencilere uygulanmıştır. Araştırma sonucunda Alice programı ile çalışan öğrencilerin temel programlama kavramlarını daha iyi anladıkları belirlenmiştir. İki grupta yer alan öğrencilerin bilgisayar programlama öğrenme motivasyonları veya öğrenme deneyimleri arasında anlamlı bir farklılık bulunmamıştır.

Al-Linjawi ve Al-Nuaim (2010) Alice programı ile kalıtım yapısını öğretmek amacıyla deneysel bir çalışma yapmışlardır. Çalışma grubunu 45 lisans seviyesindeki öğrenci oluşturmaktadır. Çalışma kapsamında hazırlanan akademik başarı testi, öntest ve sontest olarak kullanılmıştır. Araştırma sonucunda Alice programının kalıtım yapısını öğrenmede anlamlı yönde etki ettiği belirlenmiştir.

Schultz (2011) bilgisayar bilişim sistemleri öğrencilerine programlama mantığını öğretmek amacıyla Alice programıyla çalışarak bir araştırma yapmıştır. Alice programı ile öğrenim gören ve görmeyen iki grupta yer alan öğrencilerin akış diyagramları ve pseudocode cevapları karşılaştırılmıştır. Araştırma sonucunda iki grup cevapları arasında anlamlı bir farklılık çıkmamıştır.

Biju (2013) programlama öğreniminde Alice programı kullanımının etkisini belirlemek amacıyla nicel bir çalışma yapmıştır. İki yarıyıl süren çalışmada C++ dersi birinci yarıyıl geleneksel yöntemle, ikinci yarıyıl Alice programı ile öğrencilere anlatılmıştır. Her yarıyılda 15 öğrenci ile çalışılmış ve iki gruba da aynı testler uygulanmıştır. Araştırma sonucunda Alice programı ile çalışan öğrencilerin başarı ortalamalarının %15-20 arasında artış gösterdiği belirlenmiştir.

Price (2013) çalışmasında görsel olarak algoritma oluşturmayı sağlayan Raptor programını ve Alice programını kullanarak programların, akademik başarı ve kalıcılık üzerine etkilerini araştırmıştır. Deneysel olarak yürütülen çalışmada deney grubunda dönemin ilk yarısında Raptor, ikinci yarısında Alice programı kullanılmıştır. Ders ADDIE modeli ile tasarlanmıştır. Kontrol grubunda geleneksel yöntemle ders işlenmiştir. Araştırma verileri başarı sınavı ve quizlerle toplanmıştır. Araştırma sonucunda Alice programının nesne ve sınıf kullanımında öğrenci performansını anlamlı yönde etkilediği belirtilmiştir.

Solmaz (2014), Alice programının PHP programlama dili öğretiminde öğrencilerin eleştirel düşünme eğilimleri, problem çözme becerileri, üstbilişsel farkındalık düzeyleri ve ders başarıları üzerindeki etkisini ve öğrenci görüşlerini belirlemek amacıyla bir çalışma yapmıştır. Araştırmada nicel veriler anket ve başarı testi yardımıyla, nitel veriler ise görüşme formu ile toplanmıştır. Araştırma sonucunda Alice programının eleştirel düşünme eğilimine, problem çözme becerisine, üstbilişsel farkındalığına ve ders başarısına anlamlı yönde etki etmediği belirlenmiştir. Bu durumun nedenlerinden biri olarak yedi hafta süren uygulama sürecinin üst düzey becerilerde beklenen değişimi meydana getirmemiş olabileceği belirtilmiştir. Bunun yanında kullanılan öğretim yönteminin ve nesne yönelimli olmayan bir programlama dili ile çalışılmasının da araştırma sonuçlarını etkileyebileceği araştırmacı tarafından ifade edilmiştir.

Zhang, Liu, de Pablos ve She (2014), Alice programının akademik başarı, motivasyon ve ilgi düzeyine olan etkisini belirlemek amacıya deneysel bir çalışma yapmışlardır. Deney grubu Alice programı ile kontrol grubu geleneksel yöntemler ile Java programlama dili öğrenimi almıştır. Araştırma sonucunda Alice programı ile öğrenim gören grubun akademik başarı, motivasyon ve ilgi düzeylerinin geleneksel yöntem ile öğrenim gören gruptan daha yüksek olduğu belirlenmiştir.

Alice programının akademik başarıya olan etkisinin değerlendirildiği çalışmalar incelendiğinde genellikle deneysel yöntemlerin kullanıldığı görülmektedir. Çalışma grubu olarak anabilim dalı bilgisayar olan üniversite öğrencileriyle (Moskal vd., 2004; Schultz, 2011; Solmaz, 2014), anabilim dalı bilgisayar olmayan üniversite öğrencileriyle (Al-Linjawi & Al-Nuaim, 2010; Sykes, 2007; Zhang vd., 2014), lise öğrencileriyle (Wang vd., 2009) çalışıldığı görülmektedir. Alice programının akademik başarıya olumlu yönde etkisinin olduğunu tespit eden çalışmaların (Al-Linjawi & Al-Nuaim, 2010; Biju, 2013; Johnsgard & McDonald, 2008; Price, 2013; Sykes, 2007; Zhang vd., 2014) olmasının yanı sıra, akademik başarıya olumlu yönde etki etmediğini belirleyen çalışmalar (Moskal vd., 2004; Schultz, 2011; Solmaz, 2014; Wang vd., 2009) da bulunmaktadır.

Alice programının problem çözme becerisi algısına olan etkisine bakıldığı çalışmalar incelenmiştir. Çalışmalara ait detaylı bilgi aşağıda verilmektedir.

Solmaz (2014), PHP programlama dili öğretiminde Alice programını kullanarak öğrencilerin farklı eğitsel çıktıları üzerindeki etkisini ve öğrenci görüşlerini belirlemek amacıyla deneysel bir çalışma yapmıştır. Öğrencilerin problem çözme becerisi algısını belirlemek amacıyla araştırma sonunda katılımcılara bir anket uygulamıştır. Araştırma sonucunda Alice programının problem çözme becerisine anlamlı yönde etki etmediği belirlenmiştir.

Alice programı ile programlama eğitiminin problem çözme becerisi algısına olan etkisinin incelendiği çalışmaların sınırlı sayıda olduğu görülmektedir. Yapılan çalışmanın anabilim dalı bilgisayar olan üniversite öğrencileri ile gerçekleştirildiği ve Alice programının problem çözme becerisi algısına olumlu yönde etki etmediğinin belirlendiği görülmektedir.

Alice programının motivasyona olan etkisine bakıldığı çalışmalar incelenmiştir. Çalışmalara ait detaylı bilgi aşağıda verilmektedir.

Wang vd. (2009) lise öğrencileriyle çalışmış ve C++ programlama dili öğreniminde Alice programı kullanımının etkisini araştırmışlardır. Deneysel olarak yapılan çalışmada motivasyonu belirlemek amacıyla öğrencilere anket uygulanmıştır. Araştırma sonucunda iki grupta yer alan öğrencilerin programlama öğrenme motivasyonları veya öğrenme deneyimleri arasında anlamlı bir farklılık bulunmamıştır.

Zhang vd. (2014), Alice programı ile programlama öğretiminin öğrencilerin akademik başarı, motivasyon ve ilgi düzeyine olan etkisini belirlemek amacıya deneysel bir çalışma yapmışlardır. Araştırma sonucunda deney grubunun motivasyon ve ilgi düzeyinin kontrol grubundan daha yüksek olduğu belirlenmiştir.

Alice programının motivasyona olan etkisinin değerlendirildiği çalışmalar incelendiğinde sınırlı sayıda olduğu görülmektedir. Çalışma grubu olarak lise (Wang vd., 2009) ve anabilim dalı bilgisayar olmayan üniversite öğrencileri (Zhang vd., 2014) ile çalışıldığı görülmektedir. Alice programının motivasyona olumlu yönde etki ettiğini tespit eden çalışmalar (Zhang vd., 2014) ve motivasyonu olumlu yönde etki etmediğini belirten çalışmalar (Wang vd., 2009) bulunmaktadır.

Alice programının programlama mantığını ve temel programlama kavramlarını anlamada, ön bilgi düzeyini arttırmada etkisinin belirlendiği çalışmalar incelenmiştir. Çalışmalara ait detaylı bilgi aşağıda verilmektedir.

Cooper vd. (2000a) ve Cooper vd. (2000b) yaptıkları çalışmalarda Alice programını kullanarak algoritmik düşünmeyi, programlamayı ve problem çözmenin temel kavramlarını geliştirmek amacıyla bir ders tasarlamışlardır. Derste öğrenciler bazen küçük gruplar halinde bazen de bireysel olarak çalışmışlardır. Veriler gözlem yoluyla elde edilmiştir. Araştırma sonucunda öğrencilerin Alice programını rahatça kullanabildikleri, nesne ve metotlarla kolay bir şekilde çalışabildikleri, programı eğlenceli buldukları, derse katılımlarının arttığı ve verilen süreden daha fazla programda vakit geçirdikleri belirtilmiştir. Ayrıca grup çalışmalarında, bireysel çalışmalara göre daha iyi projeler geliştirildiği gözlenmiştir. Öğrencilerin programlama becerileri konusunda özgüven duygusu geliştirdikleri belirtilmiştir. Çalışmada genel olarak, Alice programında değişkenlere gerek olmadığı ifade edilmiştir. Öğrencilerin, hareketli dünyadaki nesneleri doğrudan etkileyen komutlar verdiğini bu yüzden, öğrencinin x değişkeninin beş değerine sahip olup olmadığını düşünmesine gerek kalmadığını ifade etmişlerdir. Öğrencilerin, programlama dili yapılarına ve bunların değişkenler yerine nasıl işlediğine, nasıl değiştirildiğine odaklanabildiğini belirtmişlerdir.

Cooper vd. (2003) çalışmalarında Alice programı kullanarak programlama deneyimi olmayan ve programlama deneyimi çok az olan üniversite öğrencilerine problem çözme

basamaklarını ve programlamanın temel adımlarını öğretmek amacıyla bir ders tasarlamışlardır. Derste öğrencilerden verilen görevleri bireysel olarak yerine getirmeleri, ders dışı verilen görevlerin ise grup halinde yapılması istenmiştir. Nitel olarak toplanan veriler sonucunda öğrenciler Alice ortamında çalışmanın rahat olduğunu, sınıf dışında verilen görevleri yerine getirdiklerini, programlama konusunda öz güvenlerinin arttığını ifade etmişlerdir. Sürükle-bırak editörünün bulunmasının yeni başlayan programcılar için faydalı olduğunu belirtmişlerdir.

Zaccone, Cooper ve Dann (2003) çalışmalarında programlamaya yeni başlayanların temel programlama bilgilerini öğrenmelerini, programlama bilgisi olanların bilgilerini sağlamlaştırmasını ve öğrencilerin işbirlikli projeler gerçekleştirmelerini sağlamak amacıyla Alice programını kullanmışlardır. Her ders görevler verilmiş ve süreç sonunda üçer kişilik grupların proje teslim etmesi istenmiştir. Araştırma verileri anket yardımıyla toplanmıştır. Anket sonucunda öğrencilerin programlamaya karşı ilgilerinin arttığı, dersten memnun kaldıkları görülmüştür. Çalışmada programlama ön bilgisi olmayan öğrencilerin bile temel kavramları öğrendikleri ve farklı projelerde kullanabildikleri, Alice programının programlamayı desteklediği belirtilmiştir.

Moskal vd. (2004), deneysel bir çalışma ile Alice programı kullanımının etkisini belirlemeyi amaçlamışlardır. İki kontrol bir deney grubuyla çalışmışlardır. Kontrol gruplarında geleneksel yöntem ile deney grubunda Alice programı ile eğitim verilmiştir. Araştırma sonucunda deney grubunun programlamaya karşı tutumlarının daha olumlu olduğu, daha akılda kalıcı şekilde öğrendikleri, performanslarının daha yüksek olduğu görülmüştür.

Howard vd. (2006) çalışmalarında anabilim dalı bilgisayar olmayan 89 öğrenciye Alice programı ile eğitim vermişlerdir. İki buçuk hafta süren eğitim sonunda öğrencilerden Alice programı deneyimlerini içeren yansı toplanmış ve öğrencilerle odak grup görüşme yapılmıştır. Araştırma sonucunda öğrencilerin % 69'u Alice programı ile rahat programlama yaptıklarını, programın kolay ve basit olduğunu belirtmişlerdir. Buna karşın öğrencilerin % 33'ü başlangıçta beklediklerinden daha karmaşık bir programlama sürecine girdiklerini düşünürken,% 26'sı programlama yeteneklerine güvensizlik duyduklarını belirtmişlerdir. Öğrencilerin çoğunluğu, Alice programındaki bir öykünün bir algoritma ile nasıl ilişkilendirildiğini anladıklarını ifade etmiştir. Öğrencilerin %4'ü Alice programı ile

programlama sürecinden hoşlanmadıklarını belirtmiştir. Öğrencilerin %91'i Alice programındaki kısıtlamayı sevmediklerini ifade etmiştir.

Bishop-Clark, Courte, Evans ve Howard (2007) çalışmalarında Alice programının kullanılmasını nicel ve nitel perspektiften incelemişlerdir. Nicel olarak programlama bilgisi, programlamaya olan beğeni ve güvenin incelendiği çalışmada nitel olarak öğrencilerden deneyimlerini yorumlamaları ve deneyimlerini içeren bir makale yazmaları istenmiştir. Araştırma sonucunda öğrencilerin programlamaya olan beğeni, programlamada kendine güven düzeyleri ve programlama kavramlarını anlamalarında anlamlı düzeyde artışın olduğu belirlenmiştir.

Sykes (2007) karma yöntem kullandığı çalışmasında programlama eğitiminde Alice programı kullanımının etkisini belirlemeyi amaçlamıştır. İki karşılaştırma, bir deney grubu ile çalışma yürütülmüştür. İki karşılaştırma grubunda geleneksel yöntemle, deney grubunda Alice programı ile eğitim verilmiştir. Her hafta öğrencilerle bireysel görüşmeler yapılmış, öğrencilere anket uygulanmış ve gözlem yapılmıştır. Araştırma sonucunda öğrenciler Alice programını kullanışlı, yararlı, eğlenceli, kullanımını kolay ve anlaşılır bulmuşlardır. Alice programının programlama öğrenmeyi kolaylaştırdığını belirtmişlerdir.

Cliburn (2008) çalışmasında üniversite öğrencilerine önce Alice programı ile ardından geleneksel yöntem ile Java programlama dilini öğretmiştir. Sonrasında Alice programına ait programlama süreçleri ile ilgili öğrencilerle görüşme yapmıştır. Öğrenciler Alice programının kolay olduğunu, programın görselliğini, kod hatası bulmak zorunda kalmamalarını ve kodları animasyon şeklinde görme özelliğini beğendiklerini, ayrıca Java programlama dili öğrenimini kolaylaştırdığını belirtmişlerdir. Ancak Alice programının gerektirdiği sistem belleği miktarı ve yükleme zamanının fazla olması gibi problemler beğenilmeyen özellikler olarak ifade edilmiştir. Çalışmada Alice programının yeni başlayan veya programlama bilgisi çok az olan kullanıcılar için uygun bir araç olduğu belirtilmiştir.

Wang vd. (2009) lise öğrencilerine C++ programlama dilini Alice programı ile öğretmeyi amaçladıkları çalışmada deneysel bir yöntem kullanmışlardır. Kontrol grubuna geleneksel yöntemle, deney grubuna Alice programı ile eğitim verilmiştir. Araştırmacılar Alice programını kullanan öğrencilerin derse katılım isteklerinin daha fazla olduğunu, döngü ve

kontrol yapıları gibi karmaşık kavramları daha iyi kullandıklarını, programın C++ programlama dili öğrenimini kolaylaştırdığını ifade etmiştir.

Werner, Denner, Bliesner ve Rex (2009) çalışmalarında, ortaokul öğrencileri için düzenlenen yaz kursunda, Alice programını kullanılarak öğrencilerin oyun programlama üzerine elde ettikleri deneyimlerini paylaşmışlardır. Eğitim sonrası öğrencilere uyguladıkları anket sonucuna göre Alice programının öğrencilerin düşündüğü oyunları oluşturmaya çok elverişli olduğu, eğlenceli ve kolay kullanılabildiği sonucuna ulaşılmıştır.

Garlick ve Cankaya (2010) temel programlama dilli yapılarını tanıtmak amacıyla yaptıkları deneysel çalışmada deney grubuna Alice programı ile, kontrol grubuna Java programlama dili ile iki yarıyıllık bir programlama eğitimi vermişlerdir. Araştırma sonuçlarına göre Java programlama dili ile çalışan öğrencilerin temel programlama dili yapılarını daha iyi anladıkları belirlenmiştir. Alice programından Java programlama diline geçerken öğrencilerin zorluk çektikleri belirtilmiştir. Anket sonucunda Java programlama dili ile öğrenim gören öğrencilerin, Alice programı ile çalışan öğrencilere göre temel programlama kavramlarını daha iyi anladıkları, programlamadan daha fazla zevk aldıkları, yeni bilgilere daha fazla açık oldukları ve programlamada araştırma isteklerinin daha fazla olduğu sonuçlarına ulaşılmıştır.

Daly (2011) Alice programının Java programlama dili öğrenimindeki etkisini belirlemek amacıyla deneysel bir çalışma yapmıştır. Deney grubu Java programlama dili bileşenlerine geçmeden önce altı hafta Alice programıyla sonrasında Java programlama dili ile, kontrol grubu süreç boyunca sadece Java programlama dili ile çalışmıştır. Deneysel sürecin başında, ortasında ve sonunda; öğrencilerin programlama kavramlarını öğrendikçe, kendilerine olan güven seviyelerini belirlemek amacıyla anketler uygulanmıştır. Araştırma sonucunda Alice programı kullanılarak Java programlama dili öğrenimi alan öğrencilerin programlamaya güven seviyelerinin ve temel programlama dili kavramları öğrenimlerinin sadece Java programlama dili ile çalışan öğrencilerden daha yüksek olduğu bulunmuştur.

J. Liu, Lin, Hasson ve Barnett (2011) öğretmenlerin bilgisayar bilgilerini geliştirmek amacıyla düzenledikleri eğitimde, Alice programını kullanmıştır. Çalışma sonucunda bilgisayar bilimleri öğretiminde, %52 güven seviyesinde artış ve %44 bilgi birikim seviyesinde artış olduğu tespit edilmiştir.

Solmaz (2014), Alice programı ile PHP programlama dilini öğretmeyi hedeflemiştir. Araştırmacı tarafından geliştirilen görüşme formu aracılığıyla öğrencilerle yapılan görüşmelerde Alice programının programlama ile algoritma mantığını kavramaya ve programlamanın temel kavramlarını öğrenmeye yardımcı olduğu, programlamayı eğlenceli hale getirdiği, içeriğin Alice ortamında öğrenilmesinin PHP programlama dilinde aynı içeriği uygulamayı kolaylaştırdığı, kodların ve sonuçlarının canlandırılmasının öğrenmeye yardımcı olduğu, programın eğlenceli olduğu, kodların canlandırılmasına olanak tanıdığı ve kodların hazır olmasının programlamaya yardımcı olduğu sonucuna ulaşılmıştır. Ayrıca öğrenciler Alice programı ile PHP programlama dili arasında bağlantı kuramama, Alice ortamı ile PHP programlama dilini bir arada götürmenin zor olması ve yazılımın yüksek donanım özellikleri gerektirmesi durumlarını programın olumsuz özellikleri olarak değerlendirmişlerdir.

Hayat vd. (2017) çalışmalarında, Alice programını, K-12 öğrencilerine öğretmiş ve etkilerini incelemişlerdir. Çalışma kapsamında öğrencilere, Alice programını kullanarak basit sahnelerin nasıl oluşturulacağı ve bu sahnelerde nesnelerin nasıl taşınacağı öğretilmiştir. Ardından proje gerçekleştirmeleri istenmiş ve süreç sonunda anket uygulanarak öğrencilerin görüşleri alınmıştır. Araştırmacıların gerçekleştirdiği nitel analiz sonucuna göre; öğrencilerin %61.6'sı Alice programını anlamanın kolay olduğunu, %73.3'ü programın kullanımının kolay olduğunu, %86.4'ü Alice programı ile çalışmanın eğlenceli ve motivasyonu artırıcı olduğunu belirtmişlerdir.

Alice programının programlama sürecine olan etkisinin incelendiği çalışmalarda verilerin genellikle görüşme ve anketlerden (Bishop-Clark vd., 2007; Cliburn, 2008; Daly, 2011; Hayat vd., 2017; Howard vd., 2006; J. Liu vd., 2011; Solmaz, 2014; Sykes, 2007; Werner vd., 2009; Zaccone vd., 2003) toplandığı görülmektedir. Çalışmalarda araştırmacıların ve öğrencilerin çoğu Alice programının kullanımını kolay (Bishop-Clark vd., 2007; Cooper vd., 2000a, 2000b, 2003; Hayat vd., 2017; Howard vd., 2006; Solmaz, 2014; Sykes, 2007; Werner vd., 2009), eğlenceli (Cooper vd., 2000a, 2000b; Hayat vd., 2017; Solmaz, 2014; Werner vd., 2009), programlama mantığını ve programlamanın temel kavramlarını öğrenmeye yardımcı (Bishop-Clark vd., 2007; Cliburn, 2008; Cooper vd., 2000a, 2000b; Daly, 2011; Howard vd., 2006; Jiau vd., 2009; J. Liu vd., 2011; Moskal vd., 2004; Solmaz, 2014; Sykes, 2007; Wang vd., 2009; Zaccone vd., 2003) olarak bulmuşlardır. Alice

programı ile çalışan öğrencilerin programlamada kendilerine güven seviyelerinde artış olduğunu belirtmişlerdir (Bishop-Clark vd., 2007; Cooper vd., 2000a, 2000b, 2003; Daly, 2011; J. Liu vd., 2011; Moskal vd., 2004). Alice programının programlama derslerine olan ilgiyi ve motivasyonu arttırdığını ifade etmişlerdir (Bishop-Clark vd., 2007; Cooper vd., 2000a, 2000b, 2003; Hayat vd., 2017; Wang vd., 2009). Ancak bazı çalışmalarda geleneksel programlama dili öğretimi ile öğrencilerin programlama bilgi birikim seviyelerinin ve programlamaya karşı olan ilgilerinin arttığı, Alice programı ile çalışırken öğrencilerin zorlandığı belirtilmiştir (Garlick & Cankaya, 2010). Alice programının sistem gereksiniminin yüksek olması çalışmalarda eleştirilen bir durumdur (Cliburn, 2008; Solmaz, 2014).

Alice programının programlama dili öğrenim sürecinde öğretim ortamlarında kullanımının incelendiği çalışmalar araştırılmıştır. Çalışmalara ait detaylı bilgi aşağıda verilmektedir.

Powers, Ecott ve Hirshfield (2007) çalışmalarında, Bilgisayar Bilimine Giriş sınıfında Alice programını kullanarak, deneyim ve gözlem yoluyla programın avantaj ve dezavantajlarını analiz etmişlerdir. Çalışmada önceden bilgisayar deneyimi olmayan ve matematik altyapısı iyi olmayan öğrencilerin Alice programını kullanarak programlama dersine geçmelerinin faydalı olabileceği belirtilmiştir. Program çıktılarının görsel olması, öğrencilerin kodların etkilerini görmelerini kolaylaştırdığını ifade etmişlerdir. Java ve C++ gibi nesne yönelimli programlama dili öğrenimine ilk geçişi kolaylaştırdığı ancak ileri düzey programlama için yetersiz kaldığını vurgulamışlardır.

Brown (2008) Bilgisayara Giriş dersinde Java programlama dili öğreniminde Alice programı kullanan ve kullanmayan öğrencilerin programlama süreçlerini incelemiştir. Araştırma sonucunda Alice programı ile öğrenim gören öğrencilerin daha yaratıcı uygulamalar geliştirdiği, motivasyonlarının ve kendine güvenlerinin arttığı belirtilmiştir. Soyut kavramların öğretiminde Alice programının yararlı olduğu ifade edilmiştir. Sürükle-bırak mantığı ile doğru sözdizimi yapıldığı belirtilmiştir. Ancak programın donanımsal gereksinimlerinin fazla olması, Alice 2'nin kalıtım yönünün zayıf olması ve programdaki karakterler ile sahnelerin sınırlı nitelikte olmasından dolayı farklı uygulamaların yapılamadığı Alice programının eksik yönleri olarak ifade edilmiştir. Ayrıca Alice programı kullanılarak programlama yapma ile geleneksel programlama dilleri kullanarak programlama yapma arasında uçurumun olduğu belirtilmiştir.

McKenzie (2009) azalan öğrenci ilgilisi sebebiyle, bilgisayar bilgi sistemleri branşını yeniden canlandırmaya yardımcı olmak amacıyla Alice programı kullanarak yeni bir müfredat geliştirmiştir. Alice programının programlama dili değil bir öğrenme ortamı olduğunu ifade etmiştir. Programın öğrencileri programlama yazmaktan uzaklaştırdığını belirtmiştir.

Mullins, Whitfield ve Conlon (2009) Programlamaya Giriş kursunda Alice programının kullanım şeklini araştırmak amacıyla, farklı dönemlerde öğrenim gören öğrencilere uygulanan sınavlarla elde edilen verilerin değerlendirildiği bir çalışma yapmışlardır. Araştırma sonucunda Alice programının kullanılmasıyla kursu geçen öğrenci sayısının arttığı ve kursu bırakan öğrenci sayısının azaldığı tespit edilmiştir.

Aktunc (2013) Alice programı kullanılarak Java programlama diline geçiş sürecine bir yaklaşım sunmuştur. Çalışmasında öğrenme hedeflerine sadece Alice programı ile ulaşılamayacağını, Alice programının programlamanın temel kavramlarının öğretiminde kullanılması gerektiğini belirtmiştir. Alice programı ve Java programlama dili birleşiminin, öğrencilere sağlam bir programlama geçmişi vereceğini ve öğrencileri üst düzey programlama dersleri için hazırlayacağını ifade etmiştir.

Kayabaşı (2016), öğretmen adaylarının Alice programına ilişkin deneyimlerinin değerlendirilmesi ve öz yeterlilik algılarındaki değişimin belirlenmesi amacıyla bir çalışma yapmıştır. Araştırmada deneyimleri içeren veriler yarı yapılandırılmış görüşme formu ile, nicel veriler anket yardımıyla toplanmıştır. Araştırma sonucunda katılımcılar Alice programı ile programlama derslerine olan ilginin ve katılımın artabileceği, programın ders başarısını olumlu yönde etkileyebileceği, kalıcılığı arttırabileceği, liderlik, yaratıcı düşünme ve problem çözme becerilerini geliştirebileceği yönünde görüş belirtmişlerdir. Ayrıca Alice programının eğlenerek öğrenmeye olanak tanıdığını, hayal güçlerini kullanarak somutlaştırma imkânı verdiğini ifade etmişlerdir. Anket sonucunda Alice programının istatistiksel olarak öz yeterlilik algısını olumlu yönde etkilediği belirtilmiştir. Araştırmacı programlamaya giriş derslerinde Alice programının kullanımının uygun olabileceğini ifade etmiştir.

Araştırmalarda özellikle programlamaya yeni başlayan kullanıcıların Alice programı ile programlamaya başlamalarının, programlama mantığını öğrenme ve sağlam bir

programlama temeli oluşturma açısından yararlı olacağı vurgulanmıştır (Aktunc, 2013; Kayabaşı, 2016; Powers vd., 2007). Alice programının soyut kavramların öğreniminde yararlı olduğu belirtilmiştir (Brown, 2008; Kayabaşı, 2016; Powers vd., 2007). Alice programı ile çalışan öğrenciler programlamada kendilerine olan güvenlerinin arttığını ifade etmişlerdir (Brown, 2008; Kayabaşı, 2016). Ancak programın ileri düzey programlama yapmak için yetersiz olduğu (Powers vd., 2007) ve Alice programının bir programlama dili değil programlama öğrenme ortamı olmasından dolayı, programlama diliyle programlama yapma ile Alice programıyla programlama yapma arasında büyük bir farkın olduğu belirtilmiştir (Brown, 2008; McKenzie, 2009).

Yapılan çalışmalar incelendiğinde Alice programının özellikle programlama bilgisi olmayan veya az olan öğrenciler üzerinde, programlamanın temel kavramlarının ve programlama yapısının öğrenilmesini arttırdığı, programlama dili öğrenimini kolaylaştırdığı, öğrencileri ders sürecinde aktif kıldığı ve öğrenciler tarafından kullanımının kolay bulunduğu görülmektedir. Alanyazın incelendiği zaman Alice programı ile ilgili yapılan çalışmaların sınırlı sayıda olduğu tespit edilmiştir. Özellikle Alice programının motivasyon ve problem çözme becerisi algısına olan etkisinin incelendiği, öğretimde nerede ve nasıl kullanılmasının uygun olacağına dair yeni araştırmalara ihtiyaç duyulduğu görülmektedir.

BÖLÜM III

YÖNTEM

Bu bölümde çalışmada kullanılan araştırma yöntemleri, evren ve örneklem, kullanılan veri toplama araçları ve araştırma sorularının nasıl analiz edileceği ile ilgili bilgilere yer verilmektedir.

Araştırma Modeli

Araştırmada karma yöntemlerden sıralı açıklayıcı desen kullanılmıştır. Sıralı açıklayıcı desen; daha çok nicel veriler toplandıktan sonra nitel verilerin toplandığı ve veri analizlerinin birbiriyle ilişkili olup olmadığının incelendiği desendir (Creswell & Clark, 2011). Araştırmanın nicel boyutunda deneysel yöntem tercih edilmiştir. Deneysel modeller, değişkenler arasındaki neden sonuç ilişkilerini keşfetmeyi amaçlamaktadır. Burada amaç, iç geçerliliği korumak için bozucu olan dış değişkenleri kontrol altında tutmak şartıyla, bağımsız değişkenler manipüle edilerek bağımlı değişkenler üzerinde ölçme yapabilmektir (Büyüköztürk, 2001).

Araştırmada deneysel model olarak; yansız atama ile oluşturulup deney ve karşılaştırma grupları olarak adlandırılmış çalışma grupları bulunan ve gruplara çalışma öncesi öntest, çalışma sonrası sontest uygulanan *Öntest-Sontest Kontrol Gruplu Yarı Deneysel Model* kullanılmıştır.

Öntest-sontest kontrol gruplu yarı deneysel modelin şematik gösterimi Tablo 2'deki gibidir:

Tablo 2

Modelin Simgesel Görünümü

G_1	T_1	R	X	T_2	T_3
G_2	T_1	R		T_2	T_3

Modelde kullanılan simgelerin anlamı aşağıda belirtilmiştir:

G_1 : Deney grubu

G_2 : Karşılaştırma grubu

R : Grupların oluşturulmasındaki yansızlık

X : Alice programı ile tasarlanan öğrenme ortamı

T_1 : Deney öncesi ölçme (Akademik başarı testi, problem çözme becerisi algısı ölçeği, motivasyon ölçeği)

T_2 : Deney sonrası ölçme (Akademik başarı testi, problem çözme becerisi algısı ölçeği, motivasyon ölçeği)

T_3 : Programlamaya hazır bulunuşluk düzeyi belirleme testi

Araştırmanın bağımsız değişkeni Alice programı ile tasarlanan öğrenme ortamıdır. Bağımlı değişkenler ise öğrencilerin akademik başarıları, problem çözme becerisi algıları, motivasyonları ve programlamaya hazır bulunuşluk düzeyleridir.

Araştırmada önceden oluşturulmuş olan gruplardan biri deney, diğeri karşılaştırma grubu olarak yansız bir şekilde atanmıştır. Deneysel süreç öncesinde gruplara akademik başarı testi, problem çözme becerisi algısı ölçeği ve motivasyon ölçeği öntest olarak uygulanmıştır. Deneysel sürecin ardından gruplara; akademik başarı testi, problem çözme becerisi algısı ölçeği ve motivasyon ölçeği sontest olarak uygulanmıştır. Öğrencilerin programlamaya hazır bulunuşluk düzeylerini belirlemek amacıyla deneysel süreçten 18 hafta sonra yani deneysel sürecin yürütüldüğü dönemden bir sonraki dönemde, başka bir programlama dersinin başlangıcında programlamaya hazır bulunuşluk düzeyi belirleme testi uygulanmıştır. Bu araştırmada Java programlama dili; deney grubunda Alice programı kullanılarak, karşılaştırma grubunda herhangi bir ek araç kullanılmadan öğretilmiştir.

Deneysel süreç sonunda Alice programı ile programlama öğretimi gören deney grubunda yer alan öğrencilerle, Alice programının kullanımında engelleyici ve kolaylaştırıcı faktörleri belirlemek amacıyla iki odak gruplu görüşme yapılmıştır.

Çalışmanın Bağlamı

Araştırma Nevşehir ilinde, Nevşehir Hacı Bektaş Veli Üniversitesi Meslek Yüksekokulunda yapılmıştır. Meslek Yüksekokulu'nda toplamda dört bilgisayar laboratuarı bulunmaktadır. Çalışma iki bilgisayar laboratuarında gerçekleştirilmiştir. Laboratuarın birinde 38, diğerinde 39 masaüstü bilgisayar bulunmaktadır. Ayrıca her laboratuarda bir adet projeksiyon cihazı ve internet bağlantısı bulunmaktadır.

Araştırmanın çalışma gurubunu meslek yüksekokulunda öğrenim gören bilgisayar teknolojileri bölümü öğrencileri oluşturmaktadır. Bireylerin bilgi ve becerilerini arttırarak işgücü piyasasında aktif rol almaları, mesleki ve teknik eğitimin önemini ön plana çıkarmaktadır (Kalkınma Bakanlığı, 2014). Mesleki ve teknik eğitim veren kurumlardan biri de meslek yüksekokullarıdır.

Meslek yüksekokulları; sektörlere ara eleman yetiştirme misyonunu üstlenmiş, uygulama ve teoriye dayalı eğitim veren kurumlardır. Bu kurumlar; meslekler arasında geçişlerin veya yeni bir mesleğe uyum sürecinin sağlanmasını, yenilikçiliğin ve girişimciliğin yaygınlaştırılmasını desteklemektedir. Bireylerin istihdamlarının arttırılması amacıyla kurum ve sektörlerin işbirliği içinde olduğu katılımcı bir anlayışla eğitim verilmektedir (Kalkınma Bakanlığı, 2014).

Her geçen gün sayıları artmakta olan bu kurumların sayıları 2017-2018 öğretim yılı Yükseköğretim Bilgi Yönetim Sistemi verilerine göre 876 devlet üniversitesi, 100 vakıf üniversitesi ve 5 vakıf meslek yüksekokulu olmak üzere toplamda 981'e ulaşmaktadır. Kuruluş misyonları doğrultusunda farklı alanlarda istihdam edilmek üzere bireyler yetiştirilmektedir.

Meslek Yüksekokulları Bilgisayar Teknolojileri Bölümünde Eğitim

Türkiye'de programlama eğitimi bilgisayar programcılığı, bilgisayar mühendisliği, yazılım mühendisliği gibi bölümlerde verilmektedir (Ölçme Seçme ve Yerleştirme Merkezi [ÖSYM], 2011a, 2011b). Bu araştırmanın çalışma grubunu meslek yüksekokulu bilgisayar teknolojileri bölümü öğrencileri oluşturmaktadır. Meslek yüksekokulları, Türkiye'de yazılım sektörünün yetişmiş insan gücü talebini öncelikli olarak sağlayan kurumlardır. 2017-2018 öğretim yılı Yükseköğretim Bilgi Yönetim Sistemi verilerine göre bilişim ve iletişim teknolojileri bölümlerinde ön lisans ve lisans düzeydeki toplam öğrenci sayısı 94.794'tür. Meslek yüksekokullarında ilgili bölümlerde öğrenim gören öğrenci sayısı 87.305'tir. Yani %92'lik bir oranla Türkiye'nin bilişim ve iletişim teknolojileri sektörünün yetişmiş insan gücünü meslek yüksekokulları karşılamaktadır.

Bilgisayar teknolojileri bölümünde öğrencilere; yazılım, donanım ve temel bilişim bilgileri alabilecekleri dersler verilmektedir. Bölümden mezun olabilmek için bölüm derslerinden başarılı olmanın yanında 30 iş günü kamu veya özel sektörün bilişim faaliyeti gösteren birimlerinde staj yapma zorunluluğu vardır. Bilgisayar teknolojileri bölümünde verilen iki yıllık eğitim sonunda öğrenciler tekniker unvanı almaktadır. Kamu ve özel sektörde; sistem analisti, teknik servis sorumlusu, bilgisayar programcısı, donanım personeli, veritabanı asistanı, bilgisayar operatörü, internet programlayıcısı ve web tasarımcısı olarak çalışabilmektedirler. Araştırmanın yürütüldüğü Nevşehir ilinde bilgisayar teknolojileri bölümünden mezun olan öğrenciler kamu ve özel sektörde çalışma imkânı bulmaktadır.

İki yıllık öğrenim gören bireyler bu süreç içerisinde teknolojiyi farklı ortamlarda kullanabilecekleri, programlama altyapısına sahip olabilecekleri, problem çözme yeteneklerini geliştirebilecekleri ayrıca gelişen teknolojiyi kolaylıkla izleyebilecekleri içeriğe sahip dersler almaktadır (Baştemur Kaya & Çakır, 2015). Ancak günümüzde öğretim ortamlarında düz anlatım, gösterip yaptırma vb. tekniklerin kullanılması, öğrencilerin derste aktif olmasını sağlayan stratejilerin benimsenmemesi, öğrencilerin bu derslerde edinmeleri gereken becerileri kazanamamalarına neden olabilmektedir (Binici & Necdet, 2004; Şahin & Fındık, 2008; Ünal & Çakır, 2016).

Sektörün Bilgisayar Teknolojileri Bölümünden Beklentisi

Bilişim teknolojileri sektörü, küresel düzeyde pazar ve rekabet koşullarının hızla değişmesi sebebiyle sürekli ve dinamik bir gelişim içindedir. Bu özellikleri nedeni ile bilişim teknolojileri sektörü, stratejik bir sanayi olarak ülkelerin yakın ilgisini çekmekte ve bu sektör için devletler tarafından özel planlamalar yapılmaktadır. Özellikle hızla küreselleşen iş dünyasında rekabet büyük yoğunluk kazanmakta, sanayileşmiş ülkeler bu sektörün korunması ve rekabet gücünün geliştirilmesi için özel politikalar uygulamaktadır.

Bilişim teknolojileri alanından mezun olan öğrenciler bilgisayar teknik servisi hizmeti veren bilgisayar firmalarında; kamu kurum ve kuruluşlarında; ağ kurulum ve yönetimi hizmeti veren ya da bu hizmete ihtiyaç duyan firma, kamu kurum ve kuruluşlarında; kullanıcı arayüzüne sahip uygulama ve veri tabanı programları kullanımı ve yönetimi hizmeti veren ya da bu hizmetlere ihtiyaç duyan firma, kamu kurum ve kuruluşlarında; web tasarımı hizmeti veren veya web ortamında çalışan etkileşimli programlar hazırlayan yazılım şirketlerinde ya da bu hizmetlere ihtiyaç duyan firma, kamu kurum ve kuruluşlarında çalışabilirler (Mesleki ve Teknik Eğitim Programlar ve Öğretim Materyalleri [MEGEP], 2007).

Evren ve Örneklem

Araştırmanın evrenini, 2015-2016 eğitim-öğretim yılı Nevşehir ilinde bulunan Nevşehir Hacı Bektaş Veli Üniversitesi Meslek Yüksekokullarında öğrenim gören Bilgisayar Teknolojileri Bölümü öğrencileri oluşturmaktadır. Araştırmanın örneklemini çalışma evreninde bulunan Meslek Yüksekokulu Bilgisayar Teknolojileri Bölümü 1. sınıfta öğrenim gören 34 normal öğretim, 29 ikinci öğretim öğrencisi olmak üzere toplam 63 öğrenci oluşturmaktadır. Normal öğretim ve ikinci öğretim öğrencilerinin öğrenim zamanları yönetmeliğe göre farklılık göstermektedir. Bu öğrencilerin aynı anda ders görmeleri durumunda uygulama sürecine katılımın sağlanmasında sıkıntı yaşanma ihtimali bulunmaktadır. Bu durum çalışmayı aksatabileceğinden çalışma kapsamında normal öğretim ve ikinci öğretimde yer alan öğrenciler karıştırılarak deney ve karşılaştırma grubu olarak atanmamıştır.

Önceden oluşturulmuş bu iki grubun denklik durumlarına bakmak amacıyla öğrencilerin bir önceki dönem Programlama Temelleri dersi başarı ortalamaları ile 1. dönem genel akademik not ortalamaları (GANO) karşılaştırılmıştır. Karşılaştırma için analiz yöntemi belirlenirken öncelikle verilerin normal dağılım gösterip göstermediği Kolmogorov-Smirnov (K-S) ve Shapiro-Wilk testleri ile incelenmiş, sonrasında varyansların homojenlik durumu Levene testi ile belirlenmiştir. K-S ve Shapiro-Wilk testleri sonuçların göre verilerin ,05 anlamlılık düzeyinden büyük olduğu ve normal dağılım gösterdiği tespit edilmiştir. Levene testi sonucu incelendiğinde; ,05 anlamlılık düzeyine göre varyansların homojen olduğu belirlenmiş ve parametrik testlerden ilişkisiz ölçümler için t testi kullanımının uygun olduğu görülmüştür. Tablo 3'te ilişkisiz ölçümler için t testi kullanılarak analiz edilen Programlama Temelleri dersi ve GANO verileri sonuçları verilmektedir.

Tablo 3

Programlama Temelleri Dersi ile GANO Verileri t Testi Sonuçları

	Grup	**n**	**X̄**	**SS**	**sd**	**t**	**p**
GANO (p değeri)	**Deney**	34	2,35	,46	61	1,79	,07
	Karşılaştırma	29	2,12	,57			
Programlama Temelleri Dersi (p değeri)	**Deney**	34	46,79	25,52	61	,13	,89
	Karşılaştırma	29	47,69	27,88			

Tablo 3 incelendiğinde ilişkisiz ölçümler için t testi sonucuna göre grupların GANO ve Programlama Temelleri dersi puanları arasında anlamlı bir fark bulunmadığı (p>,05) görülmektedir. Analizler sonucunda grupların eşit olduğu varsayılmış ve kura çekilerek normal öğretim öğrencileri deney grubu, ikinci öğretim öğrencileri karşılaştırma grubu olarak yansız atama yoluyla belirlenmiştir.

Grupların cinsiyet dağılımları, yaş aralıkları, lise mezuniyet derece ortalamaları, mezun oldukları okul türü, önceden programlama dersi alıp almadıkları ve bölümlerini neden seçtikleri ile ilgili bilgilere ilişkin demografik özellikler ayrıntılı bir şekilde incelenmiştir.

Deney ve karşılaştırma grubundaki öğrencilerin cinsiyet bilgilerini içeren betimsel analiz sonucu Tablo 4'te verilmektedir.

Tablo 4

Öğrencilerin Cinsiyetlerinin Gruplara Göre Dağılımı

	Deney Grubu		**Karşılaştırma Grubu**		**Toplam**	
	n	**%**	**n**	**%**	**n**	**%**
Kadın	13	20,63	9	14,28	22	34,92
Erkek	21	33,33	20	31,74	41	65,07
Toplam	34	53,96	29	46,03	63	100

Tablo 4 incelendiğinde deney grubunda 13 kadın, 21 erkek öğrenci; karşılaştırma grubunda 9 kadın, 20 erkek öğrenci bulunmaktadır. Deney grubu çalışma grubunun %53,96'sını, karşılaştırma grubu %46,03'ünü oluşturmaktadır. Grupların yaş aralığı bilgileri Tablo 5'te verilmektedir.

Tablo 5

Öğrencilerin Yaş Aralığı Dağılımları

Yaş Aralığı	**Grup**				**Toplam**	
	Deney		**Karşılaştırma**			
	n	**%**	**n**	**%**	**n**	**%**
18-20	30	47,61	24	38,09	54	85,71
21-23	2	3,17	3	4,76	5	7,93
24 ve üzeri	2	3,17	2	3,17	4	6,34
Toplam	34	53,96	29	46,03	63	100

Tablo 5 incelendiğinde deney grubunda yer alan 30 öğrencinin yaş aralığının 18-20, iki öğrencinin 21-23, iki öğrencinin 24 ve üzeri olduğu; karşılaştırma grubunda yer alan 24 öğrencinin yaş aralığının 18-20, üç öğrencinin 21-23, iki öğrencinin 24 ve üzeri olduğu görülmektedir. Çalışma grubunun %85,71 ile en yüksek yaş aralığının 18-20 olduğu belirlenmiştir. Öğrencilerin cinsiyete göre yaş aralığına bakıldığında, deney ve

karşılaştırma grubunda yer alan öğrencilerin benzer yaş aralığında olduğu görülmektedir. Grupların lise mezuniyet derecesi ortalamaları Tablo 6'da verilmektedir.

Tablo 6

Grupların Lise Mezuniyet Derecesi Ortalamaları

	Grup	
	Deney	**Karşılaştırma**
Gruplara Göre Genel Ortalama	71,58	66,27

Deney ve karşılaştırma grubundaki öğrencilerin lise mezuniyet derece ortalamaları incelendiğinde deney grubundaki öğrencilerin ortalamasının 71,58; karşılaştırma grubundaki öğrencilerin ortalamalarının 66,27 olduğu belirlenmiştir. Grupların lise mezuniyet ortalamalarının birbirine yakın olduğu görülmektedir. Grupların mezun oldukları okul türü bilgileri Tablo 7'de verilmektedir.

Tablo 7

Grupların Mezun Oldukları Okul Türü Dağılımları

Mezun Olunan Okul Türü	**Grup**				**Toplam**	
	Deney		**Karşılaştırma**			
	n	**%**	**n**	**%**	**n**	**%**
Genel Lise	6	9,52	5	7,93	11	17,46
Meslek veya Teknik Lise	10	15,87	16	25,39	26	41,26
Anadolu Meslek veya Teknik Lise	14	22,22	5	7,93	19	30,15
Diğer	4	6,34	3	4,76	7	11,11
Toplam	34	53,96	29	46,03	63	100

Tablo 7 incelendiğinde deney grubundaki öğrencilerin altısının genel liseden, 10'unun meslek veya teknik liseden, 14'ünün anadolu meslek veya teknik lisesinden, dördünün belirtilen lise türlerinden farklı bir liseden mezun olduğu; karşılaştırma grubundaki öğrencilerin beşinin genel liseden, 16'sının meslek veya teknik liseden, beşinin anadolu meslek veya teknik lisesinden, üçünün belirtilen lise türlerinden farklı bir liseden mezun

olduğu görülmektedir. Meslek veya teknik liselerden mezun olma oranının %41,26 ile çalışma grubunun en yüksek mezun olunan okul türü olduğu belirlenmiştir. Deney gurubu ve karşılaştırma grubunda yer alan öğrencilerin mezun olunan lise türü dağılımlarının birbirine benzer olduğu görülmektedir. Öğrencilerin çalışmanın yapıldığı bölümde öğrenim görmeye başlamadan önce herhangi bir programlama dersi alıp almadıkları bilgileri Tablo 8'de verilmektedir.

Tablo 8

Grupların Bölümlerinde Öğrenim Görmeden Önce Programlama Dersi Alma Bilgileri

Programlama Dersi Alma Bilgisi	**Grup**				**Toplam**	
	Deney		**Karşılaştırma**			
	n	**%**	**n**	**%**	**n**	**%**
Evet	26	41,26	16	25,39	42	66,66
Hayır	8	12,69	13	20,63	21	33,33
Toplam	34	53,96	29	46,03	63	100

Tablo 8 incelendiğinde deney grubundaki öğrencilerin 26'sının, karşılaştırma gurundaki öğrencilerin 16'sının daha önceden programlama dersi aldığı görülmektedir. Çalışma grubundaki öğrencilerin %66,66 ile mevcut bölümlerinde öğrenim görmeye başlamadan önce programlama dersi aldıkları belirlenmiştir. Grupların bölümlerini seçme nedenlerinin dağılım bilgileri Tablo 9'da verilmektedir.

Tablo 9

Grupların Bölümlerini Seçme Nedenleri

Bölüm Seçme Nedeni	**Deney**		**Karşılaştırma**		**Toplam**	
	n	**%**	**n**	**%**	**n**	**%**
Bölümü sevdiğim için seçtim	10	15,87	15	23,80	25	39,68
Ailemin ve/veya çevremin bu bölümde okumamı istediği için seçtim	4	6,34	3	4,76	7	11,11
Daha fazla iş imkânı olduğu için seçtim	6	9,52	5	7,93	11	17,46
Toplumda bu mesleğe saygı duyulduğu için seçtim	1	1,58	-	-	1	1,58
Puanıma göre herhangi bir bölümü seçtim	13	20,63	6	9,52	19	30,15
Toplam	34	53,96	29	46,03	63	100

Tablo 9 incelendiğine deney grubundaki öğrencilerin 13'ünün puanlarına göre herhangi bir bölümü seçtikleri, 10'unun bölümü sevdikleri için seçtikleri, altısının daha fazla iş imkânı olduğu için bölümü seçtikleri, dördünün aile ve/veya çevrenin bu bölümde okumalarını istedikleri için seçtikleri ve birinin toplumda bu mesleğe saygı duyulduğu için seçtiği belirlenmiştir. Karşılaştırma grubundaki öğrencilerin 15'inin bölümlerini sevdikleri için seçtikleri, altısının puanlarına göre herhangi bir bölümü seçtikleri, beşinin daha fazla iş imkânı olduğu için bölümü seçtikleri ve üçünün aile ve/veya çevrenin bu bölümde okumalarını istedikleri için seçtikleri görülmektedir. %39,68 ile çalışma grubundaki öğrencilerin bölümü sevdikleri için seçtikleri belirlenmiştir.

Araştırmada deney grubunda yer alıp odak gruplu görüşme yapılan öğrencilerin belirlenmesinde amaçlı örnekleme yöntemi içerisinde yer alan maksimum çeşitlilik örnekleme kullanılmıştır. Maksimum çeşitlilik örnekleme ile çalışma grubu çeşitliliğini maksimum derecede yansıtmak amaçlanmaktadır (Yıldırım & Şimşek, 2013). Farklı özellikteki gruplardan ortaya çıkan herhangi bir ortak örüntü, özellikle ilgi çekicidir ve

etkisi araştırılan faktörün ortak boyutlarını ve temel deneyimleri yakalamak adına değerlidir (Patton, 2014). Araştırmanın yapıldığı Nesne Tabanlı Programlama I dersinin vize sınavından üst (100-67 arası), orta (66-34 arası) ve alt (33-0 arası) düzeyde puan olan öğrenci gruplarından her bir odak grup görüşme için üçer kişi seçilmiştir. Yapılan iki odak grup görüşmesinde; her odak grupta 9, toplamda 18 öğrenci ile odak gruplu görüşmeler yapılmıştır. 34 öğrencinin bulunduğu deney grubu; üst, orta ve alt düzeyde yer alan öğrencilere göre gruplandırıldığında, her bir grupta yer alan öğrenci sayıları Tablo 10'da verilmektedir.

Tablo 10

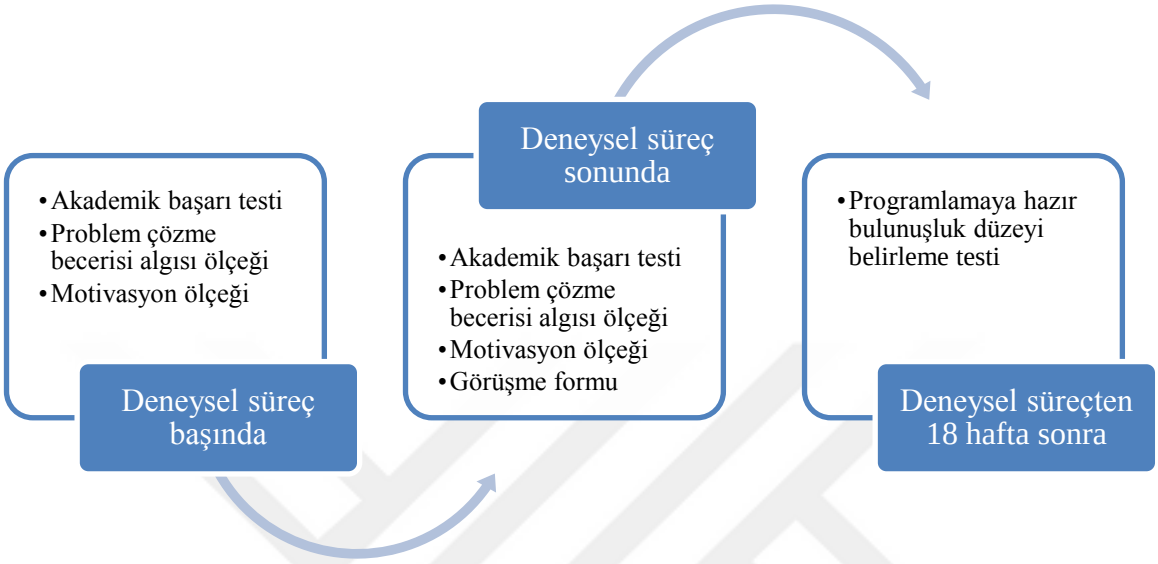
Deney Grubu Öğrencilerinin Üst, Orta ve Alt Düzeye Göre Gruplandırılması

	Üst Düzey	**Orta Düzey**	**Alt Düzey**	**Toplam**
Öğrenci Sayıları	9	12	13	34

Veri Toplama Araçları

Bu çalışmada Alice programının öğrencilerin Java programlama dili akademik başarısına etkisini belirlemek amacıyla Nesne Tabanlı Programlama I dersi için öntest ve sontest olarak kullanılmak üzere bir akademik başarı testi hazırlanmış ve başarı ölçme aracı olarak kullanılmıştır. Deneysel süreç sonunda Alice programının öğrencilerin programlamaya hazır bulunuşluk düzeyine etkisini belirlemek amacıyla genel programlama bilgisi içeren programlamaya hazır bulunuşluk düzeyi belirleme testi hazırlanmıştır. Alice programının problem çözme becerisi algısına olan etkisini belirlemek amacıyla deneysel sürecin başında ve sonunda Heppner ve Peterson (1982) tarafından geliştirilen; Şahin, Şahin ve Heppner (1993) tarafından Türkçeye uyarlanan problem çözme becerisi algısı ölçeği kullanılmıştır. Alice programının motivasyon üzerine etkisini belirlemek amacıyla deneysel sürecin başında ve sonunda Pintrich, Smith, Garcia ve McKeachie (1991) tarafından geliştirilen, Büyüköztürk, Akgün, Özkahveci ve Demirel (2004) tarafından Güdülenme ve Öğrenme Stratejileri Ölçeği (GÖSÖ) olarak Türkçeye uyarlanan Motivated Strategies for Learning Questionnaire (MSLQ) ölçeğinin motivasyon ölçeği bölümü kullanılmıştır. Ölçeklerin kullanımı için gerekli izinler, Türkçeye uyarlayan yazarlardan alınmıştır. Alice programı ile ilgili deney grubunda yer alan öğrenci değerlendirmelerini

belirlemek amacıyla açık uçlu sorulardan oluşan bir görüşme formu hazırlanmıştır. Araştırmada kullanılan ölçme araçlarının öğrencilere uygulanma zamanları Şekil 6'da verilmektedir.



Şekil 6. Uygulama sürecinde kullanılan ölçme araçlarının kullanım zamanları

Başarı Testleri

Alice programının öğrenci başarısı üzerine etkisini belirlemek amacıyla Nesne Tabanlı Programlama I dersindeki hedef çıktıları yoklayacak, deneysel sürecin başında (öntest) ve sonunda (sontest) uygulanmak üzere bir akademik başarı testi hazırlanmıştır. Testin geliştirilmesi aşamasında araştırmacı ile Nesne Tabanlı Programlama I ve II derslerini vermiş bir konu alanı uzmanının çeşitli kaynaklardan derlediği (ilgili test kitapları vb.), uygulama sürecindeki tüm hedefleri içeren ve bilişsel alan taksonomisinin farklı düzeylerindeki soruları kapsayan 55 soruluk başarı testi oluşturulmuştur. Geliştirilen başarı testi için uygulama sürecinde verilecek içeriğe göre bir belirtke tablosu hazırlanmıştır. Belirtke tablosu, öğretim sürecinde verilen içerikle hangi düzeyde davranışların kazandırılacağını tanımlar. Bir testin kapsamı, konu ve davranış boyutlarıyla birlikte iki boyutlu bir belirtke tablosu ile belirlenmiş olmalıdır (Özçelik, 1989). Bu yüzden testteki farklı bilişsel seviyelerde olan sorular belirtke tablosundaki uygun hedef çıktıya karşılık

gelecek şekilde belirtilmiştir. Oluşturulan test ile ilgili bilgisayar teknolojileri bölümünde daha önce bu dersi vermiş üç alan uzmanının görüşüne başvurulmuştur. Daha önce bu dersi almış iki öğrenciye sorular okutulmuş ve soruların anlaşılırlığı kontrol edilmiştir. Nesne Tabanlı Programlama I dersinin hedef çıktılarına ve öğrenci yapısına uygunluğu dikkate alınarak; bazı sorular eklenmiş, bazı sorular değiştirilmiş, bazı sorular düzeltilmiştir. Bu adımlar sonunda kapsam geçerliliğine karar verilen çoktan seçmeli beş şıklı 62 sorudan oluşan akademik başarı testi pilot uygulama için hazır hale getirilmiştir.

Alice programının öğrencilerin programlamaya hazır bulunuşluk düzeylerine etkisini belirlemek amacıyla, programlamaya hazır bulunuşluk düzeyini ölçen ve genel programlama bilgisi sorularını içeren bir akademik başarı testi hazırlanmıştır. Testin geliştirilmesi aşamasında araştırmacı ile birlikte programlama dersi veren dört alan uzmanı, programlamaya hazır bulunuşluk düzeyi belirleme testi için hedef çıktıları belirlemiştir. Araştırmacı ile birlikte programlama dersi veren bir alan uzmanının çeşitli kaynaklardan derlediği (ilgili test kitapları vb.), programlama dillerinin ortak konularını içeren ve bilişsel alan taksonomisinin farklı düzeylerindeki soruları kapsayan 32 soruluk bir test oluşturulmuştur. Geliştirilen programlamaya hazır bulunuşluk düzeyi belirleme testi için bir belirtke tablosu hazırlanmıştır. Farklı bilişsel seviyelerde olan sorular belirtke tablosundaki uygun hedef çıktıya karşılık gelecek şekilde belirtilmiştir. Oluşturulan test ile ilgili bilgisayar teknolojileri bölümünde programlama dersi veren üç alan uzmanının görüşüne başvurulmuştur. Daha önce programlama dersi almış iki öğrenciye sorular okutulmuş ve soruların anlaşılırlığı kontrol edilmiştir. Programlama derslerinin ortak konularının hedef çıktılarına ve öğrenci yapısına uygunluğu dikkate alınarak; bazı sorular eklenmiş, bazı sorular değiştirilmiş, bazı sorular düzeltilmiştir. Bu adımlar sonunda kapsam geçerliliğine karar verilen çoktan seçmeli beş şıklı 40 sorudan oluşan programlamaya hazır bulunuşluk düzeyi belirleme testi pilot uygulama için hazır hale getirilmiştir.

Pilot uygulamada, öntest ve sontest olarak kullanmak üzere hazırlanan akademik başarı testi ve programlamaya hazır bulunuşluk düzeyi belirleme testi, çalışmanın gerçekleştirildiği okuldaki daha önce Nesne Tabanlı Programlama I dersini almış 83 ikinci sınıf öğrencisine uygulanmıştır. Testlerdeki her bir soru 1 puan değerinde olacak şekilde toplam puanlar hesaplanmış ve test maddelerine ilişkin madde analizleri yapılmıştır.

Madde analizi ile testteki sorular, madde güçlük ve ayırt ediciliklerine göre değerlendirilmiştir. Madde ayırt edicilik indeksi ile ilgili değerler Tablo 11'de verilmiştir.

Tablo 11

Madde Ayırt Edicilik İndeksi ile İlgili Değerler

Madde Ayırt Edicilik İndeksi	**Değer**
,30 ve üstü	Madde kabul edilebilir sınırlar içinde, iyi bir madde
,20 - ,29 arası	Düzeltilerek kullanılabilen madde
,19 ve altı	Atılmalısı gereken, nadiren tümüyle düzeltilip kullanılabilen madde

Tablo 11'deki değerlere göre testlerde yer alan sorulardan ayırt edicilik indeksi ,30 ve üstünde değere sahip olan sorular testlere alınmıştır. Madde analizinde diğer önemli ölçüt madde güçlük indeksidir. Maddenin ,50 civarı değer alması madde güçlüğü için ideal olarak ifade edilmektedir. Madde güçlük değerinin bire yaklaşması maddenin kolay, sıfıra yaklaşması maddenin zor bir madde olduğunu göstermektedir (Tekin, 2000). Madde güçlük indeksi ilgili değerler Tablo 12'de verilmiştir.

Tablo 12

Madde Güçlük İndeksi ile İlgili Değerler

Madde Güçlük İndeksi	**Değer**
,19 ve altı	Çok zor madde
,20 - ,34 arası	Oldukça zor bir madde
,35 - ,64 arası	Orta düzeyde bir madde
,65 - ,79 arası	Oldukça kolay bir madde
,80 ve üstü	Çok kolay bir madde

Pilot uygulama sonunda öntest ve sontest olarak kullanılan akademik başarı testinde 48 soru, programlamaya hazır bulunuşluluk düzeyi belirleme testinde 32 soru kalmıştır. Testlerde tüm konularda yer alan bütün kazanımları ve bilişsel hedefleri kapsayan sorular bulunmaktadır. Testler son halleri ile kapsam geçerliliğini sağlamaktadır. Akademik başarı

testinde bulunan tüm sorulara ait madde ayırt edicilik ve madde güçlük indeksleri Tablo 13'te verilmiştir.

Tablo 13

Öntest ve Sontest Olarak Kullanılan Akademik Başarı Testindeki Her Bir Maddeye Ait Ayırt Edicilik ve Güçlük İndeksleri

Soru No	Madde Ayırt Edicilik	Madde Güçlük	Soru No	Madde Ayırt Edicilik	Madde Güçlük
1	,70	,57	25	,52	,48
2	,57	,46	26	,39	,50
3	,35	,30	27	,57	,50
4	,65	,50	28	,39	,33
5	,61	,48	29	,43	,78
6	,57	,63	30	,61	,39
7	,39	,63	31	,30	,67
8	,57	,63	32	,78	,43
9	,48	,37	33	,70	,52
10	,61	,52	34	,52	,48
11	,52	,57	35	,48	,33
12	,43	,52	36	,57	,67
13	,39	,33	37	,52	,61
14	,61	,52	38	,43	,39
15	,61	,48	39	,48	,41
16	,83	,59	40	,61	,39
17	,65	,50	41	,57	,50
18	,57	,67	42	,35	,35
19	,35	,30	43	,48	,37
20	,61	,61	44	,52	,39
21	,43	,39	45	,57	,50
22	,52	,65	46	,43	,48
23	,65	,59	47	,52	,43
24	,78	,57	48	,65	,50

Tablo 13 incelendiğinde akademik başarı testi sorularının madde ayırt edicilik indeksi ,30 ile ,83 arasında, madde güçlük indeksi ,30 ile ,78 arasında değişmekte olduğu görülmektedir. Akademik başarı testinin ortalama güçlük indeksi ,50 olarak ölçülmüştür. Akademik başarı testi Ek 1'de, akademik başarı testine ait belirtke tablosu Ek 2'de verilmektedir. Programlamaya hazır bulunuşluk düzeyi belirleme testinde bulunan tüm sorulara ait madde ayırt edicilik ve güçlük indeksleri Tablo 14'te verilmiştir.

Tablo 14

Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testindeki Her Bir Maddeye Ait Ayırt Edicilik ve Güçlük İndeksleri

Soru No	Madde Ayırt Edicilik	Madde Güçlük	Soru No	Madde Ayırt Edicilik	Madde Güçlük
1	,39	,80	17	,52	,70
2	,43	,57	18	,65	,63
3	,35	,83	19	,61	,52
4	,30	,85	20	,61	,52
5	,35	,74	21	,35	,57
6	,48	,63	22	,65	,67
7	,57	,67	23	,43	,61
8	,43	,74	24	,74	,63
9	,78	,43	25	,52	,74
10	,30	,54	26	,65	,50
11	,48	,37	27	,61	,57
12	,52	,70	28	,43	,35
13	,78	,57	29	,30	,46
14	,48	,54	30	,57	,63
15	,61	,65	31	,65	,67
16	,57	,63	32	,35	,70

Tablo 14 incelendiğinde programlamaya hazır bulunuşluk düzeyi belirleme testi sorularının madde ayırt edicilik indeksi ,35 ile ,74 arasında, madde güçlük indeksi ,35 ile ,85 arasında değiştiği görülmektedir. Programlamaya hazır bulunuşluk testinin ortalama güçlük indeksi ,62 olarak ölçülmüştür. Programlamaya hazır bulunuşluk düzeyi belirleme

testi Ek 3'te, programlamaya hazır bulunuşluk düzeyi belirleme testine ait belirtke tablosu Ek 4'te verilmektedir.

Testlerin iç tutarlılık güvenirliğini test etmek amacıyla Kuder Richardson-20 (KR-20) değeri hesaplanmıştır. KR-20 değeri ile testin her bir maddesinin testin tümüyle uyumluluk derecesi saptanmaya çalışılır. Belirlenen güvenirlik katsayısının +1,00'a yakın olması güvenirliğin yüksek olduğunu gösterir (Kuder & Richardson, 1937). Analizler sonucunda akademik başarı testinin KR-20 değeri ,96 olarak, programlamaya hazır bulunuşluk düzeyi belirleme testinin KR-20 değeri ,95 olarak bulunmuştur. Bu bilgiler doğrultusunda geliştirilen testlerin oldukça güvenilir olduğu ve testlerde yer alan soruların birbiriyle tutarlı olduğu söylenebilir.

Problem Çözme Becerisi Algısı Ölçeği

Araştırmada Alice programının öğrencilerin problem çözme becerisi algısı üzerindeki etkisini belirlemek amacıyla; Heppner ve Peterson (1982) tarafından geliştirilen, Şahin vd. (1993) tarafından Türkçeye uyarlanan problem çözeme becerisi algısı ölçeği kullanılmıştır (Ek 5). 35 maddeden oluşan ölçek 1ile 6 arasında puanlanmaktadır ve likert türündedir. Test değerleri aşağıdaki şekildedir:

1. Hep böyle davranırım
2. Çoğunlukla böyle davranırım
3. Sıklıkla böyle davranırım
4. Arada sırada böyle davranırım
5. Ender olarak böyle davranırım
6. Hiç böyle davranmam

Ölçeğin iç tutarlılık katsayısı ,88; güvenilirlik katsayısı ,81 olarak bulunmuştur (Şahin vd., 1993). Ölçek puanlamasında 9., 22. ve 29. maddeler puanlama dışı tutulur. 1., 2., 3., 4.,13., 14., 15., 17., 21., 25., 26., 30. ve 32. maddeler ters olarak puanlanır. Ölçeğin puan aralığı 32-192'dir. Ölçekten alınan toplam puanların yüksekliği, bireyin problem çözme becerisi konusunda kendini yetersiz olarak algıladığını gösterir.

Motivasyon Ölçeği

Alice programının motivasyona etkisini belirlemek amacıyla; Pintrich vd. (1991) tarafından geliştirilen, Büyüköztürk vd. (2004) tarafından GÖSÖ olarak Türkçeye uyarlan MSLQ kullanılmıştır.

Ölçek motivasyon ölçeği ve öğrenme stratejileri ölçeği şeklinde iki ana bölümden meydana gelmektedir. İki ayrı boyuttan oluşan GÖSÖ, modüler bir yapıya sahiptir ve uygulayıcının kullanım amacına göre alt ölçeklerden elde edilecek puanlar ayrı ayrı kullanılabilmektedir (Pintrich, Smith, Garcia & McKeachie, 1993). Çalışma kapsamında motivasyon ölçeği kullanılmıştır (Ek 6).

Motivasyon ölçeği için Cronbach alfa değerlerinin ,86 ile ,59 arasında değiştiği ortaya konulmuştur. Ölçek 31 maddeden oluşmaktadır. Madde soruları 1 ile 7 arasında puanlandırılmaktadır. En düşük puan olan 1 *benim için kesinlikle yanlış*, en yüksek puan olan 7 *benim için kesinlikle doğru* değerlerini taşımaktadır. Aradaki değerler bu iki ifadeye yakınlık düzeyine göre işaretlenmektedir.

Görüşme Formu

Öğrencilerin Alice programıyla tasarlanan öğrenme ortamına ve Alice programına ilişkin görüşlerini belirlemek amacıyla açık uçlu 9 sorudan oluşan yarı yapılandırılmış bir görüşme formu hazırlanmıştır. Görüşme formu hazırlanırken öncelikle alanyazında yer alan görüşme soru tipleri ve görüşme sorularının sahip olması gereken özellikler belirlenmiştir. Ardından alanyazında Alice programı ile ilgili görüşlere yer veren çalışmalar incelenmiştir. Bu çalışmalarda Alice programının genel programlama kavramlarını anlamalarına ne derece yararlı olduğu, Alice programı kullanımının programlama dili öğreniminde ne derece kullanışlı olduğu (Sykes, 2007; Daly, 2013), programlama dili öğreniminde ne derece merak uyandırdığı (Daly, 2013), programın en sevilen ve sevilmeyen özelliklerinin neler olduğu (Howard vd., 2006; Cliburn, 2008), Java dilinin öğrenimini ne derece kolaylaştırdığı, programlama dersinde kullanımına devam edilmesinin ne derece etkili olabileceği (Cliburn, 2008), programın kullanımına devam edilip edilmeyeceği (Sykes, 2007) sorularının öğrencilere yöneltildiği görülmüştür.

Alanyazın incelemesinden sonra görüşme formu hazırlanmıştır. Beş uzman görüşüne sunularak gerekli düzenlemeler yapılmış ve görüşme formuna son hali verilmiştir (Ek 7).

Pilot Çalışma Süreci

Deneysel süreçte yaşanabilecek olası sorunları tespit etmek ve Alice programı kullanılarak tasarlanan öğrenme ortamının olası eksikliklerini belirlemek amacıyla deneysel süreçten önce, beş hafta boyunca pilot çalışma yapılmıştır. Haftada dört saat olmak üzere 20 saat süren bir süreçte pilot uygulama gerçekleştirilmiştir. Pilot çalışma Nevşehir Hacı Bektaş Veli Üniversitesi Meslek Yüksekokulu Bilgisayar Teknolojileri Bölümü normal öğretimde öğrenim gören ve gönüllülük esasına göre seçilen 20 ikinci sınıf öğrencisi ile yürütülmüştür. Öğretim Gagné'nin dokuz aşamalı öğretim modeli göz önünde bulundurularak tasarlanmıştır. Pilot çalışmada kullanılan ders içeriği deney grubunda kullanılan ders içeriğinin ilk beş haftasını kapsamaktadır. Deneysel süreç kapsamında deney grubuna uygulanacak örnek ve uygulamalar kullanılmıştır. Pilot çalışma, iki farklı laboratuarda gerçekleştirilmiştir. İki laboratuarda da yeterli bilgisayar bulunmaktadır ancak bir laboratuar sonradan oluşturulduğu için donanımsal olarak daha yüksek özelliklere sahip bilgisayarlar bulunmaktadır.

Ders sürecinde öncelikle Alice programı ile Java programlama dilinde ilgili haftanın konusu anlatıp örnekler yapılmış, öğrencilerin Alice programı ile uygulamalar yapması istenmiştir. Sonrasında NetBeans programı ile ilgili haftanın konusu anlatılmış ve öğrencilerin uygulamalarını yapmaları sağlanmıştır. İki haftada bir öğrencilerden yansı raporları toplanmış ve araştırmacı alan notları tutmuştur.

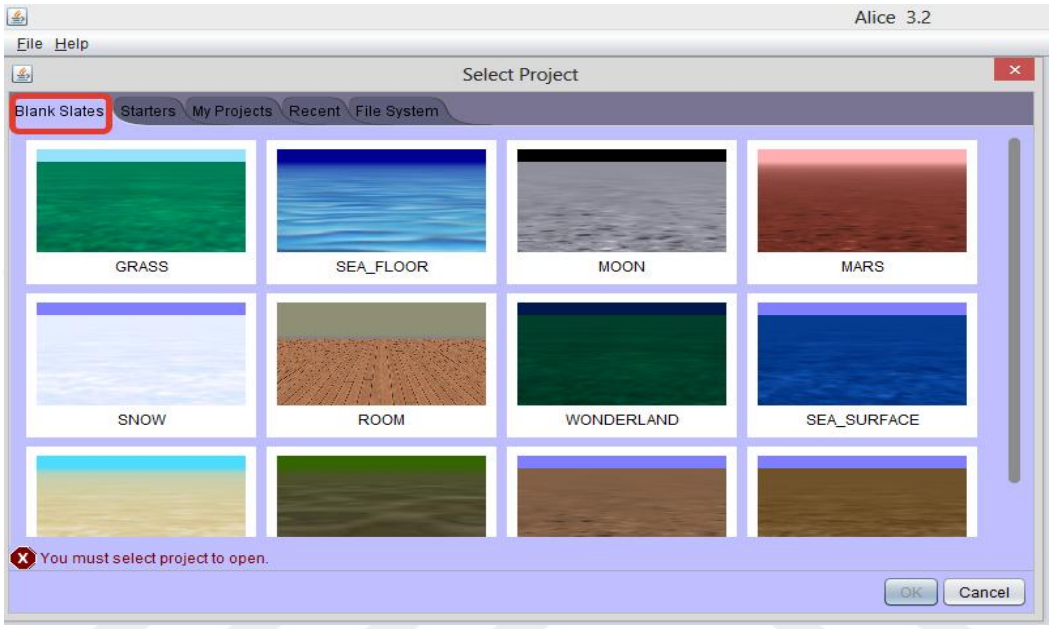
Öğrenci yansı notları incelendiğinde öğrencilerin genel olarak Alice programı ile programlama yapmayı sevdikleri görülmüştür. Çoğunluk, Alice programı ile programlama öğrenimine başlansaydı programlamayı daha çok sevebileceklerini ifade etmiştir. Öğrencilerin birçoğu programlama kavramlarının Alice programı ile daha iyi anlaşıldığını, kod sonuçlarını görerek çalıştıkları için kalıcılığın arttığını belirtmiştir. Bunun yanında Alice programının kullanımını ilk başlarda karmaşık bulduklarını, çalışma ilerledikçe programı daha iyi anladıklarını ifade etmişlerdir. Ayrıca Alice programında çalışırken programda takılma ve kilitlenmelerin çok olduğunu, bu durumun sevilmediğini ifade

etmişlerdir. Takılma ve kilitlenmelerin nasıl olduğu öğrencilere sorulduğunda, yeni laboratuardaki bilgisayarlarda herhangi bir sorunla karşılaşmadıklarını ancak eski laboratuardaki bilgisayarlarda uygulamalarını genellikle bu yüzden bitiremediklerini belirtmişlerdir.

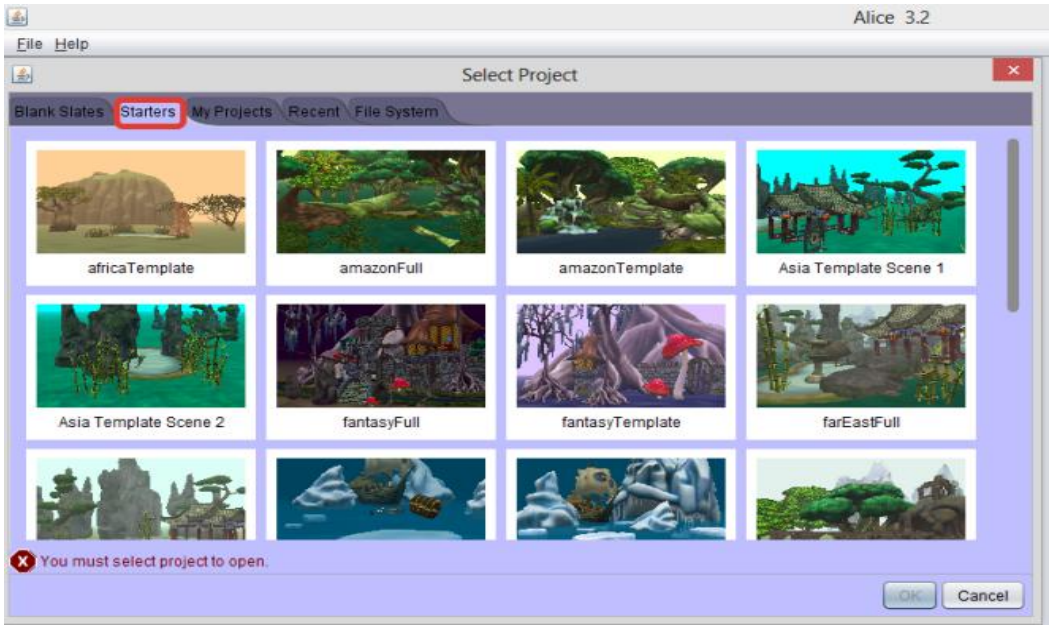
Pilot çalışma sonucunda deneysel süreçte, deney grubu öğrencilerinin Alice programı ile çalışırken yeni laboratuarı kullanmaları sağlanmıştır. Böylece programın yüksek donanımsal özelliklerinden dolayı oluşabilecek sorunların önüne geçilmesi amaçlanmıştır. Deneysel süreç başlangıcında deney grubuna Alice programının tanıtımı yapılmıştır. Bu şekilde Alice programının daha iyi anlaşılması ve süreç içerisinde programı anlamaktan ziyade programlama dili öğrenimine yoğunlaşmaları hedeflenmiştir. Ayrıca pilot çalışma sırasında öğrencilere verilen uygulamalarda nesne hareketlerinin ayrıntılı bir şekilde yaptırılması sonucunda, öğrencilerin programlama dili öğreniminden çok, nesne hareketinin yapılmasına odaklandıkları görülmüştür. Deneysel süreçte deney grubu öğrencilerine verilen Alice örnek ve uygulamalarda nesnelere basit hareketler verilerek programlama öğretilmiştir. Böylece öğrencilerin nesne hareketlerinin yapılmasına değil programlama dili öğrenimine yoğunlaşmaları amaçlanmıştır. Alice programında çalışırken kodlar çalışma ekranına sürüklenip bırakıldığında ya tanımlanan değişkenler girilmesi ya da bir değer girilmesi istenmektedir. Ancak pilot çalışma sürecinde öğrencilerin değişken tanımlamak yerine değer girdikleri görülmüştür. Deneysel süreçte deney grubunda Alice programı ile örnek ve uygulamalar yapılırken, değişkenler tanımlatıldıktan sonra kodlamaya geçiş sağlanmıştır. Bu şekilde öğrencilerin değişken tanımlamanın önemini anlamaları hedeflenmiştir. Pilot çalışma sürecinde döngü kavramları ile Alice programında çalışılırken, döngü yapısının kullanımında öğrencilerin zorlandıkları görülmüştür. Deneysel süreçte döngülerle ilgili Alice programı örnek ve uygulamaları yeniden düzenlenmiştir. Böylece döngü yapısının daha iyi anlaşılması amaçlanmıştır. Pilot çalışma süreci sonunda deneysel sürece ve Alice programı kullanılarak tasarlanan öğrenme ortamına son hali verilmiştir.

Uygulama Süreci

Uygulama süreci 2015-2016 eğitim-öğretim yılı bahar döneminde, Nesne Tabanlı Programlama I dersinde, iki grupta da haftada dört ders saati olmak üzere toplam sekiz hafta sürmüştür. Deney grubuna Alice programı kullanılarak, karşılaştırma grubuna Alice programı kullanılmadan Java programlama dili öğretilmiştir.

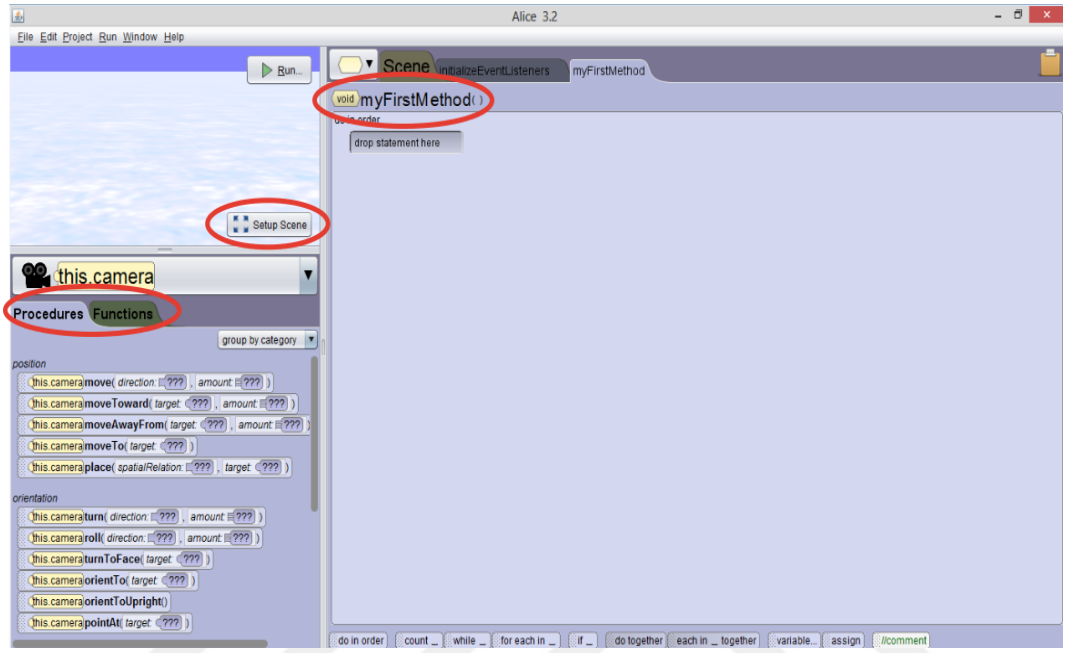


Şekil 7a. Alice programı açılış ekranı



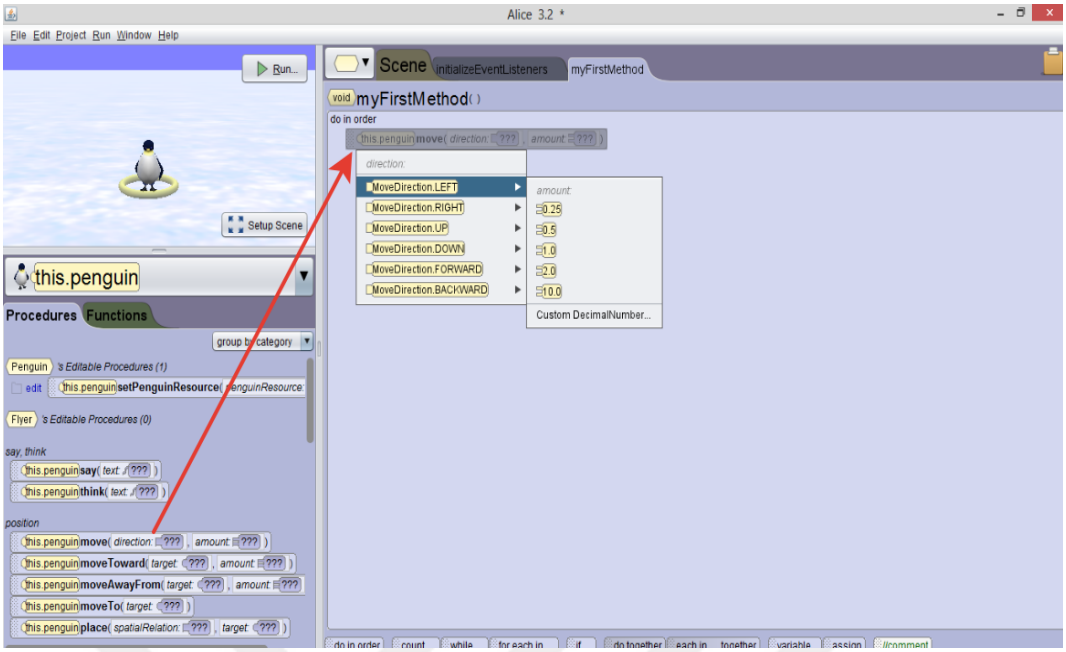
Şekil 7b. Alice programı açılış ekranı

Alice 3.2 programının ilk açılış ekranı Şekil 7a'daki gibidir. Kullanıcıdan ya Şekil 7a'daki gibi düz bir sahne ya da Şekil 7b'deki gibi Alice programında tanımlanmış bir sahne seçmesi istenir. Ayrıca önceden kaydedilen projeler de yine bu ekranda seçilebilir. Kullanıcı bir sahne seçtikten sonra programın ana çalışma sahnesine geçilir.

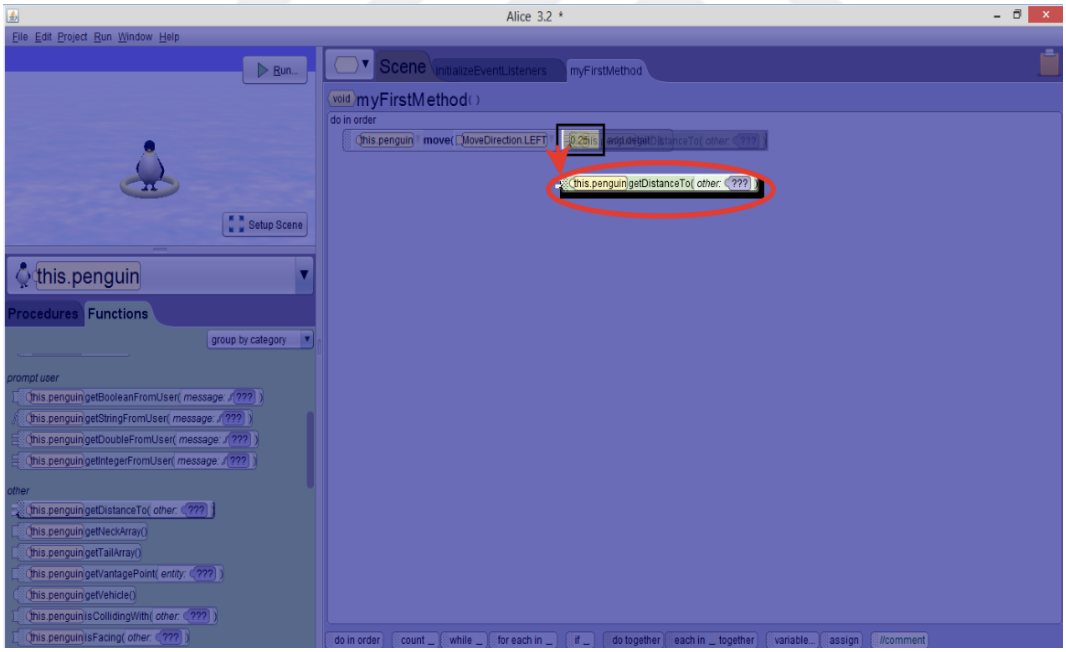


Şekil 8. Alice programı çalışma ekranı

Şekil 8'de Alice programı çalışma ekranı verilmiştir. Sol üst köşedeki ekranda, hazırlanan animasyondaki nesne ve karakterlerin konumları görülür. *Run* düğmesi ile oluşturulan animasyon izlenebilir. *Setup Scene* düğmesine tıklanıldığında sahneye nesne veya karakterler eklenip konumlandırılabilir. Sol alt tarafta *Procedures* ve *Functions* sekmeleri bulunmaktadır. *myFirstMethod* ekranına animasyon ile ilgili kodlar sürüklenip bırakılır. Bu ekranın en altında, kontrol ifadelerinin ve döngülerin bulunduğu alan bulunmaktadır. Animasyonda bir şart veya döngü gerektirecek bir hareket olursa ilgili kod *myFirstMethod* ekranına sürüklenerek bırakılır ve uygun kodlar ile desteklenir.



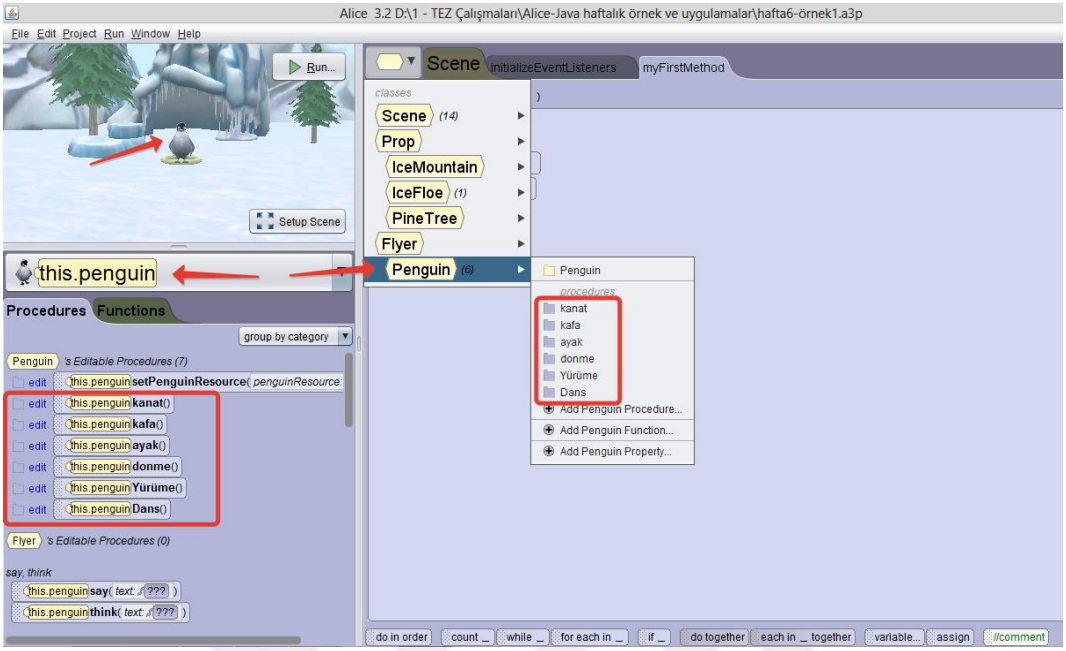
Şekil 9a. Alice programı uygulamasına kod ekleme



Şekil 9b. Alice programı uygulamasına eklenen koda ek özellik verme

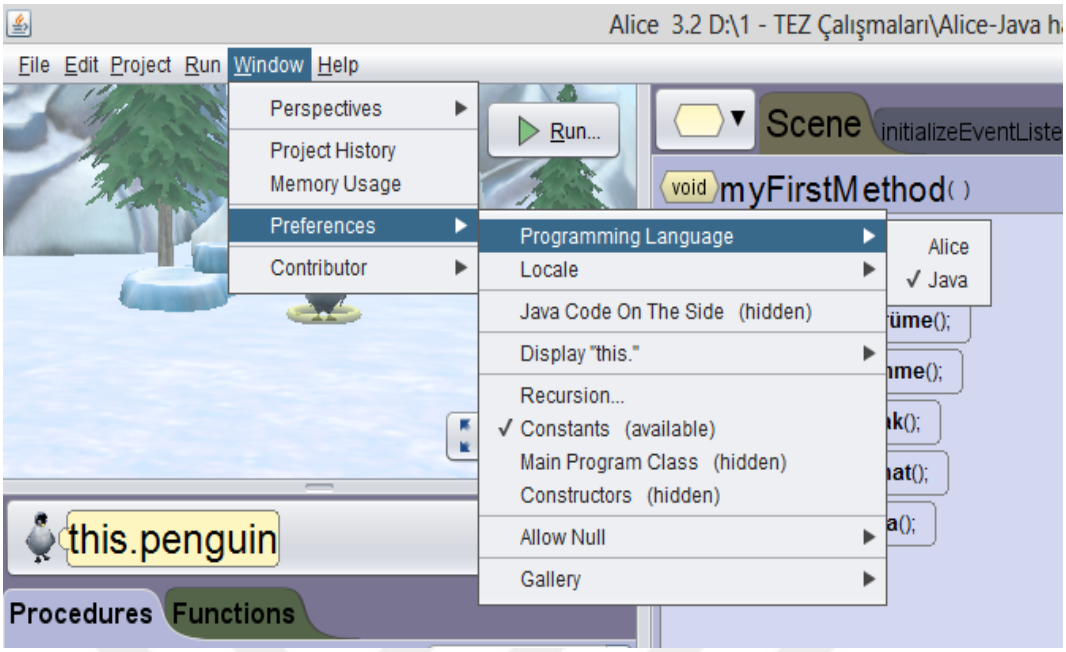
Procedures sekmesi ile seçilen karaktere hangi hareket özelliği verilmek isteniyorsa Şekil 9a'daki gibi kod *myFirstMethod* ekranına sürüklenerek bırakılır. Harekete yön, zaman,

duraklatma vb. özellikler verilebilmesinin yanında Şekil 9b'deki gibi *Functions* sekmesi ile de ek özellikler verilebilir.



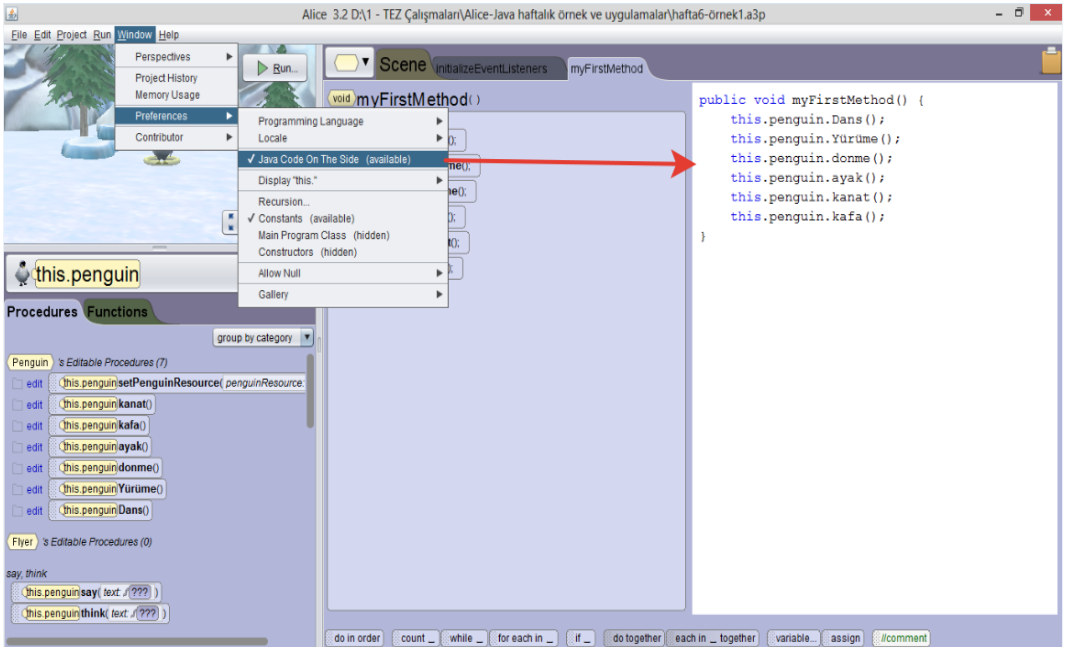
Şekil 10. Alice programında nesne sınıfları

Şekil 10 incelendiğinde Alice programında eklenen her nesnenin bir sınıfı bulunduğu görülmektedir. Nesne sınıfında o nesneye özgü hareketler bir metot olarak kaydedilebilir. Kaydedilen metotlar *Procedures* sekmesinde görülür ve kullanıcı defalarca kullanabilir. Böylece aynı hareket kodlarının tekrar tekrar yazılmasına gerek kalmaz.



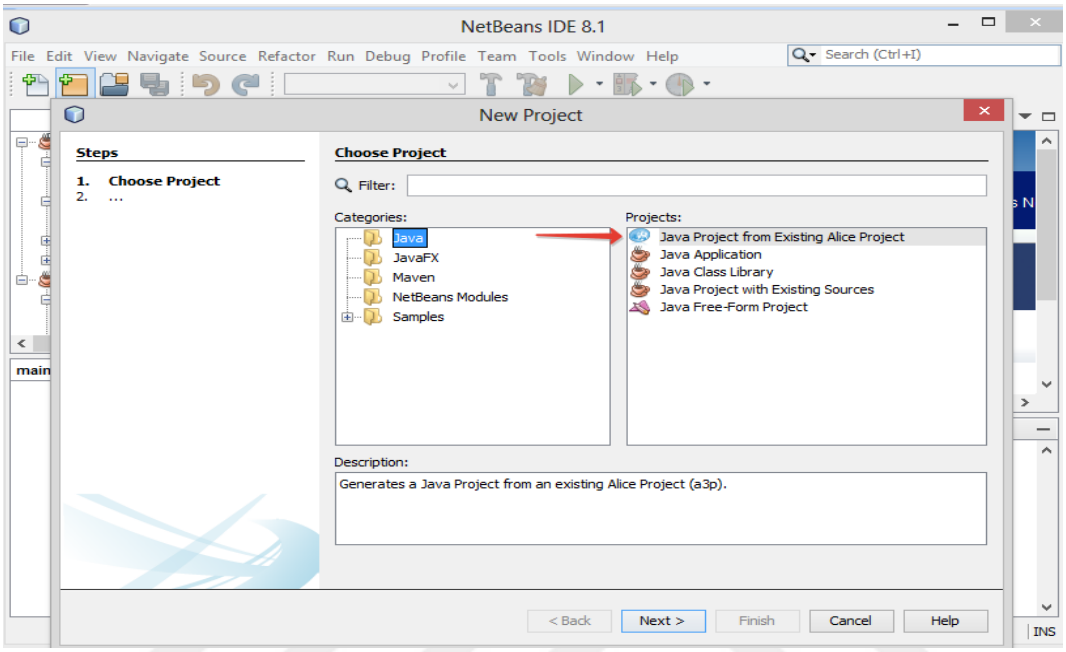
Şekil 11. Alice programında programlama dili seçenekleri

Programda Şekil 11'de görüldüğü gibi, programlama dili seçeneklerinden Java seçilerek eklenen değişken isimleri ve bazı özellikler Java programlama diline uygun isimlendirilmiş şekilde görülebilir.



Şekil 12. Alice programında java kodlarının görülmesi

Alice programında eklenen kodların istenildiği takdirde Java programlama dilinde görünümü sağlanır. Böylece Şekil 12'deki gibi, kullanıcı eklediği kodların Java programlama dili ile eşgüdümlü olduğunu ve Java derleyicisinde nasıl görüneceğini görebilir.



Şekil 13. Alice programı ile NetBeans derleyici programı uyumu

Alice programı, www.alice.org adresinden yüklenen bir eklenti sayesinde NetBeans derleyici programı ile eşgüdümlü çalışabilmektedir. Kullanıcı Alice programı ile geliştirdiği projelerin NetBeans programında nasıl derlendiğini görebilir. Ancak NetBeans programı ile açılan Alice projesine herhangi bir kod eklemesi yapılamaz ya da NetBeans programında oluşturulan Java programları Alice programı ile açılmaz. Şekil 13'te Alice programı ile oluşturulan çalışmaların NetBeans programı ile nasıl açılabildiği görülmektedir.

Uygulama sürecinde; deneysel süreç öncesinde, deneysel süreç içerisinde ve deneysel süreç sonrasında izlenen adımlar Tablo 15'de verilmektedir.

Tablo 15

Uygulama Sürecinde Deney ve Karşılaştırma Gruplarında İzlenen Adımlar

Süreç	**İzlenen Adımlar**	**Konu Anlatımında Deney Grubunda Kullanılan Programlar**	**Konu Anlatımında Karşılaştırma Grubunda Kullanılan Programlar**
Deneysel Süreç Öncesi	• Öntest (akademik başarı testi, problem çözme becerisi algısı ölçeği, motivasyon ölçeği) ölçme araçlarının uygulanması		
Hafta 1	• Hafta 1 konusunun (Veri tipleri ve değişkenler, operatörler) işlenmesi	Alice + NetBeans	NetBeans
Hafta 2	• Hafta 2 konusunun (Kontrol yapıları) işlenmesi	Alice + NetBeans	NetBeans
Hafta 3	• Hafta 3 konusunun (Döngüler) işlenmesi	Alice + NetBeans	NetBeans
Hafta 4	• Hafta 4 konusunun (Diziler) işlenmesi	Alice + NetBeans	NetBeans
Hafta 5	• Hafta 5 konusunun (Metot Kullanımı) işlenmesi	Alice + NetBeans	NetBeans
Hafta 6	• Hafta 6 konusunun (Metot Kullanımı) işlenmesi	Alice + NetBeans	NetBeans
Hafta 7	• Hafta 7 konusunun (Sınıf yapıları, kalıtım ve nesneler) işlenmesi	Alice + NetBeans	NetBeans
Hafta 8	• Hafta 8 konusunun (Sınıf yapıları, kalıtım ve nesneler) işlenmesi	Alice + NetBeans	NetBeans

Deneysel Süreç Sonrası	• Sontest (akademik başarı testi, problem çözme becerisi algısı ölçeği, motivasyon ölçeği) ölçme araçlarının uygulanması • Deney grubu öğrencileriyle odak gruplu görüşmelerin yapılması
Deneysel Süreçten 18 Hafta Sonra	• Programlamaya hazır bulunuşluk düzeyi belirleme testinin uygulanması

Hem deney hem de karşılaştırma grubunda Gagné'nin dokuz aşamalı öğretim modeli göz önünde bulundurularak öğretim tasarlanmıştır. Bilginin, belleğin aşamalarından geçerken bir dizi değişime uğradığı varsayılmaktadır. Bu değişimleri içeren süreçler; dikkat, model alma, geri çağırma, tekrar etme, kodlama vb. işlemlerdir. Bilginin işleme önceliklerini belirleyerek, bilgi akışının değiştirilmesini sağlayan aşama, yönetimsel kontrol süreçleridir. Öğrenmenin bu süreçler etkinleştirildiğinde sağlandığını savunan Gagné'ye (1985) göre; öğretimin amacı bu aktivasyonu kolaylaştırmaktır. Bu sebeple öğrenme sürecini kolaylaştırmak amacıyla öğretim aşamalarını ortaya koymuştur (Driscoll, 1994).

Gagné öğretimin gerçekleştirilebilmesi için dokuz aşamanın dizi halinde izlenmesi gerektiğini düşünse de bu düzenin mutlak olmadığını kabul etmektedir (Gagné & Driscoll, 1988). Tablo 16'da Gagné'nin dokuz aşamalı öğretim modeli basamaklarının destekledikleri iç süreçler verilmiştir.

Tablo 16

Gagné'nin Dokuz Öğretim Aşamasının Desteklediklerı İç Öğrenme Süreçleri ile İlişkilendirilmesi

İç Süreç	**Öğretim Aşaması**	**Eylem**
Alma	1. Dikkat çekme	Ani uyarıcı değişimi kullanılmalı
Beklenti	2. Öğrenciyi hedeften haberdar etme	Öğrendikten sonra neler yapabilecekleri öğrencilere söylenmeli
Çalışma belleğine geri çağırma	3. Önceki öğrenmelerle ilişkilendirme	Önceden öğrenilen bilgi veya becerilerin geri çağrılması istenmeli
Algıda seçicilik	4. Uyarıcıyı sunma	İçerik ayırt edici özelliklerle gösterilmeli
Semantik kodlama	5. Öğrenme rehberliği yapma	Anlamlı bir organizasyon önerilmeli
Tepki verme	6. Performansı ortaya çıkarma	Öğrenciden sorması istenmeli
Geri alma	7. Geribildirim sağlama	Bilgilendirici geri bildirim verilmeli
Geri alma ve takviye	8. Performansı değerlendirme	Geribildirim ile ek öğrenci performansı oluşturulmalı
Geri alma ve genelleştirme	9.Öğrenilenlerin kalıcılığını sağlama ve transferini güçlendirme	Değişken uygulama ve aralıklı değerlendirme sağlanmalı

"Gagne's Theory of Instruction" Driscoll, M., 1994, Psychology of Learning for Instruction, Boston MA, Allyn and Bacon, 329-358 kaynağından uyarlanmıştır.

Gagné'nin dokuz aşamalı öğretim modeli, öğrenmenin bütün şartlarını karşılamakta ve modelin odak noktasını bilişsel beceriler oluşturmaktadır (Driscoll, 1994). Gagné'nin dokuz aşamalı öğretim modelinde, basit seviyedeki yetenekler üst seviyedeki yeteneklere alt yapı oluşturmaktadır (Gagné, 1985). Programlama derslerinde önemli olan problemin

çözüme ulaştırılarak bir programın geliştirilmesidir. Programlama dili öğrenme, basit beceriler ile karmaşık yetenekleri ortaya çıkarma sürecidir. Programlama üst düzey becerileri gerektirmekte ve bilişsel süreçler ön plana çıkmaktadır. Bu nedenle araştırmada ilgili modelin kullanımına karar verilmiştir. Modelin haftalarda ders anlatımı sırasında uygulanırken yapılan işlemler Tablo 17'de verilmektedir.

Tablo 17

Gagné'nin Dokuz Aşamalı Öğretim Modelinin Uygulanmasında Yapılan İşlemler

Model Adımları	**Yapılan İşlenmeler**
Dikkat çekme	Haftanın konusunu içeren program çıktılarının gösterilmesi
	Çıktılarla ilgili öğrencilere merak uyandıracak sorular sorulması
Öğrenenleri hedeften haberdar etme	Haftanın konusunu içeren program çıktılarının nasıl yapılabileceği konusunda bilgilendirme yapılması
	Haftanın kazanımları ile ilgili bilgilendirme yapılması
Önceki öğrenmelerle ilişkilendirme	Öğrencilerin öğrendikleri bilgileri sonraki haftalarda da kullanabilecekleri örnekler gösterilmesi
	Uygulamalar yaptırılarak öğrenmelerin desteklenmesi
	Hangi işlem adımlarının izlenmesi gerektiğini hatırlatan soruların yöneltilmesi
	İhtiyaç duyulduğunda eski örneklerin gösterilmesi / hatırlatıcı ek örneklerin yapılması
İçeriğin Sunumu	Haftaya ait konu içeriklerinin ayrıntılı şekilde anlatılması
	Bilgisayarda içeriklerin kullanımlarının farklı örneklerle gösterilmesi
	Öğrencilere farklı problem durumları içeren uygulamalar verilmesi
	Konunun önemli noktalarına, benzerlik ve farklılıklarına ders boyunca vurgu yapılması
	Gerekli görüldüğü takdirde tahtada ya da

	bilgisayar üzerinde açıklamalar yapılması
Öğrenme rehberliği yapma	Öğrenciler uygulamalarını ve örneklerini yaparken araştırmacının laboratuarda gezerek öğrencilerin yaptıklarını incelemesi
	Problem çözüm esnasında gerekli yerlerde doğru çözüme ulaştıracak ipuçları verilmesi
	Öğrencilerin farklı açılardan düşünmelerini sağlayacak bilgiler ve örnekler verilmesi
Performansı ortaya çıkarma	Haftanın konusu ile ilgili uygulamalar verilmesi
	Uygulamaları çözmeleri için öğrencilere süre verilmesi
	Öğrencilerin ihtiyaç duydukları anlarda destek olunarak problem çözüm esnasında gerekli yerlerde rehberlik sağlanması
	Süre sonunda uygulamaların çözümlerinin yapılması ve çözemeyen öğrencilerin de çözüme ulaşmasının sağlanması
Geri bildirim sağlama	Öğrenciler uygulamalarını yaparken performansları ve gidiş yolları ile ilgili geri bildirim verilmesi
	Eksik yönlerini nasıl tamamlayacakları, yanlışlarını nasıl düzeltecekleri konusunda bilgilendirme sağlanması
	Problem çözümlerinde öğrencilerin yaratıcı ve farklı çözüm yollarının doğrulanması ve diğer öğrencilerin de görmesinin sağlanması
	Farklı çözüm yolları ile ilgili problemlerin nasıl çözülebileceği konusunda açıklama yapılması
Performansın değerlendirilmesi	Öğrencilerden haftanın konusuyla ilgili örnek ve uygulamalar yapmaları istenmesi
	Akademik başarı sınavının uygulanması

Kalıcılığı sağlama ve bilgi transferini güçlendirme	Öğrencilerin keşfederek öğrenmelerinin desteklenmesi
	Öğrencilere uygulamalar sırasında karşılaştıkları zorluklarda ne yapmaları gerektiğinin söylenmemesi, ipuçları verilerek çözüme kendilerinin gitmelerinin teşvik edilmesi
	Geri bildirimler sağlanarak çözüm yolları ile ilgili bilgi sahibi olmalarının sağlanması
	Gerekli duyulan yerlerde ek örnekler verilerek farklı çözüm yollarıyla ilişki kurmalarının sağlanması

Modeldeki her süreç ve araştırma kapsamında süreçlerde uygulanan adımlar şu şekildedir :

1. Dikkat çekme :

Öğrenme; öğrenci bir şekilde yönlendirilmediği ve gelen bilgiye açık olmadığı sürece gerçekleşemeyeceğinden, dikkat çekme öğretimde gerçekleşmesi gereken ilk aşamadır. Dikkat çekme, öğrencilerin görevden uzaklaşmış gibi göründüğü durumlarda öğrencilerin dikkatini geri kazanmak için ders boyunca çeşitli biçimlerde tekrarlanabilecek bir çeşit uyaran değişikliği ile sağlanır. Bunlar arasında öğretmenin belirli öğrencilerin adlarını söylemesi, "Herkes dinlesin" gibi sözlü işaretler kullanması bulunmaktadır (Driscoll, 1994). Ayrıca öğrencilerin dikkatini çekmek amacıyla sorular sorulabilir ya da örnek durumlarla öğrencilerde merak uyandırılabilir (Reigeluth, 1987; Dick, Carey & Carey, 2005). Bilgisayarlı öğretimde dikkat çekme, ekranda yanıp sönen sinyaller, "Ekranda bir mesaj ara" anlamına gelecek uyarı sesleri, farklı uygulamaların gösterimleri biçiminde karşımıza çıkmaktadır (Driscoll, 1994).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada öğrencilerin dikkatini ilgili haftanın problemine/görevlerine çekmek amacıyla dersin başında deney grubuna her hafta o haftanın konusunu içeren Alice programında yapılmış uygulamaların animasyonları, karşılaştırma grubuna ise o haftanın konusu ile ilgili Java uygulamalarının çıktıları gösterilmiştir. Daha sonra bu animasyon ve çıktıları verecek programların nasıl yazılacağı, hangi komutların kullanılması gerektiği, oluşturulacak programların algoritmalarının nasıl olması gerektiği gibi sorular sorularak öğrencilerin dikkati öğrenmeye çekilmiştir.

2. Öğrenenleri hedeften haberdar etme:

Öğrencileri hedeften haberdar etmek; öğrencilerde öğrenecekleri bilginin, daha sonraki öğrenmelerini etkileyeceğine dair bir beklenti içinde olmasını sağlar. Örneğin öğrenciler, belirli bir bilginin farkında olduklarında ve bu bilgiyi öğrenmeye hazır olduklarında, bu hedefe ilişkin herhangi bir uyarana karşı daha dikkatli olacaklardır (Driscoll, 1994). Beklentiler öğretim amaçlarının basit ifadeleriyle, öğrencilerin öğretimden sonra yapabileceklerine ilişkin referanslar almasıyla veya beklenen öğrenme sonuçlarının gösterilmesiyle kolayca oluşturulabilir. Öğretmen ya da eğitim materyalinin öğrenme hedefleri konusunda açık olmadığı ya da birbirleriyle çeliştiği durumlarda, öğrenciler muhtemelen sınıfta olanları ya da sınavlarda gördüklerini örnek alacaktır (Driscoll, 1994). Öğrencilerin öğretim sonunda beklenen hedefler ile ilgili bilgilendirilmesi; hangi bilginin yararlı olduğu ve nasıl bir çalışma stratejisi izlemeleri gerektiği konularında düşünmelerini sağlamakta, beklenti ve motivasyonlarını arttırmaya yardımcı olmaktadır (Gagné, 1985; Dick vd., 2005).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada, her derste dikkat çekme aşamasında gösterilen program çıktılarının nasıl yapılabileceği konusunda öğrenciler bilgilendirilmiştir. İlgili derste kazanacakları bilgi ve beceriler sayesinde gösterilen program çıktılarına ulaşabilecek programları geliştirebilecekleri belirtilmiştir. Öğrencilere ilgili haftanın kazanımları gösterilmiş ve neler öğrenecekleri konusunda bilgilendirme yapılmıştır. Böylece öğrenciler hedeflerden haberdar edilmiştir.

3. Önceki öğrenmelerle ilişkilendirme:

Yeni bir bilgi öğrenilirken eski bilgiler hatırlatılmalı ve yeni bilgi ile eski bilgi arasında ilişki kurulmalıdır. Öğrenci eski bilgiyi hatırlaması ve ilişki kurması için yönlendirilmelidir. Öğrenci yeterli deneyim sağladıktan sonra eski bilgiler yeni bilginin temelini oluşturur (Merrill, 2002). Her ne kadar yeni öğrenme büyük oranda daha önce öğrenilene bağlı olsa da öğrenciler yeni bir öğrenme görevi ile karşılaştıklarında her zaman ilgili bir bilgiyi hatırlayıp kullanmazlar. Bilginin transferi, yani daha önce öğrenilen bir bilginin yeni bir probleme ya da yeni bir bağlamda uygulanması zor olarak ifade edilmektedir (Gagné, 1985). Bu yüzden öğrencileri kodlamaya ya da transfere hazır hale

getirmek için, ilişkili ve önkoşul bilgileri hatırlamaları ve yeni yapılarını oluşturmaları amacıyla yönlendirme yapılmalıdır (Driscoll, 1994).

Geçmiş öğrenmenin hatırlanmasını teşvik etmek, öğrencilere bir önceki gün ya da geçen hafta sınıfta öğrendiklerini hatırlatmak kadar basit olabilir. Ancak bazı durumlarda, basit hatırlatıcılar yeterli değildir. Bu durumda önceden edinilen bilgi ya da becerilerin çeşitli pratik etkinlikler ile yeniden verilmesi gerekir (Gagne & Driscoll, 1988). İlgili bilgiye sahip olduklarının farkında oldukları zaman bile öğrencilerin geçmiş bilgilerini yeni durumlara aktarması için kayda değer bir çaba harcamaları gerekir (Salomon & Perkins, 1989). Ayrıca, öğrenciler kendileri çözmekten ziyade cevabı başkalarına sormayı daha kolay buluyor olabilir. Problem çözme sürecinin eğitimin önemli bir hedefi olduğu durumlarda, öğrencilerin vazgeçmeme konusundaki ısrarlarını teşvik edecek şekilde yönlendirilme yapılmalıdır (Driscoll, 1994).

Bu bilgiler doğrultusunda uygulama süreci boyunca; her iki gruptaki öğrencilere, öğrendikleri bilgileri sonraki haftalarda da kullanabilecekleri örnekler gösterilmiştir. Uygulamalar yaptırılarak öğrenmeleri desteklenmiştir. Bu aşamada gerek geliştirilen programların çözüm adımlarında algoritma oluşturulurken, gerekse program yazılırken; hangi kodları kullanmaları gerektiğini hatırlamalarını sağlayan sorular öğrencilere yöneltilmiştir. İhtiyaç halinde eski örnekler gösterilmiş ya da hatırlatıcı ek örnekler yapılmıştır.

4. İçeriğin sunumu:

Öğretimin bu aşamasında öğretilecek bilgiye göre içeriğin sunumu değişmektedir. Driscoll (1994)'a göre öğretimden:

- Bilgi edinmek amaçlanmışsa uyaran; ders kitabının bir bölümü, ders anlatımları ya da içeriğin bulunduğu bir videodan oluşabilir.
- Entelektüel beceri öğrenimi amaçlanmışsa en etkili uyaran, öğrenilecek kavram ya da kuralın ayırt edici özelliklerini belirgin bir şekilde sergilenmesi ve beceri kurallarının açık bir şekilde gösterilme olabilir.
- Motor beceri ya da bilişsel strateji öğrenimi amaçlanmışsa uyaranların sunulması, istenen sonucun gösterilmesinden veya sözlü talimatlar verilmesinden oluşabilir.

- Tutum öğrenimi amaçlanmışsa uyaran, istenen eylem ya da tercihin genellikle bir model tarafından gösterilmesi olabilir.

Aslında her türden sonuç için uyaran sunumu, algıda seçicilik süreçlerini kolaylaştırmak amacıyla istenen sonucun ayırt edici özelliklerini veya temel öğelerini vurgulamayı içerir (Driscoll, 1994).

Programlama becerileri entelektüel ile bilişsel beceriler olarak ifade edilebilir ve bu bağlamda öğretilecek kavramların farklı özellikleri açıklanmalı, farklı örnekler sunularak sonuçlar gösterilmeli ve farklı öğretim stratejileri kullanılmalıdır (Reigeluth, 1987). Öğrenme amaçları ile uyumlu olacak şekilde farklı sunum yöntemleri kullanılmalı ve bu sunumların karşılaştırılması sağlanmalıdır (Merrill, 2002).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada, deney ve karşılaştırma gruplarında, haftaya ait konu içeriklerinin (komutlar, döngüler, semboller, işaretler vb.) kullanımları tahtaya yazılarak ayrıntılı şekilde anlatılmıştır. Daha sonra bilgisayarda içeriklerin kullanımları farklı örneklerle gösterilmiştir. Öğrencilere farklı problem durumları içeren uygulamalar verilmiş, çözümleri incelenmiştir. Konunun önemli noktalarına, benzerlik ve farklılıklarına ders boyunca vurgu yapılmış, gerekli görüldüğü takdirde tahtada ya da bilgisayar üzerinde açıklamalar yapılmıştır.

5. Öğrenme rehberliği yapma:

Öğrenene rehberlik etmek uyarıcıları mümkün olduğunca anlamlı hale getirmektedir. Öğretimde nasıl ya da hangi öğrenme rehberliğinin yapılacağı, istenen sonuca bağlıdır. Bu süreçte kolaylaştırılacak öncelikli süreç, semantik kodlamadır ve özellikle öğretim faaliyetleri, öğrenilecek bilginin anlamlı bir şekilde uzun süreli belleğe alınmasını teşvik etmelidir (Gagné, 1985). Bu noktada öğretmen ya da öğretimi tasarlayan kişi, kritik ve her öğrenme sonucuna özel olan öğrenme koşullarına atıfta bulunmalıdır (Driscoll, 1994). Soyut terimler ve kavramlar için somut örnekler kullanılmalı, her bilgi öğrencilerin bildiği diğer bilgilerle ilişkilendirilerek ayrıntılandırılmalıdır (Reigeluth, 1987).

Öğrenme rehberliğinin ne kadar verileceği; öğrencilerin kabiliyeti ve bilgi düzeyine, öğretim için ayrılan zamana ve birden fazla öğrenme hedefinin varlığına bağlıdır. Çok kabiliyetli ve bilgili öğrenciler, daha az kabiliyetli öğrencilerden muhtemelen daha az

rehberliğe ihtiyaç duyar (Driscoll, 1994). Süreç devam ettikçe rehberlik giderek azaltılmalıdır (Merrill, 2002).

Driscoll (1994)'a göre rehberlik yapılırken; keşfederek öğrenme vurgulanmalı, ipuçları verilmeli, ancak öğrencilerden ne yapacakları söylenmeden kendilerinin çözmeleri beklenmelidir. Keşfederek öğrenme aynı zamanda çok zaman kaybına yol açabileceğinden, öğretimi veren kişi keşfederek öğrenmenin avantajlarını (örneğin uzun süreli bellekte tutmayı sağlaması ya da bilgiyi aktarmayı kolaylaştırması) ile dezavantajlarını (örneğin kapsamlı kaynak ve zaman gerektirmesi) dikkate alarak öğretimi planlamalıdır (Driscoll, 1994).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada, deney ve karşılaştırma grubunda yer alan öğrenciler uygulamalarını ve örneklerini yaparken araştırmacı laboratuarda gezmiş ve öğrencilerin yaptıklarını incelemiştir. Problem çözümü esnasında gerekli yerlerde doğru çözüme ulaştıracak ipuçları vermiştir. Farklı açılardan düşünmelerini sağlayacak bilgiler ve örnekler vererek rehberlik sağlamıştır.

6. *Performansı ortaya çıkarma:*

Bir ve beş arasındaki öğretim aşamalarının; öğrenmenin gerçekleşmesini, öğrenilecek bilginin yeterli bir şekilde kodlanmasını ve bilginin uzun süreli bellekte saklanmasını güvence altına aldığı varsayılmaktadır (Gagné, 1985). Bu aşama ise öğrencilerin öğrendiklerini; kendilerine, öğretmenlerine ve diğerlerine doğrulamasını sağlamaktadır. Öğrencinin bir performans, öğrenilen bilginin uygun bir göstergesi olan bir sonuç üretmesi beklenir (Driscoll, 1994). Öğrenme, yeni bilginin öğrenci tarafından uygulanması ve gerçek bir ortamda kullanılması ile daha iyi gerçekleşmektedir (Reigeluth, 1987; Merrill, 2002). Öğrenenden yeni bilgi ve becerilerini göstermesi, yorumlaması, analiz yapması, kendi çözüm yollarını oluşturması, yeni bilgisini farklı durumlarda kullanması beklenir (Merrill, 2002).

Performansın ortaya çıkarılmasındaki amaç, öğrencilerin ceza olmadan öğrendiklerini sergilemesidir. Başka bir deyişle, bu etkinlik hataların düzeltilmeye devam ettiği ve performansın hala iyileştirilmeye çalışıldığı varsayımıyla ilerlemeyi ölçmek için bir fırsat sağlamaktadır (Driscoll, 1994).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada, deney ve karşılaştırma grubunda yer alan öğrencilerden; haftanın konusu ile ilgili uygulamalar yapmaları istenmiştir. Uygulamaları çözmeleri için öğrencilere zaman verilmiştir. Süreç içerisinde araştırmacı laboratuarda gezmiş ve öğrencilerin ihtiyaç duyduğu durumlarda destek olarak problem çözüm esnasında gerekli yerlerde rehberlik sağlamıştır. Süre sonunda uygulamaların çözümleri yapılmış veya gerekli yerlerde bilgisayardan gösterilmiş böylece çözemeyen öğrencilerin de çözüme ulaşması sağlanmıştır.

7. Geri bildirim sağlama:

Ne yapabildiklerini gösteren öğrencilere performanslarına ilişkin bilgilendirici geri bildirim sağlanmalıdır. Geri bildirim ile, farklı cevaplar gerektiren bilgiler ve beceriler için öğrencilere cevaplarının doğru olup olmadığı bilgisi dolaylı yoldan gösterilir. Doğru değilse, geri bildirim öğrencilerin hatalarını tespit etmesi ve düzeltmesine yardımcı olmalıdır (Driscoll, 1994).

Öğrenilecek tüm materyaller doğru ve yanlış cevaplardan oluşmamaktadır. Motor beceriler; doğru ancak tecrübesizce ya da sakarca yapılabilir, bu durumda geri bildirim öğrencilere mevcut becerilerini nasıl iyileştireceklerini göstermeyi amaçlamalıdır (Driscoll, 1994). Bilişsel strateji öğrenme için geri bildirim, öğrencilere performanslarının nasıl daha stratejik ya da daha yaratıcı olabilecekleri konusunda bilgi verebilir (Driscoll, 1994). Böylece, öğrenciye doğru davranışlarını pekiştirme, yanlışları ise zaman geçmeden düzeltme imkânı verilmiş olunur (Merrill, 2002).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada, deney ve karşılaştırma grubunda yer alan öğrencilere, haftanın konusuyla ilgili problem durumlarını içeren uygulamalarını yaparken performansları ve gidiş yolları ile ilgili geri bildirim verilmiştir. Eksik yönlerini nasıl tamamlayacakları, yanlışlarını nasıl düzeltecekleri konusunda bilgilendirme sağlanmıştır. Problem çözümlerinde yaratıcı ve farklı çözüm yolları doğrulanmış, diğer öğrencilerin de çözümleri görmeleri sağlanmıştır. Ayrıca farklı çözüm yolları ile ilgili problemlerin nasıl çözülebileceği konusunda açıklama yapılmıştır. Böylece öğrencilerin durumlarıyla ilgili bilgilendirilmesi sağlanmıştır.

8. Performansın değerlendirilmesi:

Öğrenme; zaman içerisinde devam eden davranış ya da performans değişikliğidir. Dolayısıyla, öğrenciler bilgilerini gösterme ve işleme fırsatı bulduktan sonra resmi olarak değerlendirebilir (Driscoll, 1994). Entelektüel beceriler değerlendirilirken bilgilerin birkaç yeni örneğe uygulanması, bilişsel stratejiler değerlendirilirken orijinal ve çeşitlilik gösteren problemlerin çözülmesi sağlanmalıdır (Reigeluth, 1987). Değerlendirme aşaması tipik olarak ünite ya da bölüm testleri, projeler, portföyler, gösterimler vb. ile gerçekleştirilir. Aynı zamanda öğrenci notları da verilebilir (Driscoll, 1994). Her ne kadar bu etkinlik derste çok geç gerçekleşse bile, Gagné ve Driscoll (1988) her doğru performansa uygun dönüt verilmesini önermiştir.

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada ders içerisinde; deney grubunda yer alan öğrencilerden Alice programı ve Java programlama dili ile, karşılaştırma grubunda yer alan öğrencilerden Java programlama dili ile haftanın konusuyla ilgili örnek ve uygulamalar yapmaları istenmiştir. Ayrıca her iki gruba akademik başarı sınavı uygulanmıştır. Böylece öğrenci performansları ölçülmeye çalışılmıştır.

9. Kalıcılığı sağlama ve bilgi transferini güçlendirme:

Yeni öğrenilen bilgilerin kalıcı olması ve kolay hatırlanabilmesi için, bilgilerin uzun süreli bellekte iyi bir biçimde örgütlenmesi ve belli aralıklarla tekrar edilmesi gerekir (Gagné, 1985). Tekrar, öğrenme sırasında yoğun ya da aralıklı olarak yapılabilir. Bilgilerin bellekte iyi örgütlenmesi için de, öğrenilen bilgilerin yeni durumlarda kullanılması sağlanmalıdır (Driscoll, 1994).

Her ne kadar öğretimin bu basamağı son aşama olsa da, öğrenilenlerin kalıcılığını sağlamak ve bilginin transferini güçlendirmek için eğitim faaliyetleri genellikle öğretimin daha erken bir evresine dâhil edilmiştir (Gagné, 1985). Örneğin, Driscoll (1994); öğrencilerin entelektüel becerileri uygun şekilde transfer edebilmesi için, kritik öğrenme koşullarının bulunduğunu ve bu öğrenme koşullarını öğrencilere aktarma işleminin, genellikle öğretimde öğrenme rehberliği yapma sırasında gerçekleştirildiğini ifade etmektedir. Benzer şekilde, öğretimde aralıklı incelemeler entelektüel ve motor becerilerin kalıcı olmasını kolaylaştırır. Bu amaçla performansı ortaya çıkarma ve geri bildirim sağlama sırasında birkaç defa tekrar yapılması planlanabilir (Driscoll, 1994).

Reigeluth (1987)'a göre öğrenilenlerin kalıcılığını sağlamak ve bilginin transferini güçlendirmek amacıyla öğretim sırasında öğrencilere, yeni kazandıkları bilgileri uygulayabilecekleri problem durumları verilmelidir. Öğrencilerin olgu ve olayları yorumlayabilmesi, konulara eleştirel bir gözden bakabilmesi, karşılaştırma yapabilmesi ancak bu yolla mümkündür (Reigeluth, 1987).

Bu bilgiler doğrultusunda uygulama sürecinde bu aşamada; deney ve karşılaştırma grubunda yer alan öğrencilerin ders içi yaptıkları uygulamalar sırasında özellikle keşfederek öğrenmeleri desteklenmiştir. Bu bağlamda öğrenciler uygulamalarını yaparken karşılaştıkları zorluklarda ne yapmaları gerektiği söylenmemiş ipuçları verilerek çözüme kendilerinin gitmeleri teşvik edilmiştir. Geri bildirimler sağlanarak çözüm yollarıyla ilgili bilgi sahibi olmaları sağlanmıştır. Gerekli duyulan yerlerde ek örnekler verilerek farklı çözüm yollarıyla ilişki kurmaları desteklenmiştir.

Model hem deney hem de karşılaştırma gurubundaki öğrencilere uygulanmıştır. Uygulanan sürecinde deney ve karşılaştırma gruplarındaki farklılık şu şekildedir :

Deney grubunda dersin ilk yarısında, her haftanın konusu öncelikle Alice programı üzerinden öğretim elemanı tarafından anlatılıp gösterilmiş ve örnekler yapılmıştır. Sonrasında Alice programı ile öğrencilerin uygulamalar yapması istenmiştir. Dersin diğer yarısında önce öğretim elemanı öğrenilen kavramların Java programlama dilinde kullanımıyla ilgili örnekler yapmış, sonrasında öğrencilerin uygulamalar yapması sağlanmıştır. Alice programındaki örnekler Java programlama dili gösterim esnasında hatırlatılmış ve öğrencilerin örnekler arasında bağlantı kurmaları sağlanmıştır.

Karşılaştırma grubunda Alice programı kullanılmamış, ders Java programlama dili üzerinden işlenmiştir. Derste öncelikle haftanın konusu öğretim elemanı tarafından Java programlama dili üzerinden anlatılıp gösterilmiş ve örnekler yapılmıştır. Sonrasında öğrencilerin konu ile ilgili uygulamalar yapması istenmiştir.

Deney grubundaki Java programlama dili örnek ve uygulamalarının aynısı karşılaştırma grubunda da kullanılmıştır. Karşılaştırma grubunda, deney grubundaki Alice örnek ve uygulamalarına karşılık gelecek Java programlama dili örnek ve uygulamaları kullanılmıştır. Alice örnek ve uygulamaları ile Java programlama dili örnek ve uygulamalarının denk olmasını sağlamak amacıyla, uygulama süreci öncesinde, iki grupta

kullanılan ders içerikleri ile örnekler ve uygulamalar üç alan uzmanı tarafından incelenmiştir. Ders içeriklerini, örnek ve uygulamaları inceleyen alan uzmanları; dağıtılan uzman değerlendirme formunu (Ek 10) doldurarak görüşlerini belirtmiştir. Alan uzmanları gruplar arasındaki karşılaştırmaları yapmış ve görüşleri doğrultusunda ders içeriklerine son hali verilmiştir (Ek 8-9). Böylece ders anlatımı sırasında Alice örnek ve uygulamaları ile Java örnek ve uygulamalarının birbirine denk olması sağlanmış, herhangi bir grubun diğer gruba göre daha fazla iş yükünün olması engellenmiştir.

Veri Analizi

Araştırma hem nicel hem de nitel verileri kapsamaktadır. Bu bölümde elde edilen veri türlerine göre analiz yöntemleri açıklanmıştır.

Çalışma grubu öğrencilerine ait demografik bilgilerin ve nicel verilerin betimlenmesinde frekans, yüzde, ortalama, standart sapma vb. istatistikler kullanılmıştır. Araştırmada nicel veriler kapsamında problem çözme becerisi algısı ile motivasyon ölçeklerinden ve akademik başarı ile programlamaya hazır bulunuşluk düzeyi belirleme testlerinden elde edilen veriler analiz edilmiştir. Nicel verilerin analizi yapılırken parametrik ya da parametrik olmayan bir analiz yönteminin karar verilmesi amacıyla, deney ve karşılaştırma gruplarından elde edilen verilerin dağılımları incelenmiştir. Öncelikle verilerin normal dağılım gösterip göstermediğine, ardından varyansların homojenlik durumuna bakılmıştır. Verilerin normal dağılım gösterip göstermediğini belirlemek amacıyla K-S ve Shapiro-Wilk testleri, varyansların homojenlik durumunu belirlemek amacıyla Levene testi kullanılmıştır. Öntest araştırma verilerinin dağılım değerleri sonuçları Tablo 18'de, sontest araştırma verilerinin dağılım değerleri sonuçları Tablo 19'da, programlamaya hazır bulunuşluk düzeyi belirleme testinin dağılım değerleri sonuçları Tablo 20'de verilmektedir.

Tablo 18

Öntest Araştırma Verilerinin Dağılım Değerleri

Test	Grup	n	Problem Çözme Becerisi Algısı (p değeri)	Motivasyon (p değeri)	Akademik Başarı (p değeri)
K-S	**Deney**	34	,08	,20	,06
	Karşılaştırma	29	,20	,20	,06
Shapiro-Wilk	**Deney**	34	,12	,50	,13
	Karşılaştırma	29	,28	,30	,09
Levene	**Deney**	34	,11	,07	,06
	Karşılaştırma	29			

Tablo 19

Sontest Araştırma Verilerinin Dağılım Değerleri

Test	Grup	n	Problem Çözme Becerisi Algısı (p değeri)	Motivasyon (p değeri)	Akademik Başarı (p değeri)
K-S	**Deney**	34	,08	,20	,20
	Karşılaştırma	29	,20	,20	,20
Shapiro-Wilk	**Deney**	34	,12	,18	,49
	Karşılaştırma	29	,07	,15	,14
Levene	**Deney**	34	,06	,09	,06
	Karşılaştırma	29			

Tablo 20

Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Araştırma Verilerinin Dağılım Değerleri

Test	Grup	n	Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testi (p değeri)
K-S	**Deney**	34	,16
	Karşılaştırma	29	,20
Shapiro-Wilk	**Deney**	34	,28
	Karşılaştırma	29	,41
Levene	**Deney**	34	,84
	Karşılaştırma	29	

Tablolar incelendiğinde verilere ait anlamlılık değerinin ,05 değerinden yüksek olduğu belirlenmiştir. Bu durum verimlerin normal dağılım gösterdiğini ve varyansların homojen olduğunu göstermektedir. Bu sebeple araştırmada parametrik testlerin kullanılmasına karar verilmiştir.

Deney ve karşılaştırma grubundaki öğrencilerin problem çözme becerisi algısı, motivasyon ve akademik başarı değerlerinin analiz edilmesi için öncelikle deney ve karşılaştırma gruplarının öntest puanları arasında anlamlı bir fark olup olmadığına bakılmıştır. İlişkisiz ölçümler için t testi kullanılarak analiz edilen öntest sonuçları Tablo 21'de verilmektedir.

Tablo 21

Öntest Verileri t Testi Sonuçları

Test	Grup	n	$\bar{X}$	SS	sd	t	p
Problem Çözme Becerisi Algısı	**Deney**	34	2,84	,49	61	1,10	,28
	Karşılaştırma	29	2,99	,56			
Motivasyon	**Deney**	34	5,33	,56	61	1,76	,08
	Karşılaştırma	29	5,02	,84			
Akademik Başarı	**Deney**	34	,48	,14	61	,43	,67
	Karşılaştırma	29	,46	,17			

Tablo 21 incelendiğinde grupların öntest puan ortalamaları arasında anlamlı bir fark bulunmadığı görülmektedir (p>,05). Verilerin normal dağılım göstermesi, varyanslarının homojen olması ve grupların öntest puanları arasında anlamlı bir farklılığın bulunmaması sonuçlarına göre araştırma soruları veri analiz yöntemleri belirlenmiştir.

Alice programını kullanan deney grubu öğrencileri ile Alice programını kullanmayan karşılaştırma grubu öğrencilerinin akademik başarı puanları, problem çözme becerisi algıları ve motivasyonları arasında anlamlı bir fark olup olmadığını belirlemek amacıyla ANCOVA kullanılmıştır. Büyüköztürk (2017), ANCOVA'nın genellikle öntest-sontest kontrol gruplu desenlerde, öntest ölçümlerinin ortak değişken olarak tanımlanarak, sontest ölçümleri arasında istatistiksel bir farkın olup olmadığına bakmak için kullanıldığını belirtmektedir.

Öğrencilerin programlamaya hazır bulunuşluk düzeyleri arasında Alice programını kullanıp kullanmama durumlarına göre anlamlı bir farkın olup olmadığını belirlemek amacıyla ilişkisiz ölçümler için t testi kullanılmıştır. İlişkisiz ölçümler için t testi ile ilişkisiz iki grubun ortalama puanları arasında istatistiksel olarak anlamlı bir fark olup olmadığı belirlenmektedir (Büyüköztürk, 2017).

Nicel verilerin analizinde, belirtilen analiz yöntemlerinin kullanılmasının yanı sıra etki büyüklüğü de hesaplanmıştır. Etki büyüklüğü bağımsız değişkenin bağımlı değişken üzerinde ne derece etkili olduğunu gösteren bir değerdir (Cohen, 1988). Etki büyüklüğünün hesaplanmasında Cohen'in d değeri kullanılmıştır. İki grup ortalaması ile çalışılan istatistiksel yöntemlerde, etki büyüklüğü hesaplanması amacıyla genellikle Cohen's d formülü kullanılmaktadır (Özsoy & Özsoy, 2013). Grupların ortalamaları ve harmanlanmış standart sapmaları ile elde edilen Cohen's d formülü aşağıdaki gibidir.

$$SS_{harmanlanmış} = \sqrt{((N_A-1)\ SS^2_A + (N_B-1)\ SS^2_B\ ) / N_{A+}\ N_B}$$

$$\text{Etki Büyüklüğü} = \frac{\text{GrupA ortalaması-GrupB ortamalası}}{\text{Harmanlanmış Standart sapma } (SS_{harmanlanmış})}$$

Cohen's d değeri ,20 ve ,50 arasında bir değere sahipse düşük; ,50 ve ,80 arasında bir değere sahipse orta; ,80 ve daha yüksek bir değere sahipse geniş etki büyüklüğü mevcuttur denilmektedir (Cohen, 1988).

Nitel veriler kapsamında öğrencilerle yapılan görüşme sonuçlarını analiz etmek amacıyla içerik analizi kullanılmıştır. İçerik analizi ile öncelikle verilere ait kodlar oluşturulmuş, oluşturulan kodlara uygun temalar belirlenmiştir (Yıldırım & Şimşek, 2013). Güvenirliği sağlamak amacıyla, araştırmacı haricinde başka bir uzman tarafından kodlamalar yeniden yapılmıştır. Araştırmacı ve uzmanın birbiriyle uyumlu olan maddeleri eşleşen madde, uyumlu olmayan maddeleri eşleşmeyen madde olarak belirlenmiştir. (Eşleşen madde/(Eşleşen madde+Eşleşmeyen madde)*100) formülü kullanılarak veri analizinin güvenirliği hesaplanmıştır (Miles & Huberman, 1994). Kodlayıcılar arasındaki güvenirlik %91,66 olarak bulunmuştur.

BÖLÜM IV

BULGULAR VE YORUMLAR

Bu bölümde öğrencilere uygulanan akademik başarı testine, problem çözme becerisi algısı ölçeğine, motivasyon ölçeğine ve programlamaya hazır bulunuşluk düzeyi belirleme testine ait bulgulara yer verilmektedir. Deney grubu öğrencilerinin programlama dili öğretiminde Alice programı kullanımına ilişkin görüşlerine ait bulgular sunulmaktadır. İlgili bulgulara ait yorumlara, araştırma hipotezlerine uygun olarak yer verilmiştir.

1. Akademik Başarı Testine Ait Bulgular

Hem deneysel süreç öncesinde hem de deneysel süreç sonrasında deney ve karşılaştırma gruplarına uygulanan akademik başarı testinde toplamda 48 soru bulunmaktadır ve her soru bir puan olarak değerlendirilmiştir. Maksimum 48, minimum sıfır puan alınabilen akademik başarı testinde deney ve karşılaştırma gruplarına ait puanların ortalama ve standart sapma değerleri Tablo 22'deki gibidir.

Tablo 22

Akademik Başarı Testine Ait Puanların Ortalama ve Standart Sapma Değerleri

Grup	**Öntest**			**Sontest**		
	n	$\bar{X}$	**SS**	**n**	$\bar{X}$	**SS**
Deney	34	23,00	6,60	34	29,47	6,99
Karşılaştırma	29	22,20	8,11	29	27,00	8,60

Deney grubundaki öğrencilerin deneysel süreç öncesinde akademik başarı testi puan ortalamaları 23,00; deneysel süreç sonrasında puan ortalamaları 29,47'dir. Karşılaştırma grubu öğrencilerinin deneysel süreç öncesinde akademik başarı testi puan ortalamaları 22,20; deneysel süreç sonrasında puan ortalamaları 27,00'dır. Her iki grubunda süreç içerisinde programlama bilgilerini arttırıp puan ortalamalarını yükselttikleri görülmektedir.

Deney ve karşılaştırma gruplarının sontest akademik başarı testi puanları arasında anlamlı bir fark olup olmadığını belirlemek amacıyla ANCOVA kullanılmıştır. Grupların öntest akademik başarı testi puanlarına göre düzeltilmiş sontest akademik başarı testi ortalama puanları Tablo 23'te verilmiştir.

Tablo 23

Önteste Göre Düzeltilmiş Sontest Akademik Başarı Testi Ortalama Puanları

Grup	**n**	$\bar{X}$	**Düzeltilmiş** $\bar{X}$
Deney	34	29,47	29,51
Karşılaştırma	29	27,00	26,95

Grupların düzeltilmiş sontest akademik başarı testi puanları arasında yapılan Bonferroni testi sonuçlarına göre, deney grubunun akademik başarısının ($\bar{X}$=29,51) karşılaştırma grubunun akademik başarısından ($\bar{X}$=26,95) yüksek olduğu görülmektedir. Gözlenen farkın anlamlı olup olmadığını belirlemek amacıyla ANCOVA testi kullanılmıştır. Öntest kontrol edilen ANCOVA kullanılarak analiz edilen sontest sonuçları Tablo 24'te verilmektedir.

Tablo 24

Önteste Göre Düzeltilmiş Sontest Akademik Başarı Testi Puanlarının Gruba Göre ANCOVA Sonuçları

Varyansın Kaynağı	**Kareler Toplamı (KT)**	**sd**	**Kareler Ortalaması (KO)**	**F**	**p**
Öntest (Reg.)	37,721	1	37,721	,620	,434
Grup	**101,910**	**1**	**101,910**	**1,676**	**,200**
Hata	3648,750	60	60,812		
Toplam	3782,000	62			

Tablo 24 incelendiğinde gruplar arasında öntest sonuçlarına göre düzeltilmiş sontest ortalama puanları arasında anlamlı bir farklılık olmadığı görülmektedir (F(1,60)=1,676; p>,05). Bu sonuç Alice programı kullanımının öğrencilerin akademik başarılarına anlamlı bir etkisi olmadığı şeklinde yorumlanabilir.

Alice programı kullanımının akademik başarıya olan etkisine bakıldığı çalışmalar incelendiğinde benzer şekilde Moskal vd. (2004), Schultz (2011) ve Solmaz (2014) yaptıkları çalışmalarda Alice programı kullanımının akademik başarıya anlamlı yönde etki etmediğini tespit etmişlerdir. Programlama öğretiminde Alice programı kullanımının akademik başarıya etki etmediği sonucuyla ilgili olarak; Moskal vd. (2004), kullandıkları düşük güvenirlik gösteren ölçme aracından kaynaklanabileceğini belirtmişlerdir. Schultz (2011), Alice programını uyguladığı programlama dersinin önceki yıllara ait gerçek problemlere uyarlanan sunumlarının katılımcıların derse katılmaları amacıyla önceden gösterilmesi ve Alice programında çözülen problem türleri ile ilk programlama dersinde sunulan iş sorunları arasında bir kopukluk olması nedenlerinden kaynaklanabileceğini ifade etmiştir. Solmaz (2014), Alice programının nesne yönelimli programlama dillerine uygulanmasında olumlu sonuç alınabildiğini ancak kendisinin PHP diline uyguladığını ve bu nedenle olumlu sonuç alamamış olabileceğini vurgulamıştır.

Buna karşın Al-Linjawi ve Al-Nuaim (2010), Biju (2013), Johnsgard ve McDonald (2008), Price (2013), Sykes (2007), Wang vd. (2009) ve Zhang vd. (2014) çalışmalarında Alice programının akademik başarıya olumlu yönde etki ettiğini ifade etmişlerdir. Programlama öğretiminde Alice programı kullanımının akademik başarıya olumlu yönde etki ettiği sonucuyla ilgili olarak; Al-Linjawi ve Al-Nuaim (2010), Alice programında programlama kavramlarının ne işe yaradığı görsel olarak görülebildiği için kavramlar arasında ilişki kurulabildiğini ve nesne yönelimli programlama dilleri ile ilgili bir sezgi geliştirilebildiğini belirtmişlerdir. Biju (2013), öğrencilerin programlama dillerinden bağımsız yazılımlar aracılığıyla programlama kavramları hakkında daha iyi ve derin bir anlayış kazanabildiklerini ve bu nedenle başarı sağlanabileceğini vurgulamıştır. Johnsgard ve McDonald (2008), Alice programının programlama kavramlarını öğrencilere daha açık hale getirmede etki sahibi olduğunu ve programlamaya giriş sınıfında kalitatif gelişmeler sağladığını ifade etmişlerdir. Sykes (2007), Alice programının birçok bilgisayar oyununa benzer nitelikte bağımlılık yapabildiğini, bu bağımlılığın öğrencilerin dikkatini çektiğini ve

programı daha iyi kullanmaya teşvik ettiğini belirtmiştir. Zhang vd. (2014), Alice programının nesne yönelimli programlama dillerinin karmaşıklığına hakim olmayı sağladığını ifade etmişlerdir.

Alanyazın incelendiğinde Alice programının akademik başarıya olumlu yönde etki eden çalışmalardan Al-Linjawi ve Al-Nuaim (2010), Sykes (2007) ile Zhang vd. (2014) araştırmalarında bilgisayar ile ilgili olmayan anabilim dallarında öğrenim gören öğrencilerle çalıştıkları, ayrıca Alice programının akademik başarıya olumlu yönde etki etmediği çalışmalar olan Moskal vd. (2004), Schultz (2011) ve Solmaz (2014)'ın çalışma grubunun bilgisayar ile ilgili anabilim dallarında öğrenim gören öğrenciler olduğu görülmektedir. Araştırmada Nesne Tabanlı Programlama I dersi kapsamında bilgisayar teknolojileri programı öğrencileriyle çalışılmıştır. Nesne Tabanlı Programlama I dersi, öğrencilerin mezun olmak için geçmek zorunda oldukları bir derstir. Yani anabilim dalı bilgisayar olan çalışma grubunda başarılı olmak zorunda oldukları bir ders kapsamında araştırma yapılmıştır. İki grupta derste başarılı olabilmek için çaba göstermişlerdir. Grupların akademik başarı puanları arasında anlamlı bir farkın çıkmamasının nedeni olarak, anabilim dalı bilgisayar olan çalışma grubuyla geçmek zorunda oldukları bir ders kapsamında çalışmanın yürütülmesinden kaynaklandığı düşülmektedir.

2. Problem Çözme Becerisi Algısı Ölçeğine Ait Bulgular

Hem deneysel süreç öncesinde hem de deneysel süreç sonrasında deney ve karşılaştırma gruplarına uygulanan problem çözme becerisi algısı ölçeğinde toplamda 35 madde bulunmaktadır. Ölçeğin puan aralığı 32-192'dir. Ölçekte düşük puan almak yüksek problem çözme becerisi algısını ifade etmektedir. Grupların problem çözme becerisi algısı ölçeğine ait ortalama puan ve standart sapma değerleri Tablo 25'te verilmiştir.

Tablo 25

Problem Çözme Becerisi Algısı Ölçeğine Ait Ortalama Puan ve Standart Sapma Değerleri

Grup	**Öntest**			**Sontest**		
	n	$\overline{X}$	**SS**	**n**	$\overline{X}$	**SS**
Deney	34	99,43	17,02	34	103,23	17,81
Karşılaştırma	29	104,51	19,69	29	106,21	29,99

Deney grubundaki öğrencilerin deneysel süreç öncesinde problem çözme becerisi algısı ölçeği puan ortalamaları 99,43; deneysel süreç sonrasında puan ortalamaları 103,23'dür. Karşılaştırma grubu öğrencilerinin deneysel süreç öncesi puan ortalamaları 104,51; deneysel süreç sonrasında 106,21'dir. Ortalama puanlara bakılarak öğrencilerin deneysel süreç öncesi ve sonrasında orta düzeyde problem becerisi algısına sahip oldukları görülmektedir.

Deney ve karşılaştırma gruplarının sontest problem çözme becerisi algısı ölçeği puanları arasında anlamlı bir fark olup olmadığını belirlemek amacıyla ANCOVA kullanılmıştır. Grupların öntest problem çözme becerisi algısı ölçeği puanlarına göre düzeltilmiş sontest problem çözme becerisi algısı ölçeği ortalama puanları Tablo 26'da verilmiştir.

Tablo 26

Önteste Göre Düzeltilmiş Sontest Problem Çözme Becerisi Algısı Ölçeği Ortalama Puanları

Grup	**n**	$\bar{X}$	**Düzeltilmiş** $\bar{X}$
Deney	34	103,23	103,80
Karşılaştırma	29	106,21	105,55

Grupların düzeltilmiş sontest problem çözme becerisi algısı ölçeği puanları arasında yapılan Bonferroni testi sonuçlarına göre, deney grubunun problem çözme becerisi algısının ($\bar{X}$=103,80) karşılaştırma grubunun problem çözme becerisi algısından ($\bar{X}$=105,55) yüksek olduğu görülmektedir. Gözlenen farkın anlamlı olup olmadığını belirlemek amacıyla ANCOVA testi kullanılmıştır. Öntest kontrol edilen ANCOVA kullanılarak analiz edilen sontest sonuçları Tablo 27'de verilmektedir.

Tablo 27

Önteste Göre Düzeltilmiş Sontest Problem Çözme Becerisi Algısı Ölçeği Puanlarının Gruba Göre ANCOVA Sonuçları

Varyansın Kaynağı	**KT**	**sd**	**KO**	**F**	**p**
Öntest (Reg.)	1198,811	1	1198,811	2,086	,154
Grup	**46,733**	**1**	**46,733**	**,081**	**,776**
Hata	34473,885	60	574,565		
Toplam	35811,314	62			

Tablo 27 incelendiğinde gruplar arasında öntest sonuçlarına göre düzeltilmiş sontest ortalama puanları arasında anlamlı bir farklılık olmadığı görülmektedir (F(1,60)=,081; p>,05). Bu sonuç programlama dili öğretiminde Alice programı kullanımının öğrencilerin problem çözme becerisi algılarına istatistiksel olarak anlamlı bir etki etmediği şeklinde yorumlanabilir.

Alice programı kullanımının problem çözme becerisi algısına olan etkisinin araştırıldığı çalışmalar incelendiğinde, benzer şekilde Solmaz (2014) yaptığı çalışmada Alice programı kullanımının problem çözme becerisi algısına anlamlı yönde etki etmediğini belirtmiştir. Solmaz (2014) programlama öğretiminde, görsel bir dilin kullanılması veya kullanılmaması durumunun, ya da eğitimde kullanılan ortamın türünün öğrencilerin problem çözme becerisi algıları üzerinde farklı etkilere yol açabildiğini ifade etmiştir.

Programlamanın yapısı incelendiğinde temelinde problem ya da problemleri çözmek olduğu görülmektedir (Gomes & Mendes, 2007). Her ne kadar araştırmada deney grubu Alice ve NetBeans programlarını, karşılaştırma grubu NetBeans programını kullanarak çalışmış olsalar da, amaçları Java programlama dilini öğrenmektir. Her iki gruba verilen örnek ve uygulamalar NetBeans veya Alice programı kullanımına bakılmaksızın problem çözme becerisi kullanmayı gerektiren durumları içermektedir. Her iki grup da örnek ve uygulamaları çözmeye, programlamayı yapmaya yoğunlaştıkları için gruplar arasında problem çözme becerisi algısında anlamlı bir fark çıkmamış olabilir.

3. Motivasyon Ölçeğine Ait Bulgular

Deneysel süreç öncesi ve sonrasında deney ve karşılaştırma gruplarındaki öğrencilerin motivasyon düzeylerini belirlemek amacıyla uygulanan motivasyon ölçeği toplam 31 maddeden oluşmaktadır. 31 ile 217 arasında puan alınabilen ölçekte, deney ve karşılaştırma grubu öğrencilerine ait ortalama puan ve standart sapma değerleri Tablo 28'de verilmektedir.

Tablo 28

Motivasyon Ölçeğine Ait Ortalama Puan ve Standart Sapma Değerleri

Grup	Öntest			Sontest		
	n	$\bar{X}$	SS	n	$\bar{X}$	SS
Deney	34	165,27	17,47	34	158,51	20,52
Karşılaştırma	29	155,53	26,17	29	153,98	28,43

Deney grubundaki öğrencilerin deneysel süreç öncesinde motivasyon ölçeği puan ortalamaları 165,27; deneysel süreç sonrasında puan ortalamaları 158,51'dir. Karşılaştırma grubu öğrencilerinin deneysel süreç öncesi puan ortalamaları 155,53; deneysel süreç sonrasında 153,98'dur. Ortalama puanlara bakılarak öğrencilerin deneysel süreç öncesi ve sonrasında yüksek düzeyde motivasyona sahip oldukları görülmektedir.

Deney ve karşılaştırma gruplarının sontest motivasyon ölçeği puanları arasında anlamlı bir fark olup olmadığını belirlemek amacıyla ANCOVA kullanılmıştır. Grupların öntest motivasyon ölçeği puanlarına göre düzeltilmiş sontest motivasyon ölçeği puan ortalamaları Tablo 29'da verilmiştir.

Tablo 29

Önteste Göre Düzeltilmiş Sontest Motivasyon Ölçeği Ortalama Puanları

Grup	n	$\bar{X}$	Düzeltilmiş $\bar{X}$
Deney	34	158,51	158,28
Karşılaştırma	29	153,98	154,26

Grupların düzeltilmiş sontest motivasyon ölçeği puanları arasında yapılan Bonferroni testi sonuçlarına göre, deney grubunun motivasyonunun ($\bar{X}$=158,28) karşılaştırma grubununkinden ($\bar{X}$=154,26) yüksek olduğu görülmektedir. Gözlenen farkın anlamlı olup olmadığını belirlemek amacıyla ANCOVA testi kullanılmıştır. Öntest kontrol edilen ANCOVA kullanılarak analiz edilen sontest sonuçları Tablo 30'da verilmektedir.

Tablo 30

Önteste Göre Düzeltilmiş Sontest Motivasyon Ölçeği Puanlarının Gruba Göre ANCOVA Sonuçları

Varyansın Kaynağı	**KT**	**sd**	**KO**	**F**	**p**
Öntest (Reg.)	82,113	1	82,113	,135	,714
Grup	**240,261**	**1**	**240,261**	**,395**	**,532**
Hata	36464,895	60	607,748		
Toplam	36868,426	62			

Tablo 30 incelendiğinde gruplar arasında öntest sonuçlarına göre düzeltilmiş sontest ortalama puanları arasında anlamlı bir farklılık olmadığı görülmektedir (F(1,60)=,395; p>,05). Bu sonuç programlama dili öğretiminde Alice programı kullanımının öğrencilerin motivasyonlarına istatistiksel olarak anlamlı bir etki etmediği şeklinde yorumlanabilir.

Alice programı kullanımının motivasyona olan etkisinin araştırıldığı çalışmalar incelendiğinde benzer şekilde Wang vd. (2009) yaptıkları çalışmalarda Alice programı kullanımının motivasyona anlamlı yönde etki etmediğini belirtmişlerdir. Programlama öğretiminde Alice programı kullanımının motivasyona etki etmediği sonucuyla ilgili olarak; Wang vd. (2009), öğrencilerin Alice programıyla oluşturulan programlarla geleneksel programlama dilleriyle oluşturulan programlar arasında büyük farklılıklar gördüklerini, oluşturdukları programları nitelik olarak eş değer görmediklerini belirttiklerini ifade etmişler ve bu durumun motivasyonlar arasında fark çıkmama nedeni olabileceğini vurgulamışlardır.

Zhang vd. (2014) çalışmalarında Alice programı kullanımının motivasyona olumlu yönde etki ettiğini ifade etmişlerdir. Programlama öğretiminde Alice programı kullanımının motivasyona olumlu yönde etki ettiği sonucuyla ilgili olarak; Zhang vd. (2014), Alice programının programlama sürecinin karmaşıklığını azalttığını ve bu durumun motivasyona olumlu etki yapmış olabileceğini belirtmişlerdir.

Alanyazın incelendiğinde Alice programıyla programlama yapmak ile geleneksel programlama diliyle programlama yapmak arasında büyük bir farkın bulunduğunu; Alice programının sözdizimi hatalarını yapmanın önüne geçmesi, Alice programında program çıktılarının görsel olması nedeniyle programlama mantık hatalarının nereden

kaynaklandığının görülmesini sağlaması özelliklerinin programlama yapmayı kolaylaştırdığını, ancak geleneksel programlama dilinde programlama yapmanın daha zor olduğunu belirten çalışmaların olduğu görülmektedir (Brown, 2008; Cliburn, 2008; Garlick & Cankaya, 2010; Klassen, 2006; McKenzie, 2009; Powers vd., 2007; Sykes, 2007). Klassen (2006) kodlar Alice programında hazır bir şekilde sürükle-bırak ile eklenirken, programlama dilinde yazılmak zorunda olduğunu, öğrencilerin sözdizimi hataları, kodlama ve mantık hataları ile uğraşmak zorunda kaldıklarını belirterek bu durumun öğrencilerin hayal kırıklığına uğramasına sebep olduğunu ifade etmiştir. Uygulama sürecinde deney grubundaki öğrenciler haftada dört saat olan dersin ilk iki saatinde Alice programıyla çalışmış, son iki saatinde NetBeans programıyla geleneksel programlama dilindeki kodlamalarını yapmışlardır. Öğrencilerin ders kapsamında önce Alice programı ile kolay bir şekilde üç boyutlu animasyonlar yaptıkları ancak sonrasında Java programlama diline geçtiklerinde yeniden sözdizimi hataları ile karşılaştıkları ve yaptıkları bir hatayı bulmakta zorluk çektikleri görülmüştür. Önce Alice programı ile kolay bir şekilde uygulama yapmak, sonrasında geleneksel programlama dilinin sözdizimi ve hata ayıklama işlemleri ile uğraşarak uygulama yapmak, öğrencilerin derse olan ilgililerinin azalmasına sebep olmuş olabilir. Bu durum grupların motivasyonları arasında anlamlı bir farkın çıkmaması nedenlerinden biri olarak düşünülmektedir.

Alanyazında yer alan bir çok çalışmada öğrencilerin; Alice programının programlama yapmayı kolaylaştıran özelliklerinden dolayı, geleneksel programlama diline geçiş sürecinde zorlandıkları belirtilmektedir (Brown, 2008; Cliburn, 2008; Garlick & Cankaya, 2010; Klassen, 2006; McKenzie, 2009; Powers vd., 2007; Sykes, 2007). Wang vd. (2009) öğrencilerin programlama dili algılarında, tüm programlama dillerinde yalnızca metin tabanlı çıktı üretilebileceği düşüncesi olduğunu, bu nedenle Alice programının motivasyona etki etmediğini ifade etmişlerdir. Garlick ve Cankaya (2010) öğrencilerin programlama dili öğrenimde, önceki tecrübelerinden kaynaklı daha geleneksel bir yöntem beklentilerinin olduğunu belirtmişlerdir. Ayrıca Sykes (2007) çalışmasına katılan bir kısım öğrencilerin; hata ayıklama, sözdizimi hatalarını belirleme gibi bilgi becerileri de dahil olmak üzere temel programlama becerilerini geliştirmek için geleneksel bir programlama ortamında daha mutlu olacaklarını belirttiklerini ifade etmiştir. Benzer şekilde araştırmada öğrencilerle yapılan görüşme sonuçları incelendiğinde öğrenciler, Alice programının

programlamayı eğlenceli hale getirdiğini ancak programda kodların değiştirilememesinden ve kodların hazır olarak eklenmesinden dolayı programlamayı ilerletmede yetersiz olduğunu belirtmişlerdir. Alice programı ile çalışmak öğrenciler açısından eğlenceli olsa da programın gerçek dünya problemlerinin çözümü amacıyla program yazmada öğrenciler tarafından yetersiz olarak nitelendirildiği görülmüştür. Bu durum araştırmada grupların motivasyonları arasında anlamlı bir farkın çıkmamasının nedenlerinden biri olabilir.

Brown (2008), Cliburn (2008), Schultz (2011) ve Sykes (2007) kurulum ve işlevsellikle ilgili sorunların Alice programına yönelik tutumları olumsuz yönde etkileyebileceğini belirtmişlerdir. Alice programı ile çalışırken programın yüksek donanım gereksinimlerinden dolayı bilgisayarda kilitlenmeler meydana gelebilmektedir. Ayrıca araştırmada öğrenciler, Alice programının Türkçe dil desteğinin bulunmamasını olumsuz bir özellik olarak belirtmişlerdir. Bu durumların da grupların motivasyonları arasında anlamlı bir fark çıkmamasının nedenlerinden biri olarak düşünülmektedir.

4. Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testine Ait Bulgular

Programlama dillerinin kendilerine özgü sözdizimleri bulunsa da hepsinde kullanılan temel programlama kavramlarının çoğu ortaktır (Gomes & Mendes, 2007; Sureau, 2012). Başarılı bir programcı olmak, temel kavramları çok iyi bilmeyi gerektirmektedir (Kak, 2009). Bu bağlamda programlamaya hazır bulunuşluk düzeyinin önemi ortaya çıkmaktadır. Araştırmada deneysel süreçten 18 hafta sonra gruplara uygulanan programlamaya hazır bulunuşluk düzeyi belirleme testinde toplam 32 soru bulunmaktadır. Her soru bir puan olarak değerlendirilmiş ve testten maksimum 32, minimum sıfır puan alınmaktadır. Gruplar arasındaki programlamaya hazır bulunuşluk düzeyi belirleme testi verilerinin anlamlı olup olmadığını belirlemek amacıyla, ilişkisiz ölçümler için t testi kullanılarak sonuçlar analiz edilmiştir. İlişkisiz ölçümler için t testi kullanılarak analiz edilen programlamaya hazır bulunuşluk düzeyi belirleme testi sonuçları Tablo 31'de verilmiştir.

Tablo 31

Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testi Verileri t Testi Sonuçları

Grup	n	$\bar{X}$	SS	sd	t	p
Deney	34	20,35	6,26	61	2,60	,01
Karşılaştırma	29	16,17	6,45			

Tablo 31 incelendiğinde grupların programlamaya hazır bulunuşluk düzeyi belirleme testi ortalama puanları arasında deney grubu lehine anlamlı bir farklılık bulunduğu belirlenmiştir (p<,05). Deney grubundaki öğrencilerin programlamaya hazır bulunuşluk düzeyleri karşılaştırma grubundaki öğrencilere göre daha yüksektir. Bu durum programlama öğretiminde Alice programı kullanımı ile programlamaya hazır bulunuşluk düzeyi arasında deney grubu lehine anlamlı bir ilişki olduğu şeklinde yorumlanabilir. Bunun yanında etki büyüklüğü değeri (Cohen's d) ,66 olarak bulunmuştur. Yani Alice programı kullanımının programlamaya hazır bulunuşluk düzeyine orta düzeyde etki ettiği söylenebilir.

Alice programı kullanımının programlama öğretimine etkileri ile ilgili çalışmalar incelendiğinde, öğrencilerin programlamaya hazır bulunuşluk düzeylerini arttırdığı ve programlamanın temel kavramlarının ne işe yaradığını daha iyi bir şekilde anlamalarını sağladığı görülmektedir (Berge vd., 2003; Bishop-Clark vd., 2007; Cliburn, 2008; Cooper vd., 2000b, 2003; Daly, 2011; Howard vd., 2006; Liu vd., 2011; Wang vd., 2009; Werner vd., 2009; Zaccone vd., 2003). Programlama öğretiminde Alice programı kullanımının programlamaya hazır bulunuşluk düzeyi üzerinde olumlu etki yaptığı sonucuyla ilgili; Bishop-Clark vd. (2007) programcı olmayan katılımcılara kısa bir ünite kapsamında Alice programı kullanılmasının, programlama becerisi ve temel programlama kavramlarının anlaşılması konularında güven duyulmasını sağladığını belirtmişlerdir. Cliburn (2008), Alice programının sürükle-bırak özelliği sayesinde öğrencilerin söz dizimi hatalarıyla uğraşmak ya da kod ayıklamak yerine programlama kavramlarına odaklandığını vurgulamıştır. Cooper vd. (2000b), Alice programında kod işlevlerinin animasyon şeklinde görülebilmesi sayesinde kod ve komutların daha iyi anlaşıldığını ifade etmişlerdir. Alice programının kullanıcılarda programlama yapma becerisiyle ilgili güven duygusu geliştirdiğini belirten Daly (2011), Howard vd. (2006) ve Liu vd. (2011) arasından Daly

(2011) bu güven duygusunun rahat bir programlama ortamı sağladığını bu sayede kullanıcıların programlama mantığını ve programlama kavramlarını daha iyi bir şekilde anladıklarını vurgulamış, Howard vd. (2006) ilgili güven duygusuyla kullanıcıların algoritmalar ile Alice hikayeleri arasındaki ilişkiyi kavrayarak temel programlama kavramlarını daha iyi anladıklarını belirtmiş, Liu vd. (2011) bu durumun programlamanın temel kavramlarını öğrenmeyi kolaylaştırdığını ifade etmişlerdir.

Buna karşın Garlick ve Cankaya (2010) çalışmalarında Alice programı kullanımının programlamanın temel kavramlarını öğrenmeye anlamlı bir etkisinin olmadığını belirtmişlerdir. Programlama öğretiminde Alice programı kullanımının programlamaya hazır bulunuşluk düzeyi üzerinde etkisinin olmadığı sonucuyla ilgili; Garlick ve Cankaya (2010), öğrencilerin programlama dili öğretiminden beklentilerinin daha geleneksel yöntemler olduğunu, bu durumun Alice programı kullanılarak tasarlanan öğretim ortamlarının programlamanın yapısının ve temel kavramlarının daha iyi öğrenilmesinin önüne geçtiğini vurgulamışlardır.

Brown (2008) Alice programının soyut kavramları öğretmede avantajlı olduğunu; Cooper vd. (2000a), Hutchinson, Moskal, Cooper ve Dann (2006), Powers vd. (2007) grafiksel çıktıların öğrencilerin programlama kavramlarını daha iyi anlamalarına yardımcı olduğunu belirttikleri görülmektedir. Öğrenciler Alice programı ile çalışırken, çalışma ekranına ekledikleri her bir kodun etkisine teker teker bakabilir ve böylece kodların görsel olarak ne işe yaradığını görebilir. Programdaki görsellik, programlamadaki soyut kavramları ve programlamanın yapısını hayal etmekten çıkararak kodlar ve görseller arasında ilişki kurmayı sağlar. Bu durumun öğrencilerin programlamanın temelini oluşturan kavramları daha iyi öğrenmelerini ve programlamaya hazır bulunuşluk düzeylerini arttırmalarını sağlayan avantajlardan biri olduğu düşünülmektedir.

Alice programının programlama öğretiminde kullanımına ilişkin öğrenci görüşleri incelendiğinde öğrencilerin Alice programında sürükle-bırak özelliğinin olmasından dolayı programla çalışırken sözdizimi hataları ile uğraşmadıklarını ve kodlamada iş yüklerinin azaldığını belirttikleri görülmektedir. Ayrıca öğrenciler, Alice programında her bir kodun etkisine animasyon halinde görsel bir şekilde bakabildikleri için kalıcılığı arttırdığını ifade etmişlerdir. Alice programıyla çalışırken öğrenciler kod yazmak ve sözdizimi hatalarıyla uğraşmak yerine kodların ne işe yaradığına odaklanmaktadırlar. Ekledikleri her kodun

nasıl bir etki gösterdiğini animasyon halinde görebilmektedirler. Böylece Alice programıyla programlama dili temel kavramlarının, programlama ve algoritma mantığının iyi bir şekilde öğrenilmesi sağlanıp, öğrenmedeki kalıcılık arttırılabilir. Bu durumun Alice programı kullanılarak tasarlanan programlama öğretiminin programlamaya hazır bulunuşluk düzeyine deney grubu lehine anlamlı yönde olumlu olarak etki yapmasını sağlayan avantajlardan biri olduğu düşünülmektedir.

5. Programlama Öğretiminde Alice Programı Kullanımına İlişkin Öğrenci Görüşleri

Bu araştırma sorusu kapsamında, beş uzman görüşüne sunularak son hali verilen ve 9 sorudan oluşan yarı yapılandırılmış görüşme formu kullanılmıştır. Uygulama sürecinde Alice programı kullanılarak programlama öğretimi gören grup deney grubu olduğu için, deneysel süreç sonunda deney grubu öğrencileriyle 9'ar kişilik iki grup oluşturulmuş ve toplam 18 öğrenciyle iki odak gruplu görüşme yapılmıştır. Her odak gruplu görüşme yaklaşık iki saat sürmüştür ve toplamda 17 sayfalık görüşme kaydı elde edilmiştir. Görüşme sonuçları içerik analizi kullanılarak analiz edilmiştir. Güvenirliği sağlamak amacıyla araştırmacı haricinde başka bir uzman tarafından kodlamalar yeniden yapılmış ve güvenirlik %91,66 olarak bulunmuştur. Analiz sonuçları aşağıda verilmektedir.

Analiz sonuçlarına göre elde edilen görüşler dört tema altında toplanmıştır. Temalar aşağıda verilmiştir:

- Alice programının programlama dili öğreniminde sağladığı yararlar
- Alice programının olumlu özellikleri
- Alice programının olumsuz özellikleri
- Alice programının ileride kullanımı

Alice programının programlama dili öğreniminde sağladığı yararlar teması; Alice programının özelliklerinin, programlamaya ve programlama dili öğrenimi üzerine etkilerini içeren kodları kapsamaktadır. Tema; programlamaya olan etkisi, görsellik ve sürükle-bırak olmak üzere üç kategori altında incelenmiştir. Programlamaya olan etkisi; Alice programının programlama öğrenirken sağladığı avantaj ve kolaylıkların, öğrencilerin programlamaya olan tutumlarına etkilerinin incelendiği kategoridir. Bu kategori; temel kod

kavramlarının öğrenilmesini kolaylaştırması, programlamayı öğrenme isteğini arttırması, programlamayı öğrenmeyi kolaylaştırması, programlamayı eğlenceli hale getirmesi, programlamayı araştırma ve sorgulama isteğini arttırması, algoritma mantığını öğrenmeye yardımcı olması kodlarından oluşmaktadır. Görsellik; Alice programında eklenilen her kodun, tasarlanan her programın çıktısının animasyon şeklinde görülmesinin programlamaya olan etkisinin incelendiği kategoridir. Bu kategori, kalıcılığı arttırması ve kodlama hatalarının görülmesini kolaylaştırması kodlarından oluşmaktadır. Sürükle-bırak, Alice programında kod yazmak yerine kodların programa hazır bir şekilde fare yardımıyla eklenmesinin programlamaya olan etkisinin incelendiği kategoridir. Bu kategori, sözdizimi hatalarının önüne geçmesi ve kodlamada iş yükünü azaltması kodlarından oluşmaktadır. Alice programının programlama dili öğreniminde sağladığı yararlar temasına ait analizler Tablo 32'de verilmektedir.

Tablo 32

Alice Programı ile Tasarlanan Ortamın Kullanışlılığı ve Programlama Dili Öğrenimi Üzerine Yararı ile İlgili Görüşler

Öğrencilerin Alice Programı ile Tasarlanan Ortamın Programlama Dili Öğreniminde Sağladığı Yararlar ile İlgili Görüşleri		**n**	**f**
Programlamaya Olan Etkisi	Temel kod kavramlarının öğrenilmesini kolaylaştırması	18	35
	Programlamayı öğrenme isteğini arttırması	17	38
	Programlamayı öğrenmeyi kolaylaştırması	16	17
	Programlamayı eğlenceli hale getirmesi	13	15
	Programlamada araştırma, sorgulama isteğini arttırması	11	17
	Algoritma mantığını anlamaya yardımcı olması	1	1
Görsellik (Program çıktılarının görsel bir şekilde görülmesi)	Kalıcılığı arttırması	9	14
	Kodlama hatalarının görülmesini kolaylaştırması	7	11
Sürükle -Bırak	Sözdizimi hatalarının yapılmasının önüne geçmesi	8	8
	Kodlamada iş yükünü azaltması	7	13

Alice programının programlama dili öğreniminde sağladığı yararlar ile ilgili öğrenci görüşleri incelendiğinde, öğrenciler en çok; temel kod kavramlarının öğrenilmesini kolaylaştırdığını belirtmiştir. Temel kod kavramlarının öğrenilmesini kolaylaştırması kodu; Alice programında her kodun etkisine ayrı ayrı, animasyon halinde, görsel bir şekilde bakılmasının, programlamanın temel kod kavramlarının ne işe yaradığını kolaylaştırdığını ifade etmektedir. Temel kod kavramlarının öğrenilmesini kolaylaştırması ile ilgili bir

öğrenci "*Alice'de kodların temelini ve belirli kalıpların nasıl kullanıldığını anladım. Bunu Java'ya aktardım.*" yorumunu, başka bir öğrenci "*Alice'de kodlamanın mantığını, Java'da kodlamayı öğrendim. Alice'de öğrendiğim temel kavramlar ile Java'da hangi yapının kullanılacağına karar verdim.*" yorumunu yapmıştır. Öğrenci görüşlerinde en çok yoğunlaşılan ikinci kod olan programlamayı öğrenme isteğini arttırması, Alice programında kolay bir şekilde programlama yapabilmenin Java programında da programlama yapma isteğini arttırmasını ifade etmektedir. Programlamayı öğrenme isteğini artırması ile ilgili bir öğrenci "*Alice programında kolay bir şekilde programlama yapabildiğimi gördüğümde Java'yı daha çok öğrenmek ve programlar yapmak istedim.*" görüşünü belirtmiştir. Öğrenci görüşlerinde yoğunlaşılan üçüncü kodun, Alice programında çalışırken kodların hangi sırayla kullanılması gerektiğini görsel bir şekilde görmenin, programlama mantığını anlamayı kolaylaştırmasını ifade eden, programlamayı öğrenmeyi kolaylaştırması olduğu görülmektedir. Bir öğrenci "*Ekranda kodları gördüğümüz için kodları nasıl ve nerede kullanacağımıza dair bir fikir oluştu. Alice Java'ya göre daha kolaydı ama Java'da Alice sayesinde kolaylaştı.*" yorumu ile, başka bir öğrenci "*Alice'de neyi niçin yaptığımı görüyorum. Alice'de gördüğüm için Java'da da aynı şekilde programlamayı yapabiliyorum.*" yorumuyla programlama öğrenmeyi kolaylaştırması ile ilgili görüşünü ifade etmiştir.

Cevaplar ayrıntılı bir şekilde incelendiğinde, 13 öğrenci Alice programının, programlamayı eğlenceli hale getirdiğini ifade etmiştir. Bir öğrenci Alice programının programlamaya olan etkisi ile ilgili "*Programlamayı animasyonla öğrenmek çok eğlenceli. Java'da da ilerlemek, farklı şeyler yapmak istiyorum.*" görüşünü belirtmiştir. 11 öğrenci, Alice programının programlamada araştırma ve sorgulama isteğini arttırdığını belirtmiştir. Programlamada araştırma, sorgulama isteğini arttırması kodu, Alice programında farklı uygulamaların yapılabilmesinin, Java programlama dilinde de farklı uygulamalar yapılabileceği ve buna yönelik araştırma yapılması görüşlerini kapsamasının yanında, Alice programında istenilen programlarının kod yazamama nedeniyle geliştirilememesinin programlama dilinde ilerleme ve sorgula isteğini arttırmasını da ifade etmektedir. Bir öğrenci "*Alice'de bu uygulamaları yapabiliyorsam Java'da da yapabilirim diye düşündüm ve başka hangi kodların olduğunu araştırıyorum.*" yorumunu, başka bir öğrenci "*Alice belirli bir düzeye geldikten sonra isteklere cevap vermiyor. Bu yüzden farklı platformlarda*

yazmaya başlıyorsunuz. Bu durum beni Java'da program yazmaya, araştırmaya itti." yorumunu yapmıştır. Bir öğrenci, Alice programının algoritma mantığını anlamaya yardımcı olduğunu ifade etmiştir. Algoritma mantığını anlamaya yardımcı olması kodu, programların yazılmadan önce nasıl bir yol izlenmesi gerektiğinin planlandığı algoritmanın oluşturulma mantığını daha iyi anlamaya yardımcı olduğu görüşlerini içermektedir. Öğrenci, "*Alice algoritma oluşturma için faydalı. Algoritma mantığını güzel bir şekilde öğretiyor.*" yorumu ile Alice programının algoritma oluşturmaya olan etkisini belirtmiştir.

Alice programının programlama dili öğrenimi üzerine etkisi ile ilgili alanyazın incelendiğinde; Biju (2013), Cooper vd. (2000a), Cooper vd. (2003), Hayat vd. (2017), Johnsgard ve McDonald (2008), Schultz (2011), Solmaz (2014) çalışmalarında Alice programının temel kod kavramlarını öğrenmeye yardımcı olduğunu ifade etmişlerdir. Courte, Howard ve Bishop-Clark (2006), Hayat vd. (2017) yaptıkları çalışmalarda, öğrencilerin Alice programının kendilerini programlamaya teşvik ettiğini belirttiklerini vurgulamışlardır. Al-Linjawi ve Al-Nuaim (2010), Cooper vd. (2000b), Howard vd. (2006), McKenzie (2009), Powers vd. (2007), Solmaz (2014), Zhang vd. (2014) çalışmalarında Alice programının programlama dillerinin karmaşık yapısını öğrenmeyi ve programlama mantığınının anlaşılmasını kolaylaştırdığını belirtmişlerdir. Aktunc (2013), Courte vd. (2006), Howard vd. (2006), Hutchinson vd. (2006), Solmaz (2014) Alice programının algoritmik düşünme becerisini geliştirdiğini vurgulamışlardır. Brown (2008), Rodger vd. (2009) ve Wang vd. (2009) Alice programı ile çalışan öğrencilerin çoğunun, programlama yaparken kendi yöntemlerini kullandıklarını ve programlamada farklı özellikleri kendilerinin keşfetmek için istekli olduğunu ifade etmişlerdir. Courte vd. (2006), Solmaz (2014) yaptıkları çalışmalarda, öğrencilerin Alice programı ile çalışırken programlamayı eğlenceli bulduklarını belirttiklerini aktarmışlardır. Biju (2013) Alice programı ile çalışan öğrencilerin ödevlerini tamamlamada yüksek motivasyona sahip olduklarını, Mullins vd. (2009) Alice programı kullanımı ile Programlamaya Giriş kursunu bırakan öğrencilerde azalma meydana geldiğini, Sykes (2007) Bilgisayar Bilimi I dersini alan öğrenci sayısında Alice programı kullanımıyla birlikte %33'lük bir artışın olduğunu, McKenzie (2009) Alice programı ile tasarlanan programlama dersini alan öğrencilerin bir sonraki dönem yine bir programlama dersi almaya karar verdiklerini tespit etmişlerdir.

Buna karşın Garlick ve Cankaya (2010) çalışmalarında, Java programlama dili ile çalışan öğrencilerin Alice programı ile çalışan öğrencilerden daha fazla programlamadan zevk aldıklarını, programlamada yeniliklere açık olduklarını, araştırma yapma isteklerinin daha fazla olduğunu, temel programlama dili kavramlarını daha iyi anladıklarını ifade etmişlerdir. Hutchinson vd. (2006) çalışmalarında Alice programıyla öğrencilerin programlama becerilerinin geliştiğini ancak programlamaya yönelik tutumlarının aynı kaldığını belirtmişlerdir.

Alice programında program çıktılarının görsel bir şekilde görülmesiyle ilgili 9 öğrenci, kalıcılığı arttırdığı yönünde görüş bildirmiştir. Kalıcılığı arttırması kodu, görülen bilgilerin daha kalıcı olmasını ve programlama dilinde karşılaşılan problemlerin Alice uygulamalarının hatırlanarak çözülmesini kapsamaktadır. Kalıcılığı arttırmasındaki etkisini bir öğrenci "G*ördüğüm şeyi daha iyi anlıyorum. Daha iyi akılda kalıcı oluyor.*" yorumu ile ifade etmiştir. Yedi öğrenci kodlama hatalarının görülmesini kolaylaştırdığını belirtmiştir. Kodlama hatalarının görülmesini kolaylaştırması kodu, Alice programında program çıktıları görsel bir şekilde görüldüğü için yapılan hatanın nereden kaynaklandığının kolay bir şekilde bulunmasını ifade etmektedir. Kodlama hatalarının kolay bir şekilde bulunması ile ilgili bir öğrenci "*Hatalarımızı görmemiz daha da kolaylaşıyor. Java'da sadece yazı olduğu için hataları bulmada zorlanıyorum. Ama Alice'de hata görsel bir şekilde çıkıyor*." yorumunu, başka bir öğrenci "*Java'da tam olarak nerede hata yaptığımı anlayamıyorum. Alice'de ise o hatayı görebiliyorum."* yorumunu yapmıştır.

Alice programının program çıktılarının görsel bir şekilde görülmesi ile ilgili alanyazın incelendiğinde; Al-Linjawi ve Al-Nuaim (2010), Cooper vd. (2003), Solmaz (2014) çalışmalarında Alice programının görselliği sayesinde öğrencilerin programlamada neyin yanlış gittiğini izleyebildiğini, kolayca hata ayıklama ve düzeltme işlemlerini yapabildiğini belirtmişlerdir. Solmaz (2014) çalışmasında öğrencilerin, Alice programının kalıcılığı arttırdığını ifade ettiklerini aktarmıştır. McKenzie (2009) Alice programında kazanılan becerilerin, programlamadaki daha genel sorunlara aktarılabileceğini vurgulamıştır. Klassen (2006) görselliğin öğrenmeyi daha zevkli hale getirdiğini belirtmiştir.

Sürükle-bırak mantığı ile kodların hazır bir şekilde çalışmaya eklenmesiyle ilgili; sekiz öğrenci sözdizimi hatalarının yapılmasını engellediğini belirtmiştir. Sözdizimi hatalarının yapılmasının önüne geçmesi kodu, Alice programında kod yazmak yerine kodlar sürükle

bırak yöntemiyle hazır bir şekilde çalışma ekranına eklendiğinden, sözdizimi hatalarının yapılmasının önüne geçildiğini ifade etmektedir. Yedi öğrenci kodlamada iş yükünü azalttığını belirtmiştir. Kodlamada iş yükünü azaltması kodu, kodların hazır bir şekilde eklenmesi sayesinde kod yazımıyla zaman harcanmamasını ifade etmektedir. Bir öğrenci "*Alice'de kod hatası olmuyor. Kodu çalışma ekranına fareyle bırakıyorum. Kod yazmakla da uğraşmıyorum. Uygulamayı yapmaya odaklanıyorum.*" yorumu ile sürükle-bırak mantığının avantajlarını belirtmiştir.

Alice programının sürükle-bırak mantığıyla çalışması ile ilgili alanyazın incelendiğinde Brown (2008), Solmaz (2014) çalışmalarında Alice programının sürükle-bırak mantığıyla çalışması ile doğru sözdizimi yapıldığını vurgulamışlardır. Sykes (2007), Wang vd. (2009), Zaccone vd. (2003) Alice programının sürükle-bırak özelliği sayesinde öğrencilerin sözdizimi ezberleme ya da hata ayıklama yükünden kurtulduğunu, bu zamanı uygulama yapmak için kullanabildiklerini tespit etmişlerdir. Cliburn (2008) çalışmasına katılan öğrencilerin en çok sürükle-bırak özelliği sayesinde kodlama hatasının yapılmamasını sevdiklerini aktarmıştır.

Alice programının olumlu özellikleri teması; kullanımının kolay olması, sürükle-bırak mantığı ile çalışması, kodların ne işe yaradığının görsel olarak görülmesi, eğlenceli olması, nesnelerin ayrıntılı bir şekilde hareket ettirilebilmesi, arayüz tasarımının ilgi çekici olması kodlarından oluşmaktadır. Bu temaya ait analizler Tablo 33'te verilmektedir.

Tablo 33

Alice Programı ile İlgili Olumlu Öğrenci Görüşleri

Öğrencilerin Alice Programının Olumlu Özellikleri ile ilgili Görüşleri	**n**	**f**
Kullanımının kolay olması	18	21
Sürükle-bırak mantığı ile çalışması	17	17
Kodların ne işe yaradığının görsel olarak görülmesi	13	16
Eğlenceli olması	9	12
Nesnelerin ayrıntılı bir şekilde hareket ettirilebilmesi	4	4
Arayüz tasarımının ilgi çekici olması	2	2

Öğrencilerin Alice programının olumlu özellikleri ile ilgili görüşleri incelendiğinde en çok; kullanımının kolay olduğunu belirttikleri görülmektedir. Kullanımının kolay olması kodu, Alice programının arayüzünün anlaşılır olmasını ve programlama yaparken aranılan özelliklerin kolay bir şekilde bulunmasını ifade etmektedir. Kullanımının kolay olması ile ilgili bir öğrenci "*Alice'in kullanımı rahattı. Zorlayan bir şey olmadı. Aradığımı rahatça buldum.*" yorumunu yapmıştır. Öğrenci görüşlerinin ikinci olarak, sürükle-bırak mantığı ile çalışması kodu üzerinde yoğunlaştığı görülmektedir. Sürükle-bırak mantığı ile çalışması kodu, programda hazır bir şekilde tanımlanmış kodların fare yardımıyla çalışma ekranına eklenmesini temsil etmektedir. Programın kolay olması ve sürükle-bırak mantığı ile çalışmasıyla ilgili bir öğrenci "*Kullanımı çok kolaydı. Kod yazma yok. Sürükle-bırak mantığını çok sevdim.*" yorumunu yapmıştır. Öğrenci görüşlerinin üçüncü olarak, kodların ne işe yaradığının görsel olarak görülmesi kodu üzerinde yoğunlaştığı görülmektedir. Kodların ne işe yaradığının görsel olarak görülmesi kodu, Alice programında eklenilen kodların nasıl bir etkisinin olduğuna animasyon şeklinde bakılabilmesini içermektedir. Bir öğrenci "*Kodların görsel olarak görülmesi çok güzel. Böylece her birinin ne işe yaradığına ayrı ayrı bakabiliyorum.*" yorumu ile kodların görsel olarak işlevlerinin görülmesini sevdiğini belirtmiştir.

Cevaplar ayrıntılı bir şekilde incelendiğinde, 9 öğrenci Alice programı ile çalışmayı eğlenceli bulmuştur. Bir öğrenci "*Bana göre kolay ve eğlenceliydi. Uygulama yaptıkça daha çok yapmak istiyordum.*" yorumuyla programa karşı olan olumlu tutumunu belirtmiştir. Dört öğrenci, nesnelerinin ayrıntılı bir şekilde hareket ettirilebilmesini beğendiğini ifade etmiştir. Nesnelerin ayrıntılı bir şekilde hareket ettirilebilmesi kodu, programda eklenen çalışma karakterlerinin (nesne) kolları, bacakları vb. uzuvlarının ayrıntılı olarak hareket ettirilebilmesini kapsamaktadır. Bir öğrenci "*Nesnelere ayrı ayrı özellik vermek çok hoşuma gitti.*" yorumunu, başka bir öğrenci "*Karakterlerin kanatlarını, ayaklarını, kafalarını hareket ettirmek çok güzel.*" yorumunu yapmıştır. İki öğrenci programın arayüz tasarımını ilgi çekici bulmuştur. Arayüz tasarımının ilgi çekici olması kodu; programın açılış ekranı, çalışma ekranı, karakter ekleme ekranı vb. özelliklerinin dikkati üzerinde toplayan bir şekilde tasarlandığını ifade etmektedir. Bir öğrenci arayüz tasarımının ilgi çekici olması ile ilgili "*Görselliği çok iyiydi. Çok hoşuma gitti.*" şeklinde görüş belirtmiştir.

Alice programının olumlu görülen özellikleri ile ilgili alanyazın incelendiğinde Cliburn (2008), Cooper vd. (2003), Hayat vd. (2017), Howard vd. (2006), Sykes (2007) yaptıkları çalışmalarda, öğrencilerin Alice programının kullanımını kolay ve basit bulduklarını belirtmişlerdir. Cooper vd. (2003) sürükle-bırak editörünün yeni başlayan kullanıcılar için faydalı olduğunu vurgulamışlardır. Solmaz (2014) çalışmasında öğrencilerin kodların hazır olarak eklenebilmesi özelliğini sevdiklerini aktarmıştır. Kodların ne işe yaradığının görsel bir şekilde görülmesi özelliği ile ilgili; Cliburn (2008) ve Solmaz (2014) öğrencilerin bu özelliği beğendiğini, Powers vd. (2007) özelliğin öğrencilerin kodların ne işe yaradığını anlamalarına yardımcı olduğunu, Cooper vd. (2003) özelliğin büyük bir avantaj olduğunu ve öğrencilerin gözlem yeteneğini arttırıp derse olan ilgilerinin sürdürülmesine katkıda bulunduğunu vurgulamışlardır. Cooper vd. (2000b), Hayat vd. (2017), Howard vd. (2006), Solmaz (2014), Sykes (2007), Werner vd. (2009) çalışmalarında öğrencilerin Alice programını eğlenceli olarak değerlendirdiklerini tespit etmişlerdir.

Araştırmanın nitel sonuçları ve alanyazın incelendiğinde, görsel bir şekilde kod işlevlerinin ne işe yaradığının görülmesi ile sürükle-bırak editörü sayesinde kodların ne işe yaradığını anlamaya ve programlama yapmaya yoğunlaşılmasının öğrenciler ve araştırmacılar tarafından sevildiği, programlama öğrenmede büyük bir avantaj olarak görüldüğü belirlenmiştir. Çalışmada Alice programı ile programlama öğretiminin, öğrencilerin programlamaya hazır bulunuşluk düzeylerine anlamlı bir etkisinin olduğu belirlenmesi bu verileri destekler niteliktedir.

Alice programının olumsuz özellikleri teması; Alice programının özellikleri ve programlamaya olan etkisi olmak üzere iki kategori altında incelenmiştir. Alice programının özellikleri kategorisinde; programda negatif bulunan özellikler belirtilmiştir. Bu kategori; Türkçe dil desteğinin bulunmaması, nesne ve sahnelerde çeşitliliğin az olması, kodların değiştirilememesi kodlarından oluşmaktadır. Programlamaya olan etkisi kategorisi; Alice programı ile programlama öğrenirken negatif bulunan durumları içermektedir. Bu kategori; kod yazmaya imkân vermemesi nedeniyle kod yazma pratikliğinin kazanılamaması ve ileri seviye programlama için yetersiz görülmesi kodlarından oluşmaktadır. Bu temaya ait analizler Tablo 34'te verilmektedir.

Tablo 34

Alice Programı ile İlgili Olumsuz Öğrenci Görüşleri

Öğrencilerin Alice Programının Olumsuz Özellikleri ile İlgili Görüşleri		**n**	**f**
Alice Programının Özellikleri	Türkçe dil desteğinin bulunmaması	9	10
	Nesne ve sahnelerde çeşitliliğin az olması	7	7
	Kodların değiştirilememesi	3	7
Programlamaya Olan Etkisi	Kod yazmaya imkân vermemesi nedeniyle kod yazma pratikliğinin kazanılamaması	13	16
	İleri seviye programlama için yetersiz görülmesi	11	13

Öğrencilerin Alice programında gördükleri olumsuz özellikler incelendiğinde en çok; kod yazmaya imkân vermemesi nedeniyle kod yazma pratikliğinin kazanılamaması maddesi üzerinde yoğunlaşıldığı görülmektedir. Kod yazmaya imkân vermemesi nedeniyle kod yazma pratikliğinin kazanılamaması, programda kodların fare yardımıyla sürüklenip bırakılmasından dolayı, programlamada önemli bir beceri olan kod yazma pratikliğinin geliştirilememesini içermektedir. Bir öğrenci "*Alice bize kodu yazdırmıyor. Sadece sürükle-bırak yaptırıyor. Kodları kendimiz yazsak yazılışlarını öğrenme açısından daha faydalı olurdu.*" yorumuyla kod yazma pratikliğinin kazanılamamasını eleştirmiştir.

Öğrenci görüşlerinin ikinci olarak, Alice programının ileri seviye programlama için yetersiz görülmesi kodunda yoğunlaştığı görülmektedir. İleri seviye programlama için yetersiz görülmesi kodu, programda kod yazılamaması ve kod eklenilememesi nedeniyle yeni ve farklı uygulamaların tasarlanamaması, bu durumun programlama bilgisini ilerletmede yetersiz görülmesi görüşlerini içermektedir. Kod yazma pratikliğinin kazanılamaması ve programlamayı ilerletememe konusunda bir öğrenci "*Mantık açısından faydalı olduğunu düşünüyorum. Ama kod yazamaya gelecek olursak çok faydası olduğunu*

düşünmüyorum. Öğrenmede etkisi oldu fakat geliştirmede değil. Sadece ilk evrede faydası oldu." yorumunu yapmıştır. Başka bir öğrenci ileri seviye programlama için yetersiz görülmesi ile ilgili "*Alice yeni başlayanlar için güzel olabilir. Ama programlamayı ileri seviyeye taşımak için kullanışlı bir program değil.*" görüşünü belirtmiştir.

Öğrenci görüşlerinin üçüncü olarak Türkçe dil desteğinin bulunmaması kodu üzerinde yoğunlaştığı görülmektedir. Türkçe dil desteğinin bulunmaması kodu, programda Türkçe dil desteği olmamasından dolayı hem programın, hem de karakterlerin bazı özelliklerinin anlaşılamamasından kaynaklanan olumsuz görüşleri içermektedir. Bir öğrenci "*İngilizce olmasından dolayı bazı şeylerin ne işe yaradığını bulamadım. O yönünü sevmedim.*" yorumu ile Türkçe dil desteğinin olmamasını eleştirmiştir.

Cevaplar ayrıntılı bir şekilde incelendiğinde, yedi öğrencinin Alice uygulamalarına eklenebilen nesne ve sahnelerde çeşitliliğin az olmasını olumsuz bir özellik olarak ifade ettikleri görülmüştür. Nesne ve sahnelerde çeşitliliğin az olması kodu, Alice programındaki karakterlerin ve sahnelerin birçoğunun hayali olması, hayali olmayanların ise sınırlı sayıda olmasından dolayı yapılan animasyonlarda çeşitliliğin sağlanamaması görüşlerini içermektedir. Bir öğrenci "*Ekleyebileceğimiz karakterler çok sınırlı. Hayalimdeki uygulamaları yaparken sıkıntı çekiyorum.*" yorumu ile nesnelerin çeşitliliğin az olduğunu belirtmiştir. Üç öğrenci yaptıkları uygulamalarda kodların değiştirilememesini sevmediklerini ifade etmiştir. Kodların değiştirilememesi kodu, programda kodların hazır olarak tanımlanmasından dolayı kullanıcıların kodları değiştirememeleri ve istedikleri uygulamaları yapamamaları görüşlerini içermektedir. Öğrencilerden biri "*Alice bizi kodlarda kalıplara bağlıyor. Kodlarda istediğimiz değişiklikleri yapamıyoruz. Uygulamaları Alice'in bize verdiği sınırlar çerçevesinde yapmak zorunda kalıyoruz.*" yorumunu yapmıştır.

Alice programının olumsuz özellikleri ile ilgili alanyazın incelendiğinde Sykes (2007) çalışmasına katılan öğrencilerin bazılarının; kod yazma, sözdizimi, derleme veya sorun çözme gibi programlama dillerinde var olan ve programlamada ilerlemeyi sağlayan sorunlarla ilgili Alice programında herhangi bir gelişimin sağlanamaması nedeniyle kod yazma, hata ayıklama vb. çekirdek programlama becerilerini geliştirmek amacıyla geleneksel bir programlama ortamını tercih ettiklerini belirttiklerini vurgulamıştır. Solmaz (2014) çalışmasında öğrencilerin, Alice programına kod yazılamamasını, kodların hazır

olmasını ve değiştirilememesini olumsuz yönde eleştirdiklerini, program dilinin İngilizce olmasını beğenmediklerini belirtmiştir. Brown (2008), Cliburn (2008), Garlick ve Cankaya (2010), McKenzie (2009), Powers vd. (2007), Sykes (2007) çalışmalarında Alice programıyla programlama yapmak ile programlama diliyle programlama yapmak arasında büyük bir fark olduğunu ve programın gerçek dünya problemli sorunları çözmeye uygun olmadığını vurgulamışlardır. Aktunc (2013), Brown (2008) Alice programında karakterler ile sahnelerin sınırlı nitelikte olmasından dolayı farklı uygulamaların yapılamadığını tespit etmişlerdir. Howard vd. (2006), Werner vd. (2009) çalışmalarında öğrencilerin Alice programındaki yazılımsal kısıtlamaları sevmediklerini ifade etmişlerdir. Schultz (2011) Alice programının bu haliyle programlamada ilerlemede yetersiz kaldığını, programa geleneksel programlamadaki kod işlemlerinin eklenmesi ile programlamada ilerlenebileceğini belirtmiştir.

Alice programı ile ilgili öğrenci görüşleri ve alanyazın incelendiğinde; Alice programında kod yazılamaması nedeniyle kod yazma, hata ayıklama gibi temel programlama becerilerinin kazanılamadığının ve programın gerçek dünya problemlerini çözmede, ileri seviye programlama yapmada yetersiz görüldüğünün olumsuz yönde eleştirildiği tespit edilmiştir. Ayrıca Türkçe dil desteğinin olmaması olumsuz özellikler arasındadır. Temel programlama becerilerinin kazanılamaması ve gerçek problem durumlarının çözülmesinde yetersiz olduğu algıları ile Türkçe dil desteğinin bulunmaması, çalışmada Alice programı ile programlama öğretiminin, öğrencilerin motivasyonlarını anlamlı yönde etkilemediği bulgusunu destekler niteliktedir. Alice programında kod yazılamaması, hata ayıklama yapılamaması nedeniyle programlama sorunlarını çözmede kod yazma, hata ayıklama vb. konularda çaba sarf edilememesi, kodların değiştirilememesi nedeniyle farklı çözüm yolları denemeye imkân vermemesi; Alice programı ile programlama öğretiminin, öğrencilerin problem çözme becerisi algılarını anlamlı yönde etkilemediği bulgusunu destekler niteliktedir.

Öğrencilerin Alice programının ileride kullanımı ile ilgili görüşleri Tablo 35'te verilmiştir.

Tablo 35

Alice Programının İleride Kullanım Durumu ile İlgili Öğrenci Görüşleri

Öğrencilerin Alice Programının İleride Kullanıma İlişkin Görüşleri	**n**	**f**
İleride programlama dili öğretecek olsam Alice programını kullanırım	17	17
Alice programını kullanmaya devam etmek isterim	13	14

Öğrencilerin 17'sinin Alice programını ileride programlama dili öğretmeleri gerekirse tercih edecekleri görülmektedir. Bir öğrenci "*İlk başlarda kullanırım. Temel kavramları öğrenmeleri açısından etkili olur.*" yorumunu yapmıştır. Öğrenci görüşlerinde 13 öğrencinin, programı kullanmaya devam edeceklerini belirttikleri görülmektedir. Alice programını ileride kullanmaya devam etmek isterim görüşü, hem programlamayı geliştirmek, hem de programın sevilmesinden dolayı farklı animasyonlar oluşturmak amacıyla kullanımları içermektedir. Bir öğrenci "*Alice'i ileride kullanmak isterim. Yeni hikayeler, uygulamalar yaparken programlamamı geliştiririm.*" yorumunu yapmıştır. Başka bir öğrenci "*Programlama dilini öğrenme için kullandık. Bir süre sonra vakit kaybı olmaya başladı. İleride kullanmayı düşünmüyorum.*" yorumuyla Alice programını ileride kullanmayacağını belirtmiştir.

Alice programının ileride kullanımıyla ve ders anlatımında bir yaklaşım olarak ele alınmasıyla ilgili alanyazın incelendiğinde Solmaz (2014) çalışmasına katılan öğrencilerin çoğunun gelecekte programlama dili öğretiminde Alice programını kullanmak, Alice programını kullanmaya devam ederek kendilerini daha çok geliştirmek istediklerini aktarmıştır. Hayat vd. (2017) öğrencilerin Alice programını bilgisayar bilimleri öğretiminde yeni bir yaklaşım olarak gördüklerini ifade etmişlerdir. Buna karşın Cliburn (2008) programlama öğretiminde sadece Alice programının kullanılmasının yetersiz olduğunu, öğrencileri dersin amacı konusunda belirsizliğe itebileceğini ve sadece Alice programını müfredata dahil etmenin dersi zorlaştırabileceğini belirtmiştir. Aktunc (2013), Al-Linjawi ve Al-Nuaim (2010), Cliburn (2008) programlama dersinde Alice programı kullanımın yararlı olacağını ancak önce Alice programı ile temel kavramların öğretilmesi sonrasında programlama dilinde ilerlenmesi gerektiğini vurgulamışlardır. McKenzie

(2009) Bilgisayar Giriş derslerinde Alice programı kullanımının arttığı görülsede, özellikle bütün bir dersin Alice programı ile tasarlanmasının nadir görüldüğünü ifade etmiştir.

BÖLÜM V

SONUÇ VE ÖNERİLER

Bu bölümde araştırma hipotezlerine göre analizleri yapılan araştırma bulgularından elde edilen sonuçlara ve bundan sonra yapılacak çalışmalar için önerilere yer verilmiştir.

1. Sonuçlar

Bu çalışmanın amacı; Alice programı ile programlama öğretiminin öğrencilerin akademik başarısına, problem çözme becerisi algısına, motivasyonuna, programlamaya hazır bulunuşluk düzeyine etkisini, Alice programı kullanımında engelleyici ve kolaylaştırıcı faktörleri belirlemektir. Çalışma kapsamında deney ve karşılaştırma gruplarından nicel ve nitel veriler toplanarak analiz edilmiştir.

Nicel veriler analiz edilerek Alice programı ile programlama öğretiminin; akademik başarı, problem çözme becerisi algısı, motivasyon ve programlamaya hazır bulunuşluk düzeyi üzerindeki etkisi belirlenmiştir. Sonuçlar Tablo 36'da verilmektedir.

Tablo 36

Alice Programı ile Programlama Öğretiminin Ölçülen Değişkenler Üzerindeki Etkisi

Ölçülen Değişken	**Anlamlı Fark**
Akademik Başarı	Yok
Problem Çözme Becerisi Algısı	Yok
Motivasyon	Yok
Programlamaya Hazır Bulunuşluk Düzeyi	Var

Tablo 36 incelendiğinde Alice programı ile programlama dili öğretiminin öğrencilerin akademik başarıları, problem çözme becerisi algıları ve motivasyonları ile ilişkili olmadığı; Alice programı ile programlama dili öğretiminin öğrencilerin programlamaya hazır bulunuşluk düzeyleri ile ilişkili olduğu ve programlamaya hazır bulunuşluk düzeyini anlamlı olarak olumlu yönde etkilediği görülmektedir. Programlamaya hazır bulunuşluk düzeyinin programlama dili öğretiminde çok önemli bir yeri bulunmaktadır. Çünkü programlama dilleri değişse de temel programlama kavramlarının çoğu aynıdır. Öğrenenler farklı programlama dili öğrenimine geçtiklerinde her seferinde temel programlama kavramlarını yeniden öğrenmek için çaba sarf etmekte, programlamada ilerlemekten çok kavramların ne işe yaradığını anlamaya yoğunlaşmaktadır. Bu durumda programlamayı geliştirmek ve ilerletmek açısından bu bulgunun önemi ortaya çıkmaktadır.

Nitel veriler analiz edilerek öğrencilerin programlama dili öğretiminde Alice programı kullanımına ilişkin değerlendirmeleri belirlenmiştir. Öğrenciler; Alice programının temel kod kavramlarının öğrenilmesini ve programlama mantığını anlamayı kolaylaştırdığını, programlamayı öğrenme isteğini arttırdığını ifade etmiştir. Alice programının kullanımını kolay bulmuşlardır. Sürükle-bırak mantığı ile çalışmasını ve kodların görsel bir şekilde ne işe yaradığının görülmesini sevdiklerini ifade etmişlerdir. Ancak Alice programında kod yazılamaması nedeniyle kod yazma pratikliğinin geliştirilememesini ve Türkçe dil desteğinin olmamasını olumsuz yönde eleştirdikleri, Alice programını ileri seviye programlama için yetersiz gördükleri belirlenmiştir. Ayrıca öğrencilerin çoğu, ileride programlama dili öğretmeleri gerekirse Alice programını öğrenme ortamı olarak kullanacaklarını ve ileride Alice programını kullanmaya devam ederek programlamalarını geliştireceklerini belirtmiştir.

Öğrencilerin Alice programıyla programlama öğretiminin, programlamanın temel kod kavramlarını öğrenmeye yardımcı olduğu ve programlama öğrenme, araştırma, sorgulama isteğini arttırması görüşleri; programlamaya hazır bulunuşluk düzeyinin deney grubu lehine anlamlı yönde fark çıkması ile ilişkili niteliktedir. Öğrencilerin Türkçe dil desteğinin bulunmaması nedeniyle programın veya nesnelerin bazı özelliklerini anlayamamaları, Alice programını programlamayı ilerletmede yetersiz bulmaları, Alice programıyla çalışırken temel programlama becerilerinin kazanılamadığı ile ilgili görüşleri; Alice programıyla programlama öğretiminin motivasyon ile ilişki olmadığı sonucunu

destekler niteliktedir. Öğrencilerin Alice programında kodları değiştirememeleri ve kod yazma pratikliğinin kazanılamaması ile ilgili görüşleri, Alice programı ile programlama öğretiminin akademik başarı ve problem çözme becerisi algısı ile ilişki olmadığı sonucunu destekler niteliktedir. Nitel ve nicel veriler birlikte incelendiğinde öğrenci görüşleri ile nicel sonuçların birbirini desteklediği söylenebilir.

2. Öneriler

Bu bölümde araştırmada elde edilen sonuçlar kapsamında gelecekte yapılacak çalışmalara ışık tutulabilmesi amacıyla hem eğitmenlere hem öğretim tasarımcılarına hem de araştırmacılara yönelik önerilere yer verilmiştir.

2.1. Eğitmenlere Yönelik Öneriler

Bu çalışmada meslek yüksekokulu bilgisayar teknolojileri bölümü öğrencileri ile çalışılmış ve Alice programı kullanılarak tasarlanan programlama öğretiminin öğrencilerin programlamaya hazır bulunuşluk düzeylerine anlamlı yönde olumlu bir etkisinin olduğu belirlenmiştir. Meslek yüksekokulları sektöre ara eleman yetiştiren, uygulama ve teorik eğitimin bir arada verildiği önemli eğitim kurumlarıdır. Bu bölümde öğrenciler farklı programlama dersleri almalarına rağmen programlama derslerindeki başarı ortalamaları düşük olabilmektedir. Bilgisayar teknolojileri bölümünde öğrencileri derste aktif hale getirecek Alice programı vb. görsel programların kullanılması, öğrencilerin programlama temeli oluşturmalarına olanak sağlayıp, programlama yapma becerilerini arttırılabilir. Böylece bilgisayar teknolojileri bölümünde özellikle uygulamaya dönük çalışmaların arttırılması sağlanabilir.

2.2. Öğretim Tasarımcılarına Yönelik Öneriler

Araştırmada deney grubunda önce Alice programı ile sonrasında Java programlama diliyle çalışılmıştır. Öğrencilerin programlama dili öğretiminde Alice programı kullanımına ilişkin görüşleri incelendiğinde, Alice programını ileri seviye programlama için yetersiz buldukları görülmüştür. Alice programı ile programlama öğretimi yapılmak isteniyorsa,

önce birkaç hafta Alice programı ile çalışılarak programlama mantığı ve programlamanın temel kod kavramları öğretilip, sonrasında geleneksel programla dili öğretimine geçilerek örnek ve uygulamalar yapılabilir. Böylece öğrenciler; Alice programının programlamayı öğrenmeyi kolaylaştıran bir araç olduğunu, esas geleneksel programlama diliyle gerçek dünya problemlerini çözeceklerini daha iyi bir şekilde anlayabilir. Alice programının yapısı geleneksel programlama dilinin yapısından çok farklıdır. Önce Alice programı ile kolay bir şekilde uygulama yapmak, sonrasında geleneksel programlama dilinin sözdizimi ve hata ayıklama işlemleri ile uğraşarak uygulama yapmak, öğrencileri negatif etkiliyor olabilir.

2.3. Araştırmacılara Yönelik Öneriler

Araştırmada, meslek yüksekokulu bilgisayar teknolojileri bölümü öğrencileriyle çalışılmıştır. Normal öğretim ve ikinci öğretim öğrencilerinden normal öğretim öğrencileri deney, ikinci öğretim öğrencileri karşılaştırma grubu olarak yansız atanmıştır. Nicel veriler akademik başarı testi, motivasyon ölçeği, problem çözme becerisi algısı ölçeği, programlamaya hazır bulunuşluk düzeyi belirleme testi ile toplanan verilerle; nitel veriler görüşme formu aracılığıyla yapılan odak gruplu görüşme sonuçları ile sınırlıdır. Uygulama süreci; Nesne Tabanlı Programlama I dersi kapsamında Gagné'nin dokuz aşamalı öğretim modeli kullanılarak, haftada dört saat olmak üzere sekiz hafta sürmüştür.

Bu araştırmada Alice programı ile programlama öğretiminin; öğrencilerin akademik başarılarını, problem çözme becerisi algılarını ve motivasyonlarını etkilemediği, programlamaya hazır bulunuşluk düzeylerini olumlu yönde etkilediği görülmüştür. Bu değişkenler kullanılarak aynı sonuçların elde edilip edilemeyeceği farklı çalışmalarla ortaya konulabilir. İlgili alanyazın incelendiğinde Alice programının özellikle motivasyon ve problem çözme becerisi algısı değişkenleri üzerine etkisini inceleyen çalışmaların yok denecek kadar az olduğu görülmektedir.

İleriki çalışmalarda önceden programlama dersi almayan çalışma grubu ile araştırmalar yapılabilir. Öğrenciler deneysel sürecin gerçekleştirildiği dönemden bir önceki dönemde programlama dersi almışlardır. Bu nedenle öntest puanları yüksek çıkmış ve öntest-sontest puanları arasında deney grubu lehine anlamlı bir fark çıkmamış olabilir.

İleriki çalışmalarda Alice programı ile elde edilen programlama becerilerinin, sadece uygulama yapılan derse değil, o dersi takip eden başka bir programlama dersine nasıl aktarıldığı araştırılabilir. Araştırma Nesne Tabanlı Programlama I dersi kapsamında yapılmıştır. Grupların akademik başarıları arasında anlamlı yönde bir fark çıkmamış ancak programlamaya hazır bulunuşluk düzeyleri arasında deney grubu lehine anlamlı yönde bir fark bulunmuştur.

Araştırmada bilgisayar teknolojileri bölümünde öğrenim gören öğrenciler ile çalışılmıştır. İleriki çalışmalarda önlisans öğrencileriyle yapılan çalışmalar arttırılarak Alice programının önlisans düzeyindeki etkisine bakılabilir. Alice programı ile yapılan çalışmalar incelendiğinde özellikle önlisans düzeyinde yapılan çalışmalar yok denilecek kadar azdır.

Araştırma kapsamında deney grubu öğrencileriyle, hazırlanan bir görüşme formu aracılığıyla odak gruplu görüşmeler yapılmıştır. İleriki araştırmalarda Alice programı ile çalışan öğrencilerle birebir görüşmeler yapılarak veriler çeşitlendirilebilir. Ayrıca sadece deney grubu öğrencileriyle değil, deney ve karşılaştırma grubu öğrencileriyle görüşmeler yapılarak, hem Alice programıyla yapılan eğitimin hem de geleneksel öğrenme ortamında yapılan eğitimin avantaj ve dezavantajları belirlenebilir.

Araştırmada Alice programı ile programlama öğretiminin öğrencilerin programlamaya hazır bulunuşluk düzeyini olumlu yönde etkilediği görülmüştür. İleriki çalışmalarda Alice programı, programlama dersleri verilirken temel kavramların ve programlama mantığının öğretimi için kullanılabilir. Bu bulgu programlamayı ilerletmek ve geliştirmek, programlama dillerini öğretmek açısından ileriki çalışmalara ışık tutabilecek niteliktedir.

KAYNAKLAR

Akbaba, S. (2006). Eğitimde motivasyon. *Atatürk Üniversitesi Kazım Karabekir Eğitim Fakültesi Dergisi,0*(13), 343-361.

Aktunc, O. (2013). A teaching methodology for introductory programming courses using Alice. *International Journal of Modern Engineering Research (IJMER), 3*(1), 350-353.

Al-Linjawi, A. A., & Al-Nuaim, H. A. (2010). Using Alice to teach novice programmers OOP concepts. *Journal of King Abdulaziz University: Science, 22*(1), 59-68.

Alice. (2018). Retrieved from https://alice3degitim.weebly.com

Ambrosio, A. P., Costa, F. M., Almeida, L., Franco, A., & Macedo, J. (2011, October). *Identifying cognitive abilities to improve CS1 outcome.* Paper presented at the Frontiers in Education Conference (FIE), Rapid City.

Apiola, M., & Tedre, M. (2012). New perspectives on the pedagogy of programming in a developing country context. *Computer Science Education, 22*(3), 285-313.

Arslan, A. (2009). Yapılandırmacı öğrenme yaklaşımı ve Türkçe öğretimi. *Atatürk Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, 13*(1), 143-154.

Baştemur Kaya, C., & Çakır, H. (2015, May). *Meslek yüksekokulu bilgisayar teknolojileri bölümü öğrencilerinin mesleki bilgi ve kariyer yeterlilikleri: İhtiyaç analizi.* 9th International Computer & Instructional Technologies Symposium' da sunulmuş bildiri, Anadolu Üniversitesi, Afyonkarahisar.

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

Berge, O., Borge, R. E., Fjuk, A., Kaasbøll, J., & Samuelsen, T. (2003, November). *Learning object-oriented programming.* Paper presented at the Norsk Informatik Konferanse (Norwegian Informatics Conference), Oslo.

Biju, S. M. (2013). Taking advantage of Alice to teach programming concepts. *E-Learning and Digital Media, 10*(1), 22-29.

Binici, H., & Necdet, A. (2004). Mesleki ve teknik eğitimde arayışlar. *Gazi Üniversitesi Gazi Eğitim Fakültesi Dergisi, 24*(3), 383-396.

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

Bransford, J. D., & Stein, B. (1984). *The ideal problem solver: A guide for improving thinking, learning, and creativity*. New York: W.H. Freeman.

Brown, P. H. (2008). Some field experience with Alice. *Journal of Computing Sciences in Colleges, 24*(2), 213-219.

Bucci, P., Long, T. J., & Weide, B. W. (2001, February). *Do we really teach abstraction?.* Paper presented at the ACM Sigcse Bulletin, North Carolina.

Büyüköztürk, Ş. (2017). *Sosyal bilimler için veri analizi el kitabı.* Ankara: Pegem.

Büyüköztürk, Ş., Çakmak, E. K., Akgün, Ö. E., Karadeniz, Ş., & Demirel, F. (2017). *Bilimsel araştırma yöntemleri*. Ankara: Pegem.

Byrne, P., & Lyons, G. (2001). The effect of student attributes on success in programming. *The ACM Sigcse Bulletin, 3*(33), 49-52.

Cervesato, I. (2008). *On teaching programming languages using a wiki*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.362.8731&rep=rep1&type=pdf

Creswell, J. W., & Clark, V. L. P. (2017). *Designing and conducting mixed methods research*. UK: Sage.

Cliburn, D. C. (2008, October). *Student opinions of Alice in CS1.* Paper presented in 38th ASEE/IEEE Frontiers in Education Conference NY, Saratoga Springs.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences 2nd edn*. Hillsdale: Erlbaum Associates.

Cooper, S., Dann, W., & Pausch, R. (2000a). Alice: a 3-D tool for introductory programming concepts. *Journal of Computing Sciences in Small Colleges, 15*(5), 107-116.

Cooper, S., Dann, W., & Pausch, R. (2000b, November). *Developing algorithmic thinking with Alice.* Paper presented in the 17th Information Systems Education Conference ISECON 2000, 506-539, Philadelphia, PA.

Cooper, S., Dann, W., & Pausch, R. (2003). Using animated 3d graphics to prepare novices for CS1. *Computer Science Education, 13*(1), 3-30.

Cooperative, How Can Teachers Incorporate (2009). *What is cooperative learning?*Retrieved from https://www.calpro-online.org/documents/CooperativeLearningFinal.pdf

Coşar, M. (2013). *Problem temelli öğrenme ortamında bilgisayar programlama çalışmalarının akademik başarı, eleştirel düşünme eğilimi ve bilgisayara yönelik tutuma etkileri.* Doktora Tezi, Gazi Üniversitesi Eğitim Bilimleri Enstitüsü, Ankara.

Courte, J., Howard, E. V., & Bishop-Clark, C. (2006). Using Alice in a computer science survey course. *Information Systems Education Journal, 4(87*), 3-7. Retrieved from http://isedj.org/4/87/ISEDJ.4(87).Courte.pdf

Çakmak, M., & Tertemiz, N. (2002). *Problem çözme.* Ankara: Gündüz.

Çölkesen, R. (2002). *Bilgisayar programlama ve yazılım mühendisliğinde veri yapıları ve algoritmalar.* İstanbul: Papatya.

Daly, T. (2011). Minimizing to maximize: an initial attempt at teaching introductory programming using Alice. *Journal of Computing Sciences in Colleges, 26*(5), 23-30.

Deci, E. L., Koestner, R., & Ryan, R. M. (1999). A meta-analytic review of experiments examining the effects of extrinsic rewards on intrinsic motivation. *Psychological Bulletin, 125*(6), 627-668.

Deryakulu, D. (2000). Yapıcı öğrenme. A. Şimşek (Ed.), *Sınıfta demokrasi* içinde (s. 77). Ankara: Eğitim-Sen.

Dick, W., Carey, L., & Carey, J. O. (2005). *The systematic design of instruction*. USA: Pearson.

Doherty, L., & Kumar, V. (2009, August). *Teaching programming through games.* Paper presented in International Workshop on Technology for Education. Bangalore.

Driscoll, M. (1994). *Psychology of learning for instruction.* Boston, MA: Allyn and Bacon.

Davis, D. M. (2017). *Alice, BlueJ and StarLogo TNG next generation environments for learning programming*. Retrieved from http://gisshop. com/isd/portfolio/files/davis_instr_prog_envs. pdf

Dunn, R. S., & Dunn, K. J. (1978). *Teaching students through their individual learning styles: A practical approach*. Reston, VA: Reston.

Ersoy, H., Madran, R. O., & Gülbahar, Y. (2011). Programlama dilleri öğretimine bir model önerisi: Robot programlama. *Akademik Bilişim 2011 Konferansı*, 731-736.

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

Feddon, J. S., & Charness, N. (1999, January). *Component relationships depend on skill in programming.* Paper presented at the 11th Annual PPIG Workshop, Leeds.

Ford Jr, J. L. (2015). *Programming for the absolute beginner*: Boston, MA: Thomson.

Gagné, R. (1985). *The conditions of learning and theory of instruction*. New York: Holt, Rinehart and Winston.

Gagné, R. M., & Driscoll, M. P. (1988). *Essential of leaning for instruction*. Hinsdale, IL: Dryden Press.

Garlick, R., & Cankaya, E. C. (2010, June). *Using Alice in CS1: A quantitative experiment.* Paper presented at the Proceedings of the fifteenth annual conference on Innovation and technology in computer science education, Ankara.

Garner, S. (2003). Learning resources and tools to aid novices learn programming. The *Informing Science & Information Technology Education Joint Conference (INSITE)*, 213-222.

Gomes, A., & Mendes, A. J. (2007, September). *Learning to program-difficulties and solutions.* Paper presented at the International Conference on Engineering Education–ICEE, Coimbra.

Goold, A., & Rimmer, R. (2000). Factors affecting performance in first-year computing. *ACM Sigcse Bulletin, 32*(2), 39-43.

Grant, N. S. (2003, October). *A study on critical thinking, cognitive learning style, and gender in various information science programming classes.* Paper presented in the 4th Conference on Information Technology Curriculum, New York.

Greenfoot. (2018). Retrieved from https://www.greenfoot.org/overview

Gundurao, H. K., Manjunath, N. S., & Nachappa, M. N. (2010). *Computer technology and computer programming*. Mumbai, IND: Global Media.

Gültekin, K. (2006). *Çoklu ortamın bilgisayar programlama başarısı üzerine etkisi.* Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

Hagan, D., & Markham, S. (2000). *Teaching java with the BlueJ environment.* Paper presented at the Australasian Society for Computers in Learning in Tertiary Education Conference ASCILITE 2000. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.1985&rep=rep1&type=pdf

Har, L. B. (2005). *What is cooperative learning?*. Retrieved from https://www.eduhk.hk/aclass/Theories/cooperativelearning.pdf

Hayat, K., Al-Shukaili, N. A., & Sultan, K. (2017). Alice in Oman. *Education and Information Technologies, 22*(4), 1553-1569.

Haşlaman, T., & Aşkar, P. (2007). Programlama dersi ile ilgili özdüzenleyici öğrenme stratejileri ve başarı arasındaki ilişkinin incelenmesi. *Hacettepe Üniversitesi Eğitim Fakültesi Dergisi, 32*(0), 110-122.

Helm, J. H., & Katz, L. G. (2000). *Young investigators: The project approach in the early years*: New York: Teachers College.

Helminen, J., & Malmi, L. (2010). Jype-a program visualization and programming exercise tool for Python. *The Proceedings of the 5th International Symposium on Software Visualization*, 153-162.

Henderson, P. B. (1986). Proceedings of the 17th SIGCSE '86. *The Technical Symposium on Computer Science Education*, 257-263.

Hernandez, C. C., Silva, L., Segura, R. A., Schimiguel, J., Ledón, M. F. P., Bezerra, L. N. M., & Silveira, I. F. (2010). Teaching programming principles through a game engine. *CLEI electronic journal, 13*(2), 1-8. Retrieved from https://pdfs.semanticscholar.org/d2e7/b317626c1cab2695a826752e0f299a46074a.pdf

Holden, E., & Weeden, E. (2003). The impact of prior experience in an information technology programming course sequence. *The Proceedings of the 4th Conference on Information technology Curriculum*, 41-46.

Howard, E. V., Evans, D., Courte, J., & Bishop-Clark, C. (2006, July). *A qualitative look at Alice and pair-programming.* Paper presented in The 23rd Information Systems Education Conference (ISECON 2006), Dallas.

Hung, Y.-C. (2008). The effect of problem-solving instruction on computer engineering majors' performance in Verilog programming. *IEEE Transactions on Education, 51*(1), 131-137.

Hutchinson, A., Moskal, B., Cooper, S., & Dann, W. (2006, June). *The Alice curriculum: Impact on women in programming courses.* Paper presented at the Proceedings of the 2006 ASEE Annual Conference and Exposition, Chicago.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). Instructional strategy in the teaching of computer programming: a need assessment analyses. *TOJET: The Turkish Online Journal of Educational Technology, 9*(2), 125-131. Retrieved from http://tojet.net/articles/v9i2/9214.pdf

Jiau, H. C., Chen, J. C., & Ssu, K.-F. (2009). Enhancing self-motivation in learning programming using game-based simulation and metrics. *IEEE Transactions on Education, 52*(4), 555-562.

Johnsgard, K., & McDonald, J. (2008, April). *Using Alice in overview courses to improve success rates in programming i*. Paper presented in the 21st Conference on Software Engineering Education and Training, Charleston.

Kak, A. (2009). *Teaching programming.* Retrieved from https://engineering.purdue.edu/kak/TeachingProgramming.pdf

Kalkınma Bakanlığı (2014). *Onuncu kalkınma planı 2014-2018 mesleki eğitimin yeniden yapılandırılması çalışma grubu raporu*. https://abdigm.meb.gov.tr/projeler/ois/egitim/022.pdf sayfasından erişilmiştir.

Karsten, R., Kaparthi, S., & Roth, R. M. (2005). Teaching programming via the web: A time-tested methodology. *College Teaching Methods & Styles Journal (CTMS), 1*(3), 73-82.

Kayabaşı, E. (2016). *Öğretmen adaylarının Alice deneyimi: 3B ortamda programlama.* Yüksek Lisans Tezi, Uludağ Üniversitesi Eğitim Bilimleri Enstitüsü, Bursa.

Kesici, T., & Kocabaş, Z. (2001). *Liseler için bilgisayar 2*. Ankara: MEB.

Kingsley-Hughes, A., & Kingsley-Hughes, K. (2005). *Beginning Programming*. Hoboken, NJ, USA: Wiley.

Klassen, M. (2006, July). *Visual approach for teaching programming concepts.* Paper presented in The 9th International Conference on Engineering Education (ICEE 2006), San Juan.

Klopfer, E., Scheintaub, H., Huang, W., Wendel , D. & Roque, R. (2009). The Simulation cycle: combining games, simulations, engineering and science using StarlogoTNG. *E-Learning*, *6*(1), 71-96.

Klopfer, E., & Begel, A. (2003). StarLogo under the hood and in the classroom. *Kybernetes*, *32*(1/2), 15-37. https://doi.org/10.1108/03684920310452328

Koberg, D. (1981). *The all new universal traveler: A soft-systems guide to creativity, problem-solving, and the process of reaching goals.* Menlo Park: Calif Crisp.

Konvalina, J., Stephens, L., & Wileman, S. (1983). Identifying factors influencing computer science aptitude and achievement. *AEDS Journal, 16*(2), 106-112.

Kölling, M. (2010). The Greenfoot programming environment. *ACM Transactions on Computing Education (TOCE), 10*(4), Article 14. http://doi.acm.org/10.1145/1868358.1868361.

Kölling, M., & Rosenberg, J. (1996). *An object-oriented program development environment for the first programming course*. Paper presented at the 27th SIGCSE Technical Symposium on Computer Science Education, ACM, Philadelphia, March 1996. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.39.2003&rep=rep1&type=pdf

Köse, U. (2010). A web based system for project-based learning activities in "web design and programming" course. *Procedia-Social and Behavioral Sciences, 2*(2), 1174-1184.

Kwon, D., Yoon, I., & Lee, W. (2011). Design of programming learning process using hybrid programming environment for computing education. *KSII Transactions on Internet and Information Systems (TIIS), 5*(10), 1799-1813.

Lau, W. W., & Yuen, A. H. (2011). Modelling programming performance: Beyond the influence of learner characteristics. *Computers & Education, 57*(1), 1202-1213.

Lin, H., & Kuo, T. (2010, June). *Teaching programming technique with edutainment robot construction.* Paper presented in 2nd International Conference on Education Technology and Computer (ICETC), Shanghai.

Linn, M. C., & Dalbey, J. (1985). Cognitive consequences of programming instruction: Instruction, access, and ability. *Educational Psychologist, 20*(4), 191-206.

Liu, J., Lin, C., Hasson, E. P., & Barnett, Z. D. (2011, March)). *Introducing computer science to K-12 through a summer computing workshop for teachers.* Paper

presented at the Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, Dallas.

Liu, M., & Hsiao, Y.-P. (2002). Middle school students as multimedia designers: A project-based learning approach. *Journal of Interactive Learning Research, 13*(4), 311-337.

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004). Scratch: a sneak preview [education]. *Second International Conference on Creating, Connecting and Collaborating Through Computing*, 104-109.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(4), Article 16. http://doi.acm.org/10.1145/1868358.1868363

Matthíasdóttir, Á. (2006). How to teach programming languages to novice students? Lecturing or not. *The International Conference on Computer Systems and Technologies-CompSysTech*, 15-16.

McCaffrey, C. (2006). *StarLogo TNG: The convergence of graphical programming and text processing*. Retrieved from https://dspace.mit.edu/bitstream/handle/1721.1/36904/80770344-MIT.pdf?sequence=2

McKenzie, B. (2009, November). *Introductory programming with Alice as a gateway to the computing profession.* Paper presented in the 23rd Annual Conference for Information Systems Educators (ISECON 2006), Dallas.

Mesleki ve Teknik Eğitim Programlar ve Öğretim Materyalleri [MEGEP] (2007). *Bilişim teknolojileri alanı çerçeve öğretim programı*. http://maol.meb.gov.tr/web/mem/alanlar/bilisim/bilisim_cerceve.pdf sayfasından erişilmiştir.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*: UK: Sage.

Miller, M., & Nunn, G. D. (2001). Using group discussions to improve social problem-solving and learning. *Education, 121*(3), 470-476.

Moskal, B., Lurie, D., & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. *ACM Sigcse Bulletin, 36*(1), 75-79.

Moursund, D. (2001). *Problem-based learning and project-based learning*. Retrieved from http://darkwing.uoregon.edu/~moursund/Math/pbl.htm.

Mullins, P., Whitfield, D., & Conlon, M. (2009). Using Alice 2.0 as a first language. *Journal of Computing Sciences in Colleges, 24*(3), 136-143.

Norvell, T., & Bruce-Lockhart, M. (2004). Teaching computer programming with program animation. *Proc. 2004 Canadian Conference on Computer and Software Engineering Education*, 1-9.

Öğrenci Seçme ve Yerleştirme Sistemi [ÖSYS] (2011). *2011-ÖSYS: Yükseköğretim programları ve kontenjanları kılavuzu*. http://www.osym.gov.tr/TR,1092/2011-osys-yuksekogretim-programlari-ve-kontenjanlari-kilavuzu.html sayfasından erişilmiştir.

Ölçme Seçme ve Yerleştirme Merkezi [ÖSYM] (2011a). M*erkezi yerleştirme ile öğrenci alan yükseköğretim lisans programları*. https://dokuman.osym.gov.tr/pdfdokuman/2017/OSYS/YER/Tablo-4_12082017.pdf sayfasından erişilmiştir.

Ölçme Seçme ve Yerleştirme Merkezi [ÖSYM] (2011b). Merkezi yerleştirme ile öğrenci alan yükseköğretim önlisans programları. https://dokuman.osym.gov.tr/pdfdokuman/2017/OSYS/YER/Tablo-3_12082017.pdf sayfasından erişilmiştir.

Özçelik, D. A. (1989). *Test hazırlama kılavuzu*. Ankara: ÖSYM.

Özsoy, S., & Özsoy, G. (2013). Eğitim araştırmalarında etki büyüklüğü raporlanması. *İlköğretim Online, 12*(2), 334-346. http://dergipark.ulakbim.gov.tr/ilkonline/article/view/5000037779/5000036637 sayfasından erişilmiştir.

Partnership for 21st Century Skills [P21] (2009). *P21 framework definitions.* Retrieved from http://www.p21.org/storage/documents/P21_Framework_Definitions.pdf

Patton, M. Q. (2014). *Nitel araştırma ve değerlendirme yöntemleri (M. Bütün & S. B. Demir, Çev.)*. Ankara: PegemA Akademi.

Pillay, N., & Jugoo, V. R. (2005). An investigation into student characteristics affecting novice programming performance. *ACM Sigcse Bulletin, 37*(4), 107-110.

Pintrich, P. R., & De Groot, E. (1990). Motivational and self-regulated learning components of classroom academic performance. *Journal of Educational Psychology*, *82*(1), 33-40.

Powers, K., Ecott, S., & Hirshfield, L. M. (2007). Through the looking glass: teaching CS0 with Alice. *ACM Sigcse Bulletin, 39*(1), 213-217.

Price, K. W. (2013). *Using visual technologies in the introductory programming courses for computer science majors.* Retrieved from http://search.proquest.com/docview/916617376?accountid=11054. (916617376)

Radošević, D., Orehovački, T., & Lovrenčić, A. (2009, September). *New approaches and tools in teaching programming*. Paper presented in the 20th Central European Conference on Information and Intelligent Systems, Varazdin.

Reigeluth, C. M. (Ed.). (1987). *Instructional theories in action: Lessons illustrating selected theories and models*. Hillsdale, NJ: Lawrence Elrbaum Associates.

Resnick, M. (2013). *Learn to code, code to learn*. Retrieved from https://www.edsurge.com/news/2013-05-08-learn-to-code-code-to-learn.

Resnick, M., Flanagan, M., Kelleher, C., MacLaurin, M., Ohshima, Y., Perlin, K., & Torres, R. (2009). Growing up programming: democratizing the creation of dynamic, interactive media. *The CHI'09 Extended Abstracts on Human Factors in Computing Systems*, 3293-3296.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silverman, B. & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM, 52*(11), 60-67.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*(2), 137-172.

Rodger, S. H., Hayes, J., Lezin, G., Qin, H., Nelson, D., Tucker, R., Lopez, M., Cooper, S., Dann, W., & Slater, D. (2009). *Engaging middle school teachers and students with Alice in a diverse set of subjects.* Retrieved from https://pdfs.semanticscholar.org/e105/4034e4b439a3d75b92d0fdbb614f2fff9d3a.pdf

Rogalski, J., & Samurçay, R. (1991). Acquisition of programming knowledge and skills. In J.M. Hoc, T.R.G. Green, R. Samurcay & D.J. Gillmore (Eds.), *Psychology of programming* (pp. 157–174). London: Academic.

Salomon, G., & Perkins, D. N. (1989). Rocky roads to transfer: Rethinking mechanism of a neglected phenomenon. *Educational Psychologist, 24*(2), 113-142.

Schultz, L. A. (2011). Student perceptions of instructional tools in programming logic: A comparison of traditional versus Alice teaching environments. *Information Systems Education Journal, 9*(1), 60-67.

Scratch. (2018). Retrieved from https://scratch.mit.edu/about

Shneiderman, B., & Mayer, R. (1979). Syntactic/semantic interactions in programmer behavior: A model and experimental results. *International Journal of Computer & Information Sciences, 8*(3), 219-238.

Solmaz, E. (2014). *Programlama dili öğretiminde Alice yazılımının ders başarısı, eleştirel düşünme ve problem çözme becerileri ile üstbilişsel farkındalık düzeyine etkisi.* Doktora Tezi, Gazi Üniversitesi Eğitim Bilimleri Enstitüsü, Ankara.

Stepien, W., & Gallagher, S. (1993). Problem-based learning: As authentic as it gets. *Educational Leadership, 50*(7), 25-25.

Storey, M.-A. (2005). Theories, methods and tools in program comprehension: Past, present and future. *Proceedings 13th International Workshop on Program Comprehension*, 181-191.

Sureau, D. (2012). *History of programming languages and their evolution*. Retrieved from https://www.scriptol.com/programming/history.php

Sykes, E. R. (2007). Determining the effectiveness of the 3D Alice programming environment at the computer science I level. *Journal of Educational Computing Research, 36*(2), 223-244.

Şahin, İ., & Fındık, T. (2008). Türkiye'de mesleki ve teknik eğitim: Mevcut durum, sorunlar ve çözüm önerileri. *Türkiye Sosyal Araştırmalar Dergisi, 12*(3), 65-86.

Unuakhalu, M. F. (2008). Enhancing problem-solving capabilities using object-oriented programming language. *Journal of Educational Technology Systems, 37*(2), 121-137.

Utting, I., Cooper, S., Kölling, M., Maloney, J., & Resnick, M. (2010). Alice, Greenfoot and Scratch – A discussion. *ACM Transactions on Computing Education (TOCE), 10*(4), Article 17. http://doi.acm.org/10.1145/1868358.1868364

Ünal, E., & Çakır, H. (2016). İşbirlikli teknolojilerle desteklenen yapılandırmacı öğrenme ortamının akademik uğraşıya etkisi. *Journal of Instructional Technologies & Teacher Education, 5*(1), 13-18.

Van Haaster, K., & Hagan, D. (2004). Teaching and learning with BlueJ: an evaluation of a pedagogical tool. *Issues in Informing Science & Information Technology, 1*, 455-470.

Vihtonen, E., Alaoutinen, S., & Kaarna, A. (2001, October). *Computer supported learning environment for C programming language.* Paper presented at the 1st Baltic Sea Conference on Computer Science Educaiton, Finland.

Walberg, H. J., & Anderson, G. J. (1968). Classroom climate and individual learning. *Journal of Educational Psychology, 59*(6p1), 414-419.

Wang, T., Mei, W., Lin, S., Chiu, S., & Lin, J. M. (2009, October). *Teaching programming concepts to high school students with Alice.* Paper presented at 39th ASEE/IEEE Frontiers in Education Conference, San Antonio.

Werner, L., Denner, J., Bliesner, M., & Rex, P. (2009). Can middle-schoolers use Storytelling Alice to make games?: results of a pilot study. *The Proceedings of the 4th International Conference on foundations of digital games*, 207-214.

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: A study of twelve factors. *The ACM Sigcse Bulletin*, *33*(1), 184-188.

Yıldırım, A., & Şimşek, H. (2013). *Sosyal bilimlerde nitel araştırma yöntemleri*. Ankara: Seçkin.

Yükseköğretim Bilgi Yönetim Sistemi [YBYS]. (2018). *2017-2018 öğretim yılı yükseköğretim istatistikleri*. https://istatistik.yok.gov.tr sayfasından erişilmiştir.

Yükseköğretim Kurulu Başkanlığı [YÖK]. (2007). *Türkiye'nin yükseköğretim stratejisi*. http://yenieczak.eczak.org/wp-content/uploads/2012/06/Turkiyenin_Yuksekogretim_Stratejisi_Raporu_2007.pdf sayfasından erişilmiştir.

Zaccone, R., Cooper, S., & Dann, W. (2003, November). *Using 3D animation programming in a core engineering course seminar.* Paper presented in the 33rd ASEE/IEEE Frontiers in Education Conference, Boulder.

Zapusek, M., & Rugelj, J. (2013, October). *Applying ideas from intelligent tutoring systems for teaching programming in game based learning.* Paper presented in the 7th European Conference on Games Based Learning, Porto.

Zhang, X., De Pablos, P. O., & Zhang, Y. (2012). The relationship between incentives, explicit and tacit knowledge contribution in online engineering education project. *The International Journal of Engineering Education, 28*(6), 1341-1346.

Zhang, X., de Pablos, P. O., & Zhu, H. (2012). The impact of second life on team learning outcomes from the perspective of IT capabilities. *The International Journal of engineering Education, 28*(6), 1388-1392.

Zhang, X., Liu, L., de Pablos, P. O., & She, J. (2014). The auxiliary role of information technology in teaching: Enhancing programming course using Alice. *International Journal of Engineering Education, 30*(3), 560-565.

EKLER

EK 1. Akademik Başarı Testi

Değerli Öğrenciler,

Bu sınav *Java programlama dilini* kapsayan 48 çoktan seçmeli sorudan oluşmaktadır. Lütfen soruları mümkün olduğunca dikkatli okuyarak size göre doğru olan şıkkı işaretleyiniz. Bilmediğiniz soruyu boş bırakabilirsiniz. Tahmini cevaplama süresi 55 dk'dır. Katılımınız için teşekkür ederiz.

Ceren BAŞTEMUR KAYA

Danışman : Doç. Dr. Hasan ÇAKIR

SORULAR

1.

```
public static void main(String[] args) {
byte  a;
float b;
int   c;
}
```

Yukarıda verilen Java programında a,b ve c olmak üzere 3 değişken tanımlanmıştır. Aşağıdakilerden hangisinde bu değişenlere uygun değer ataması yapılmıştır?

A. a=1.1 b=2.5 c=3.7

B. a=2 b=3 c=4

C. a=1 b=2.2 c=4

D. a=1 b=5.7 c=3.3

E. a=4 b=3 c=3.1

2.

```
public static void main(String[] args) {
int sayi1=45;
double sayi2=12.8;
byte sayi3=1088;
byte sayi4=45;
double sayi5=3.14;

}
```

Yan tarafta verilen Java kodundaki hatayı düzeltmek için aşağıdaki işlemlerden hangisi yapılmalıdır?

A. *sayi1* değişkeninin türü *float* yapılmalıdır.

B. *sayi2* değişkeninin türü *int* yapılmalıdır.

C. *sayi3* değişkeninin türü *int* yapılmalıdır.

D. *sayi4* değişkeninin türü *int* yapılmalıdır.

E. *sayi5* değişkeninin türü *byte* yapılmalıdır.

3. I. % II. + III. < IV. > V. <=

Yukarıda verilen ifadelerden kaç tanesi Java programlama diline göre mantıksal operatördür?

A. 1 B. 2 C. 3 D. 4 E.5

4.
```
public static void main(String[] args) {
int x=5;
int y=10;
y--;
x++;
x-=y;
}
```

Yan tarafta verilen Java koduna göre x değişkeninin son değeri aşağıdakilerden hangisidir?

A. -1 B. -2 C. -3 D.-4 E.-5

5.
```
public static void main(String[] args) {
int x=5;
int y=10;
y--;
x++;
x-=y;
++y;
}
```

Yan tarafta verilen Java kodunda aşağıdaki atama işlemlerinden hangisi yapılmamıştır?

A.x=x-1 B. y=y-1 C. x=x+1 D. x=x-y E. y=y+1

6. Bir dikdörtgenin kısa ve uzun kenarını değişken olarak tanımlayarak, dikdörtgenin çevresini ve alanını hesaplayan Java kodu aşağıdakilerden hangisidir? (Örnek için kısa kenar 3 ve uzun kenar 6 alınacaktır)

A.
```
public static void main(String[] args)
int kisakenar =3;
int uzunkenar= 6;
}
```

B.
```
public static void main(String[] args) {
int kisakenar =3;
int uzunkenar= 6;
int cevre;
int alan;
}
```

C.
```
public static void main(String[] args
int kisakenar =3;
int uzunkenar= 6;
int cevre;
int alan;
cevre=2*(kisakenar+uzunkenar);
}
```

D.
```
public static void main(String[] args) {
int kisakenar =3;
int uzunkenar= 6;
int cevre;
int alan;
cevre=2*(kisakenar+uzunkenar);
alan=kisakenar*uzunkenar;
}
```

E.
```
public static void main(String[] args) {
int kisakenar =3;
int uzunkenar= 6;
int cevre;
int alan;
alan=kisakenar*uzunkenar;
}
```

7. Char ve Byte hangi tür değişkenler için tanımlanmaktadır?

   A. Metinsel - Tamsayı

   B. Tamsayı - Metinsel

   C. Ondalıklı - Metinsel

   D. Boolean - Metinsel

   E. Karakter - Tamsayı

8. Java programlama diline göre aşağıda verilen atamalardan hangisinin kullanımı yanlıştır?

   A. x++ B. --x C. x-- D. x** E. x*=x

9. switch-case yapısıyla ilgili aşağıdaki ifadelerden hangisi doğrudur?

   A. Bir kontrol yapısıdır.

   B. Bir döngü yapısıdır.

   C. Bir dizi türüdür.

   D. Bir metot türüdür.

   E. Bir arayüz oluşturma yöntemidir.

10. Java programlama diline göre kontrol yapıları ile ilgili aşağıda verilen ifadelerden hangisi yanlıştır?

    A. Şartlı ifadelerde kullanılmaktadır.

    B. Her zaman döngülerle birlikte kullanılmaktadır.

    C. if/else kullanılan şart yapılarından biridir.

    D. İkiden daha fazla şart olması durumunda if/else if/else yapısı kullanılmaktadır.

    E. Kontrol yapılarında switch yapısı da kullanılmaktadır.

11. Eğer A sayısı B' den büyük ise "A sayısı büyüktür" aksi halde "A sayısı küçüktür" ekran çıktısını veren Java kodu aşağıdakilerden hangisidir?

A.
```
int A=5;
int B=3;
```

B.
```
System.out.println("A Sayısı Büyüktür");
```

C.
```
System.out.println("A Sayısı Büyüktür");
System.out.println("A Sayısı Küçüktür");
```

D.
```
if (A>B) {
System.out.println("A Sayısı Büyüktür");
}
```

E.
```
if (A>B) {
System.out.println("A Sayısı Büyüktür");
}
else {
System.out.println("A Sayısı Küçüktür");
}
```

12. Java programlama diline göre aşağıda verilen kontrol yapılarından hangisinin kullanımı yanlıştır?

A.
```
if (A<B)
System.out.println("A Sayısı Büyüktür");
else if
System.out.println("A Sayısı Küçüktür");
```

B.
```
if (A>B) A++;
```

C.
```
if (A>B)
System.out.println("A Sayısı Büyüktü
```

D.
```
if (A>B) {
System.out.println("A Sayısı Büyüktür");
}
else {
System.out.println("A Sayısı Küçüktür");
}
```

E.
```
switch(A){
case 1:
      System.out.println("A sayısı birdir");
      break;
case 2:
      System.out.println("A sayısı ikidir");
      break;
case 3:
      System.out.println("A sayısı üçtür");
      break;
default:
      System.out.println("A sayısı bir iki veya
}
```

13.
```
int A=2;
int B=3;

switch(A){
case 1:
      B++;
      break;
case 2:
      B=4;
      break;
case 3:
      B=A;
      break;
default:
      --B;
}
```

Yan tarafta verilen Java programına göre B değişkeninin son değeri nedir?

A. 2 B. 3 C. 4 D. 5 E. 6

14.
```
switch(A){
case 1:
      B++;
      break;
case 2:
      B=A;
      break;
default:
      --B;
}
```

Yan tarafta Java programlama dilinde switch-case kullanılarak bir kod bloğu oluşturulmuştur. Verilen program parçasının if-else kullanılarak doğru yazılmış hali aşağıdakilerden hangisidir?

A.
```
if (A==1)       B++;
else if (A==2)  B=A;
else            --B;
```

B.
```
if (B==1)       B++;
else if (B==2)  B=A;
else            --B;
```

C.
```
if (A=1) {
    B++;
}
else if (A=2){
    B==A;

}
else {
    B--;
}
```

D.
```
if (A==1)       A++;
else if (A==2)  B--;
else            B=A;
```

E.
```
if (A==1)
    B++;
    B=A;
    --B;
```

15. Eğer A sayısının bir fazlası, B ve C sayılarının çarpımından büyük ise, ekrana "BÜYÜK", değilse ekrana "KÜÇÜK" yazdıran Java kodu aşağıdakilerden hangisidir?

A.
```
if (A>B+C)
    System.out.println("BÜYÜK");
else
    System.out.println("KÜÇÜK");
```

B.
```
if (A++>B+C)
    System.out.println("BÜYÜK");
else
    System.out.println("KÜÇÜK");
```

C.
```
if (A++>B*C)
    System.out.println("BÜYÜK");
    System.out.println("KÜÇÜK");
```

D.
```
if (A>B*C) {
    System.out.println("BÜYÜK");
    System.out.println("KÜÇÜK");
    A=A+1;
}
```

E.
```
if (++A>B*C)
    System.out.println("BÜYÜK");
else
    System.out.println("KÜÇÜK");
```

16.
```
if (A>B) (A>C)
System.out.println("A En Büyük Sayıdır");
```
Yan tarafta verilen Java kodunda bulunan hatayı gidermek için aşağıdakilerden hangisi yapılabilir?

A. System.out.println komutu silinmelidir.

B. if yerine if else yazılmalıdır.

C. (A>B) ile (A>C) arasına mantıksal operatör konulmalıdır.

D. Verilen kodlar { } içerisine yazılmalıdır.

E. Ekran çıktısı "B En Büyük Sayıdır" şeklinde düzeltilmelidir.

17.
```
int A=3;
int B=4;
if (A>B) { A=--B;
    System.out.println(A);
}
else
    System.out.println(B);
```
Yan tarafta verilen Java programının ekran çıktısı aşağıdakilerden hangisidir

A. 2 B. 3 C. 4 D. 5 E. 6

18. 1'den 100'e kadar olan sayıları döngü kullanarak toplayan Java kodu aşağıdakilerden hangisidir?

A.
```
public static void main(String[] args) {
      int Toplam;
      int i=0;
      Toplam=i*100;
}
```

B.
```
public static void main(String[] args) {
      int Toplam;
      int i=0;
      for (i=100;i>1;i--) {
          Toplam=i;

      }
}
```

C.
```
public static void main(String[] args) {
      int Toplam=0;
      int i;
      for (i=1;i<=100;i++) {
          Toplam=Toplam+i;

      }
      System.out.println(Toplam);
}
```

D.
```
public static void main(String[] args) {
      int Toplam=0;
      int i;
      for (i=0;i<=100;i=i+2) {
          Toplam++;

      }
}
```

E.
```
public static void main(String[] args) {
      int Toplam=0;
      int i;
      for (i=0;i<=100;i++) {

      }
}
```

19.
```
for (i=1;i<=Faktoriyel;i++) {
    Faktoriyel=i;
}
```
Yan tarafta verilen Java kodunda bir sayının faktöriyeli hesaplanmak istenmektedir. Doğru sonuç alınabilmesi için aşağıdaki değişikliklerden hangisi yapılmalıdır?

A. Döngü i=0 ile başlamalıdır.

B. i++ yerine i-- kullanılmalıdır.

C. { } parantezi kaldırılmalıdır.

D. Faktoriyel = Faktoriyel*i; şeklinde değiştirilmelidir.

E. <= mantıksal operatörünün yerine == kullanılmalıdır.

20. *do-while* yapısıyla ilgili aşağıdaki ifadelerden hangisi doğrudur?

A. Bir dizi türüdür.

B. Bir metot türüdür.

C. Bir döngü yapısıdır.

D. Bir arayüz oluşturma yöntemidir.

E. Bir kontrol yapısıdır.

21. Java programlama diline göre döngülerle ilgili aşağıda verilen ifadelerden hangisi yanlıştır?

A. Tekrar gereken durumlarda kullanılır.

B. Döngü içerisinde kullanılan kod sayısı birden fazla ise { } bloğu içerisine alınır.

C. Farklı döngü deyimleri bulunmaktadır.

D. İç içe döngülerde, farklı döngü deyimlerinin kullanılması gerekmektedir.

E. Do döngüsü diğerlerine göre en az bir kere çalışır.

22.
```
public static void main(String[] args) {
        int A;
        int i;
        for (i=0;i<=5;i++) {
            A=i;
        System.out.print(A);
        }
}
```
Yan tarafta verilen Java kodunun ekran çıktısı aşağıdakilerden hangisidir?

A.012345 B. 0123456 C. 12345 D. 1234 E. 123

23.
```
public static void main(String[] args) {
        int A;
        int i;
        for (i=0;i<5;i++) {
             if (i==1) A=i;
             else if (i==2) A=i*2;
             else A=i*0;
        }
}
```
Yan tarafta verilen Java koduna göre A değişkeninin son değeri aşağıdakilerden hangisidir?

A. 0 B. 1 C. 2 D. 3 E. 4

24.
```
while(sayi < 10) {
   sonuc = sonuc * sayi;
   sayi++;
}
```
Yan tarafta verilen Java kodunun *for* döngüsü kullanılarak doğru yazılmış hali aşağıdakilerden hangisidir?

A.
```
for (i=1;i<=10;--i)
    sonuc = sonuc + i;
```

B.
```
for (i=0;i<10;i=++i+2)
    sonuc = sonuc++;
```

C.
```
for (i=1;i<10;i++)
    sonuc = sonuc * i;
```

D.
```
for (i=0;i<10;i=++i)
    sonuc++;
```

E.
```
for (i=0;i<10;i=i+2)
    sonuc = sonuc - i;
```

25. Dizi kavramının tanımı aşağıdakilerden hangisidir?

A. Java programlama dilinde sınıflara verilen genel addır.

B. Kalıtım uygulanmış değişkenlerin genel adıdır.

C. Çok fazla veriyi bir arada tutan değişken türüdür.

D. GUI programlamadır.

E. Kontrol yapıları için tanımlanmış ve sadece String değer alabilen değişken türüdür.

26. 3 ile 15 arasındaki sayılarla ilgili işlem yapmak amacıyla oluşturulan döngü kodu aşağıdakilerden hangisidir?

A. for (i=1; i<=3; i=i+15) { }

B. for (i=0; i<=15; i++) { }

C. for (i=3; i<=15; i--) { }

D. for (i=3; i<=15; i=i+1) { }

E. for (i=15; i<=3; i++) { }

27. Java programlama diline göre, en az bir kere döngünün çalışmasının istenildiği durumda aşağıdaki yapılardan hangisi kullanılmalıdır?

A. For Döngüsü

B. Do Döngüsü

C. While Döngüsü

D. Foreach Döngüsü

E. Switch Döngüsü

28.
```
int notlar[ ]=new int[5];
notlar[1]=25;
notlar[2]=80;
notlar[3]=45;
notlar[4]=100;
notlar[5]=65;
```

Yan tarafta verilen Java programı hatalı çalışmaktadır. Hatanın çözülmesi için aşağıdaki işlemlerden hangisi yapılmalıdır?

A. notlar[5] yerine notlar[0] yazılmalıdır.

B. Değişken tipi olarak int yerine byte kullanılmalıdır.

C. new kaldırılmalıdır.

D. int[5] yerine int[0] yazılmalıdır.

E. Dizinin elaman sayısı 10 olarak değiştirilmelidir.

29.
```
byte vize[ ]=new byte[5];
vize[0]=25;
vize[1]=80;
vize[2]=45;
vize[3]=100;
vize[4]=65;
```

Yan tarafta verilen dizilerle ilgili kod bloğuyla ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

A. 5 elemanlıdır.

B. Sadece tamsayı türünde değişkenler almaktadır.

C. Dizinin son indisi 25'dir.

D. Dizinin sonuncu elemanının değeri 65'dir.

E. Dizinin ismi *vize*'dir.

30.
```
int[ ][ ] sihirliKare={
                       {16, 3, 2, 13},
                       {5, 10, 11, 8},
                       {9, 6, 7, 12},
                       {4, 15, 14, 1}
                       };
```

Yan tarafta verilen dizilerle ilgili kod bloğuyla ilgili olarak aşağıdaki ifadelerden hangisi doğrudur?

A. Tek boyutlu dizi tanımlanmıştır.

B. Dizi ondalıklı sayılardan oluşmaktadır.

C. Dizinin ismi Kare'dir

D. sihirliKare[1][1] 10 değerine karşılık gelmektedir.

E. Dizi 3 satır ve 3 sütundan oluşmaktadır.

31. Aşağıda verilen Java kodlarının hangisinde 5 elemanlı ve tek boyutlu bir dizi tanımlanmıştır?

A.
```
int [ ] x= new int [6] ;
```

B.
```
int[] dizi;
dizi = new int[5];
```

C.
```
int[ ] AsalSayilar={ 2, 3, 5, 7, 11, 13 };
```

D.
```
int[] diziKare = {4, 5, 6, 7, 8, 9};
```

E.
```
int[] dizi;
```

32. Java programlama diline göre metotlarla ilgili verilen aşağıdaki ifadelerin hangisi yanlıştır?

A. Parametre değeri almayan metotlarda bulunmaktadır.

B. Eğer metodun geri dönüş tipi yoksa void olarak tanımlanmalıdır.

C. Tanımlanırken metot ismi belirtilmelidir.

D. Metotların geri dönüş tipinin tanımlanması gerekmektedir.

E. Metotlar parametre almak zorundadır.

33.
```
public int Toplama(int a,int b){
    int sonuc;
    sonuc=a+b;
    return sonuc;

}
```

Yan tarafta verilen Java programı ile ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

A. Bir metot tanımlanmıştır.

B. *int* türünde iki değişkeni giriş olarak almaktadır.

C. Girilen iki sayının toplamı yapılmaktadır.

D. *sonuc* değişkeni global bir değişkendir.

E. Metot *int* türünde bir çıkış vermektedir.

34. Aşağıda verilen Java kodlarındaki metotlar gruplandırıldığında hangisi diğerlerinden farklıdır?

A. public int Toplama(int a, int b)

B. public int Cikarma(int a,int b, int c)

C. public int EnBuyuk(int x, int y)

D. public int Ortalama(int z, int w)

E. public int EnKucuk(int k, int m)

35.
```
public int x (int a){
   int sonuc=1;
   int i;
   for (i=1; i<=a;i++) sonuc=sonuc*i;
   return sonuc;

}
```

Yan tarafta verilen metot kodu ile aşağıdaki hangi işlem yapılmaktadır?

A. Girilen sayının asal sayı olup olmadığı tespit edilmektedir.

B. Girilen sayının faktöriyeli alınmaktadır.

C. Girilen sayının karesi alınmaktadır.

D. Girilen sayı kendisiyle çarpılmaktadır.

E. 1 ile 10 arasındaki sayılar toplanmaktadır.

36. Java programla diline göre, iki sayının çarpımını veren metot kodu aşağıdakilerin hangisinde doğru verilmiştir?

A.
```
int a;
int b;
int sonuc;
sonuc=a*b;
```

B.
```
public int x (int a){
   int sonuc=1;
   int i;
   for (i=1; i<=a;i++) sonuc=sonuc*i;
   return sonuc;

}
```

C.
```
public int x (int a, int b){
  return a;
}
```

D.
```
public int x (int a, int b){
  int sonuc;
  sonuc=a*b;
  return sonuc;
}
```

E.
```
public int x (int a){
  int carpim;
  carpim=a*a;
  return carpim;
}
```

37.
```
public void CarpimTablosu (){
  int Sonuc;
  for (int i=1; i<=10; i++)
      for (int y=1; y<=10; y++) {
                                  //Line 10

          System.out.println(i+"*"+y+"="+Sonuc);
      }

}
```

Yan taraftaki Java programında çarpım tablosu oluşturulmak istenmektedir. Çarpım tablosunun çalışabilmesi için *Line 10* ile gösterilen yere gelecek uygun kodu yazınız?

A. Sonuc=i*y;

B. Sonuc=y;

C. Sonuc++;

D. Sonuc=i+y;

E. Sonuc=i-y;

38. Java programlama dilinde *sınıf* kavramının tanımı aşağıdakilerden hangisidir?

A. Metotların genel ismidir.

B. Üretilecek nesnelerin özelikleri ve davranışlarının belirlendiği kod bloğudur.

C. Farklı nesnelerin, aynı mesaja (olaya ya da uyarıma) farklı şekillerde cevap verebilme yeteneğidir.

D. Farklı nesnelerin bir arada kullanılmasıdır.

E. GUI programlamadır.

39. Java programlama dilinde *sınıflarda çok biçimlilik* kavramının tanımı aşağıdakilerden hangisidir?

A. Farklı nesnelerin bir arada kullanılmasıdır.

B. Bir soyutlama türüdür.

C. Bir sınıfta tanımlanmış değişkenlerin ve/veya metotların yeniden tanımlanmasına gerek olmaksızın yeni bir sınıfa taşınabilmesidir.

D. Ana sınıftan başka bir sınıf türetildiğinde, türeyen sınıfın ana sınıf içindeki üyeleri kendine göre farklı olarak uygulama şeklidir.

E. Bir sınıfa ait kodların başka bir sınıf tarafından kullanılmasının engellenmesidir.

40.
```
public  Carpma (int a, int b){
    int sonuc;
    sonuc=a+b;
    return sonuc;
}
```

Yan tarafta verilen Java kodunda bulunan hatanın giderilmesi için aşağıdakilerden hangisi yapılmalıdır?

A. a değişkeninin türü double yapılmalıdır.

B. return komutu kaldırılmalıdır.

C. Metot isminden önce uygun değişken tipi belirtilmelidir.

D. public deyimi silinmelidir.

E. sonuc değişkeninin değeri doğru hesaplanmalıdır.

41. Girilen iki değişkenin çarpımı için *Carpım* adında, 2 değişkenin çarpımından oluşan bir metot oluşturulmuştur. Bu metot kullanıldığında aşağıdaki Java kodlarından hangisinin sonucu farklı bir değer çıkar?

A. Carpim(2,24); B. Carpim(12,4); C. Carpim(1,24); D. Carpim(48,1);E. Carpim(6,8);

42. Java programlama diline göre aşağıdaki ifadelerden hangisi yanlıştır?

A. Bir sınıftan yeni bir nesne oluşturulduğunda otomatik olarak çalışan alanlara yapılandırıcı adı verilir.

B. Bir sınıf içerisinde birden fazla yapılandırıcı oluşturulamaz.

C. This komutu sınıf içerisinde kullanılıyorsa, o nesneyi ifade edebilmek için kullanılır.

D. Super komutu ile, kalıtım almış sınıflardan ana sınıfa ait bir metodu işletebilirsiniz.

E. Super komutunun kullanılabilmesi için kalıtım alınmış olması gerekmektedir.

```java
import java.util.Scanner;
public class Ogrenci {

    public  double OrtHesapla (int vize, int finalnotu){
       double ort;
       ort=vize*0.4+finalnotu*0.6;
        return ort;
    }
    public  String HarfNotuHesapla (int vize, int finalnotu){
        double ort;
        String HarfNotu;
        ort=this.OrtHesapla(vize, finalnotu);
        if (ort>=85) HarfNotu="AA";
        else if (ort>=75) HarfNotu="BA";
        else if (ort>=65) HarfNotu="BB";
        else if (ort>=57) HarfNotu="CB";
        else if (ort>=50) HarfNotu="CC";
        else HarfNotu="FF";
        return HarfNotu;
     }

    public static void main(String[] args) {
         Scanner klavye=new Scanner(System.in);
         System.out.println("Vize Notu Giriniz:");
         int vize=klavye.nextInt();
         System.out.println("Final Notu Giriniz:");
         int finalnotu=klavye.nextInt();
         Ogrenci OgrenciBilgileri=new Ogrenci();
         double ort;
         ort=OgrenciBilgileri.OrtHesapla(vize, finalnotu);
         String hf=OgrenciBilgileri.HarfNotuHesapla(vize, finalnotu);
         System.out.println("Ortlama=:"+ort);
         System.out.println("Harf Notu=:"+hf);

    }

}
```

(43 ve 46 arasındaki soruları yukarıda verilen Java programına göre cevaplayınız)

43. Verilen Java programı ile ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

    A. Ogrenci isminde bir sınıf oluşturulmuştur.

    B. Kalıtım uygulanmıştır.

    C. Öğrenci sınıfına ait metotlar tanımlanmıştır.

    D. this deyimi kullanılmıştır.

    E. Sınıfa ait özellik bulunmamaktadır.

44. Verilen Java programı ile ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

A. Ogrenci sınıfının kullanımı için Ogrenci Bilgileri isimli bir nesne oluşturulmuştur.

B. Ogrenci Bilgileri nesnesinin tüm özellikleri kullanılmıştır.

C. Programda Ogrenci Bilgileri nesnesinin dışında farklı bir nesne daha kullanılmıştır.

D. Ogrenci sınıfına ait HarfNotuHesapla metodunun oluşturulması sırasında, farklı bir metottan yararlanılmamıştır.

E. vize ve final notunun girişi için klavye nesnesi kullanılmıştır.

45. hf=OgrenciBilgileri.HarfNotuHesapla(50,70) çalıştırıldığında hf değişkeni aşağıdaki değerlerden hangisini alır?

A. AA B. BA C. BB D. CB E. CC

46. *Dikdortgen* sınıfına ait nesne kodu aşağıdakilerden hangisinde doğru tanımlanmıştır?

A. Dikdortgen DikdortgenNesnesi = new Dikdortgen();

B. new Dikdortgen();

C. Dikdortgen DikdortgenNesnesi;

D. Dikdortgen DikdortgenNesnesi = new ();

E. Dikdortgen new() = new (DikdortgenNesnesi);

47. Bir karenin kenar değerini kullanarak, karenin alan ve çevresini hesaplayan sınıf kodları aşağıdaki ifadelerin hangisinde doğru verilmiştir?

A.
```
public class Kare {
    double kenar;
    double cevre;
    double alan;
}
```

B.
```
public class Kare {
    double kenar;

    public  double Alan(double kenar){
        double alan;
        alan=kenar*kenar;
        return alan;
    }
}
```

C.
```
public class Kare {
    double kenar;
    double cevre;
    double alan;

    public  double Alan(double kenar){
        double alan;
        alan=kenar*kenar;
        return alan;
    }
}
```

D.
```
public class Kare {
    double kenar;
}
```

E.
```
public class Kare {
    double kenar;

    public  double Alan(double kenar){
        double alan;
        alan=kenar*kenar;
        return alan;
    }

    public  double Cevre(double kenar){
        double cevre;
        cevre=4*kenar;
        return cevre;
    }
}
```

48.
```
public class Daire {

    public  double Alan (double yaricap){
        return (Math.PI*yaricap*yaricap);
    }

    public  double Cevre (double yaricap){
        return (2*Math.PI*yaricap);
    }

}
```

Yan tarafta verilen Java programıyla ilgili olarak aşağıdaki ifadelerin hangisi yanlıştır?

A. Daire isminde bir sınıf tanımlanmıştır.

B. Sınıf içerisinde üç tane metot bulunmaktadır.

C. Daire sınıfından A nesnesi oluşturulduğunda A.Cevre ile dairenin çevresi hesaplanır.

D. Daire sınıfından B nesnesi oluşturulduğunda B.Alan ile dairenin alanı hesaplanır.

E. İç içe sınıf kullanılmamıştır.

EK 2. Akademik Başarı Testi Belirtke Tablosu

	KONU	KAZANIM	BİLİŞSEL HEDEF					
			BİLGİ	KAVRAMA	UYGULAMA	ANALİZ	SENTEZ	DEĞERLENDİRME
1	Veri tipleri ve değişkenler, operatörler	Farklı veri tiplerini kullanarak değişken tanımlar	7	1				2
		Operatörleri kullanarak atama işlemi yapar	3	8		4, 5	6	
2	Kontrol yapıları	Kontrol yapılarının kullanımı kavrar	9	10, 27				
		Kontrol yapıları kullanarak program parçaları oluşturur			11	12, 13, 17	15	14, 16
3	Döngüler	Döngü yapılarının kullanımını kavrar	20	21				
		Problem çözümünde ihtiyaç duyulan döngü yapısını oluşturur			26	22, 23	18	19, 24
4	Diziler	Dizi türündeki değişkenleri kullanır	25			29		28
		Tek boyutlu ve çok boyutlu dizi tanımlar			31	30		
5	Metot kullanımı	Metotların kullanım amacını kavrar		32				
		Amaca göre uygun metot yapısı oluşturur			37, 41	33, 34, 35	36	40
6	Sınıf yapıları ve kalıtım	Sınıf yapısının kullanım amacını bilir	38, 39	42				
		Amaca göre sınıf tanımlar					47	43, 44, 48
		Sınıf yapısının özelliklerini nesneye tanımlar			46	45		

EK 3. Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testi

Değerli Öğrenciler,

Bu sınav genel programlama bilgisini kapsayan 32 çoktan seçmeli sorudan oluşmaktadır. Lütfen soruları mümkün olduğunca dikkatli okuyarak size göre doğru olan şıkkı işaretleyiniz. Bilmediğiniz soruyu boş bırakabilirsiniz. Tahmini cevaplama süresi 35 dk'dır. Katılımınız için teşekkür ederiz.

Ceren BAŞTEMUR KAYA

Danışman : Doç. Dr. Hasan ÇAKIR

SORULAR

1. A değişkenin değeri 8'dir. Aşağıda verilen atama işlemlerinin hangisinin yapılması durumunda A değişkenin son değeri 4 olur?

   A. A++ B. A=A/2 C. A-- D. A=A*2 E. ++A

2. Byte ve String hangi tür değişkenler için tanımlanmaktadır?

   A. Metinsel - Tamsayı

   B. Tamsayı - Metinsel

   C. Ondalıklı - Metinsel

   D. Boolean - Metinsel

   E. Karakter - Tamsayı

3.
```
int A=5;
int B=3;
A++;
B++;
A=(B*2)+A;
```

Yan tarafta verilen koda göre A değişkeninin son değeri nedir?

A. 4 B. 6 C. 8 D. 14 E. 15

4.
```
int A=5;
int B=3;
String C=MYO;
double D=1.2;
byte E=0;
```

Yan tarafta verilen değişkenlerle ilgili kodda bulunan hatanın giderilmesi için aşağıdakilerden hangisi yapılmalıdır?

A. A değişkeninin türü *float* yapılmalıdır.

B. B değişkeninin türü *byte* yapılmalıdır.

C. C değişkeninin değeri " " içerisine alınmalıdır.

D. D değişkeninin türü *int* yapılmalıdır.

E. E değişkeninin türü int yapılmalıdır.

5. Aşağıdaki kodlardan hangisinde bir üçgenin üç kenarını değişken olarak tanımlanıp, üçgenin çevresi hesaplanmıştır? (Örnek için kenarlar sırasıyla 3, 4, 5 alınacaktır)

A.
```
double kenar_1;
double kenar_2;
double kenar_3;
double cevre=5;
cevre++;
```

B.
```
double kenar_1=3;
double kenar_2=4;
double kenar_3=5;
double cevre;
```

C.
```
double kenar_1=3;
double kenar_2=4;
double kenar_3=5;
double cevre;
cevre=kenar_1+kenar_2+kenar_3;
```

D.
```
double kenar_1;
double kenar_2;
double kenar_3;
double cevre;
```

E.
```
double kenar_1;
double kenar_2;
double kenar_3;
```

6. I. + II. - III.< IV. == V. *

Yukarıda verilenlerden ifadelerden kaç tanesi aritmetik operatördür?

A. 1 B. 2 C. 3 D. 4 E. 5

7.
```
int x,y,z;
double q, k;
```
Yan tarafta verilen koda göre aşağıdaki ifadelerden hangisi kesinlikle yanlıştır?

A. İki farklı değişken türü bulunmaktadır.

B. Metinsel bir değişkende vardır.

C. Toplam 5 değişken tanımlanmıştır.

D. q ondalıklı bir değere sahiptir.

E. x değişkenin değeri 5 olabilir.

8. Atama ile ilgili aşağıdaki eşleştirmelerden hangisi yanlıştır?

A. x++ = x=x+1

B. --x = x=-x-1

C. x+=2 = x=x+2

D. x-=2 = x=x-2

E. x*=2 = x=x*2

9.
```
int x=5;
int y=2;
if (x>y) y*=y;
else --y;
```
Yan tarafta verilen koda göre y değişkeninin son değeri nedir?

A. 1 B. 2 C. 3 D. 4 E. 5

10.
```
int a=5;
int b=2;
int c=3;
if (a>b>c)  c=a+b;
```
Yan tarafta verilen koddaki hatanın düzeltilmesi için aşağıdakilerden hangisi yapılmalıdır?

A. if yerine switch kullanılmalıdır.

B. a, b ve c değişkenlerinin türü değiştirilmelidir.

C. (a>b>c) ifadesi mantıksal operatörler kullanılarak düzeltilmelidir.

D. if kontrol yapısı { } içerisine alınmalıdır.

E. if yerine if-else yazılmalıdır.

11. Eğer A sayısının karesi B sayısından büyükse, B sayısının değerini iki katına çıkaran kod aşağıdakilerden hangisidir?

A. if (A>B) B++;

B. if ((A*2)>B) B=B*2;

C. if ((A*A)>B) B=B*2;

D. if ((A*A)>(B*B)) B=B+2;

E. (A>B) B++;

12.
```
int z=5;
int x=10;
switch (z) {
case 1: x=+z; break;
case 2: x++; break;
case 3: x=x*z; break;
case 4: --x; break;
default: x=0;
}
```

Yan tarafta verilen programa göre x değişkeninin son değeri aşağıdakilerden hangisidir?

A. 0 B. 9 C. 11 D. 15 E. 50

13. if-else yapısı ile ilgili aşağıdaki ifadelerden hangisi doğrudur?

A. Bir döngü yapısıdır.

B. Bir dizi türüdür.

C. Bir kontrol yapısıdır.

D. Bir metot türüdür.

E. Bir arayüz oluşturma yöntemidir.

14. I. if-else II.switch-case III.for IV.while V.do while

Yukarıdaki ifadelerden kaç tanesi kontrol yapılarındandır?

A. 1 B. 2 C. 3 D. 4 E. 5

15.
```
int x=2;
int y=3;
if (x>y)
    System.out.println(x);
else
    System.out.println((y));
```

Yan tarafta verilen programının ekran çıktısı aşağıdakilerden hangisidir?

A. 1 B. 2 C. 3 D. 4 E. 5

16.
```
switch (y) {
case 1: x=+y; break;
case 2: x++; break;
default: x=0;
}
```

Yan tarafta verilen kod bloğunda switch-case ile bir kontrol yapısı oluşturulmuştur. Kod bloğunun if-else kullanılarak yazılmış hali aşağıdakilerden hangisidir?

A.
```
if (y==1) x++;
else if (y==2) x=x+y;
```

B.
```
if (x==1) x=+y;
else if (x==2) x++;
else x=0;
```

C.
```
if (y==1) x++;
else if (y==2) x=x+y;
else x=0;
```

D.
```
if (y==1) x=+y;
else if (y==2) x++;
else x=0;
```

E.
```
if (y==x) x++;
else x=0;
```

17. x değişkeninin değeri sıfırdan büyük ise "POZİTİF", sıfırdan küçükse "NEGATİF", sıfıra eşitse "SIFIR" yazdıran program aşağıdakilerden hangisidir?

A.
```
if (x>0)
    System.out.println("POZİTİF");
else if (x<0)
    System.out.println("NEGATİF");
else
    System.out.println("SIFIR");
```

B.
```
System.out.println("POZİTİF");
System.out.println("NEGATİF");
System.out.println("SIFIR");
```

C.
```
if (x>0)
    System.out.println("POZİTİF");
else if (x<0)
    System.out.println("NEGATİF");
```

D.
```
if (x==0)
    System.out.println("POZİTİF");
```

E.
```
if (x==0)
    System.out.println("POZİTİF");
else if (x<1)
    System.out.println("NEGATİF");
else
    System.out.println("SIFIR");
```

18.
```
if (ort>=85)
    System.out.println("AA");
else if (ort>=75)
    System.out.println("BA");
else if (ort>=65)
    System.out.println("BB");
else if (ort>=57)
    System.out.println("CB");
else
    System.out.println("CC");
```

Yanda verilen kod bloğunun ekran çıktısının CB olması için *ort* değişkeninin değeri aşağıdakilerden hangisi olabilir?

A. 90 B. 76 C. 68 D.59 E.50

19.
```
int a=15;
if (a>15)
    System.out.println("Üçgen");
else if (a>10)
    System.out.println("Kare");
else if (a>5)
    System.out.println("Daire");
else if (a>1)
    System.out.println("Silindir");
else
    System.out.println("Küp");
```

Yan tarafta verilen programın ekran çıktısı aşağıdakilerden hangisidir?

A. Üçgen

B. Kare

C.Daire

D. Silindir

E. Küp

20. I. for II.if III.while IV.switch V.do while

Yukarıda verilen ifadelerden kaç tanesi döngülerle ilgilidir?

A. 1 B. 2 C. 3 D. 4 E. 5

21.
```
int T=15;
for (int i=0;i<=10;i++)
    T=i;
```

Yan tarafta verilen kod bloğunda T değişkeninin son değeri nedir?

A. 0 B. 9 C. 10 D. 11 E. 15

22.
```
for (int i=8;i<=10;i++)
    System.out.print("E");
```

Yan tarafta verilen kod bloğunun ekran çıktısı aşağıdakilerden hangisidir?

A. EE B. EEE C. EEEE D. EEEEE E. EEEEEE

23.
```
for (int i=0;i<=SonDeger;i++)
    System.out.print("A");
```

Yan tarafta verilen kod bloğunun ekran çıktısının AAAAAA olabilmesi için *SonDeger* değişkeninin değeri aşağıdakilerden hangisi olmalıdır?

A. 3 B. 4 C. 5 D. 6 E. 7

24.
```
int Toplam=5;
for (int i=1;i<=100;i++)
    Toplam=Toplam+i;
```

Yan tarafta verilen kodda 1 ile 100 arasındaki sayılar toplanmak istenmektedir. Doğru sonuç alınabilmesi için aşağıdaki değişikliklerden hangisi yapılmalıdır?

A. Toplam=Toplam+i yerine Toplam=i yazılmalıdır.

B. Toplam=5 yerine Toplam=0 yazılmalıdır.

C. *int i* yerine *byte i* yazılmalıdır.

D. i<=100 yerine i>=100 yazılmalıdır.

E. i++ yerine i-- yazılmalıdır.

25. 50 ile 100 arasındaki sayılarla ilgili işlem yapmak istenmektedir. Bunun için oluşturulacak döngü aşağıdakilerden hangisidir?

A. for (i=0;i<=50;i++) {}

B. for (i=50;i<=100;i++) {}

C. for (i>50;i=100;i++) {}

D. for (i<50;i=100;i++) {}

E. for (i==50;i==100;i--) {}

26. Bir döngü içerisinde tekrar gerektiren işlemler yapılmaktadır. Program işleyişinde döngünün çalışabilmesi için, değişkenin başlangıç ve bitiş şartının buna uygun olması gerekmektedir. Aşağıda verilen döngü yapılarının hangisinde başlangıç ve bitiş şartına bakmaksızın döngü en az bir kere çalışır?

A. For Döngüsü

B. While Döngüsü

C. Foreach Döngüsü

D. Switch Döngüsü

E. Do Döngüsü

27.
```
int T=0;
for (i=100;i<=500;i++)
    if (i%2==1) T+=i;
```

Yan tarafta verilen program ile yapılmak istenen işlem aşağıdakilerden hangisidir?

A. Faktöriyel hesaplanmaktadır.

B. 100-500 arasındaki çift sayılar toplanmaktadır.

C. 100-500 arasındaki tek sayılar toplanmaktadır.

D. 100-500 arasındaki asal sayılar toplanmaktadır.

E. 100-500 arasındaki sayılar toplanmaktadır.

28. Dizilerle ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

A. Birden fazla değişken bir aradadır.

B. indis bulundurur.

C. Tek boyutlu dizi varken, çok boyutlu dizi bulunmamaktadır.

D. Dizilerin uzunluğu eleman sayısına göre değişmektedir.

E. Diziler değişken türleri ile tanımlanmaktadırlar.

29. İndis tanımı aşağıdakilerden hangisidir?

A. Dizilerdeki eleman sayısıdır.

B. Dizinin kaçıncı elemanı olduğunu gösteren değerdir.

C. Dizinin adıdır.

D. Dizinin tanımlandığı değişken türüdür.

E. Dizinin başlangıç değeridir.

30.
```
String Sayi[]=new String[3];
Sayi[0]=25;
Sayi[1]=15;
Sayi[2]=18;
```

Yan tarafta dizilerle ilgili verilen kod bloğunda hata bulunmaktadır. Hatanın çözülmesi için aşağıdaki işlemlerden hangisi yapılmalıdır?

A. Dizinin ismi değiştirilmelidir.

B. Değişken türü int yapılmalıdır.

C. String[3] değeri String[2] şeklinde değiştirilmelidir.

D. Sayi[0] 'ın değeri 15 yapılmalıdır.

E. new terimi kaldırılmalıdır.

31.
```
Byte Deger[]=new Byte[4];
Deger[0]=1;
Deger[1]=2;
Deger[2]=3;
Deger[3]=4;
```

Yan tarafta dizilerle ilgili verilen kod bloğu ile ilgili olarak aşağıdaki ifadelerden hangisi yanlıştır?

A. Deger isminde bir dizi tanımlanmıştır.

B. Dizi 3 elemandan oluşmaktadır.

C. Dizinin değişken türü tamsayıdır.

D. İndis kullanılmıştır.

E. Dizinin son değeri 4 'dür.

32. Aşağıda verilen program parçalarının hangisinde 3 elemanlı bir dizi tanımlanmıştır?

A. Byte Deger[]=new Byte[4];

B. int A[]=new int[5];

C. String Ad[]=new String [6];

D. double Sayilar[]=new double[3];

E. float Sayilar[]=new float[7];

EK 4. Programlamaya Hazır Bulunuşluk Düzeyi Belirleme Testi Belirtke Tablosu

	KONU	KAZANIM	BİLİŞSEL HEDEF					
			BİLGİ	KAVRAMA	UYGULAMA	ANALİZ	SENTEZ	DEĞERLENDİRME
1	Veri tipleri ve değişkenler, operatörler	Farklı veri tiplerini kullanarak değişken tanımlar	2			7		
		Operatörleri kullanarak atama işlemi yapar		6, 8	1	3, 4, 9	5	
2	Kontrol yapıları	Kontrol yapılarının kullanımını kavrar	13	14				
		Kontrol yapıları kullanarak program parçaları oluşturur				12, 15, 18, 19	11, 17	10, 16
3	Döngüler	Döngü yapılarının kullanımını kavrar		20, 26				
		Problem çözümünde ihtiyaç duyulan döngü yapısını oluşturur			25	21, 22, 23		24, 27
4	Diziler	Dizi türündeki değişkenleri kullanır	29	28				
		Tek boyutlu ve çok boyutlu dizi tanımlar		32		30, 31		

EK 5. Problem Çözme Becerisi Algısı Ölçeği

Lütfen bu bölümdeki soruları yanıtlamak için aşağıdaki ölçütleri kullanınız. İfadenin sizin için en uygun doğruluk düzeyini gösteren (1) ile (6) arasındaki bir rakamı işaretleyiniz. Soruları eksiksiz olarak işaretleyiniz.

1. Her zaman böyle davranırım
2. Çoğunlukla böyle davranırım
3. Sıklıkla böyle davranırım
4. Arada sırada böyle davranırım
5. Ender olarak böyle davranırım
6. Hiç böyle davranmam

1	Bir sorunumu çözmek için kullandığım çözüm yolları başarısız ise bunların neden başarısız olduğunu araştırmam.	1	2	3	4	5	6
2	Zor bir sorunla karşılaştığımda ne olduğunu tam olarak belirleyebilmek için nasıl bilgi toplayacağımı uzun boylu düşünmem.	1	2	3	4	5	6
3	Bir sorunumu çözmek için gösterdiğim ilk çabalar başarısız olursa o sorun ile başa çıkabileceğimden şüpheye düşerim.	1	2	3	4	5	6
4	Bir sorunumu çözdükten sonra bu sorunu çözerken neyin işe yaradığını, neyin yaramadığını ayrıntılı olarak düşünmem.	1	2	3	4	5	6
5	Sorunlarımı çözme konusunda genellikle yaratıcı ve etkili çözümler üretebilirim.	1	2	3	4	5	6
6	Bir sorunumu çözmek için belli bir yolu denedikten sonra durur ve ortaya çıkan sonuç ile olması gerektiğini düşündüğüm sonucu karşılaştırırım.	1	2	3	4	5	6
7	Bir sorunum olduğunda onu çözebilmek için başvurabileceğim yolların hepsini düşünmeye çalışırım.	1	2	3	4	5	6
8	Bir sorunla karşılaştığımda neler hissettiğimi anlamak için duygularımı incelerim.	1	2	3	4	5	6
9	Bir sorun kafamı karıştırdığında duygu ve düşüncelerimi somut ve açık seçik terimlerle ifade etmeye uğraşmam.	1	2	3	4	5	6
10	Başlangıçta çözümünü fark etmesem de sorunlarımın çoğunu çözme yeteneğim vardır.	1	2	3	4	5	6
11	Karşılaştığım sorunların çoğu, çözebileceğimden daha zor ve karmaşıktır.	1	2	3	4	5	6
12	Genellikle kendimle ilgili kararları verebilirim ve bu kararlardan hoşnut olurum.	1	2	3	4	5	6
13	Bir sorunla karşılaştığımda onu çözmek için genellikle aklıma gelen ilk yolu izlerim.	1	2	3	4	5	6
14	Bazen durup sorunlarım üzerinde düşünmek yerine, gelişigüzel sürüklenip giderim.	1	2	3	4	5	6
15	Bir sorunla ilgili olası bir çözüm yolu üzerinde karar vermeye çalışırken seçeneklerimin başarı olasılığını tek tek değerlendirmem.	1	2	3	4	5	6

1. Her zaman böyle davranırım
2. Çoğunlukla böyle davranırım
3. Sıklıkla böyle davranırım
4. Arada sırada böyle davranırım
5. Ender olarak böyle davranırım
6. Hiç böyle davranmam

16	Bir sorunla karşılaştığımda, başka konuya geçmeden önce durur ve o sorun üzerinde düşünürüm.	**1**	**2**	**3**	**4**	**5**	**6**
17	Genellikle aklıma ilk gelen fikir doğrultusunda hareket ederim.	**1**	**2**	**3**	**4**	**5**	**6**
18	Bir karar vermeye çalışırken her seçeneğin sonuçlarını ölçer, tartar, birbirleriyle karşılaştırır sonra karar veririm.	**1**	**2**	**3**	**4**	**5**	**6**
19	Bir sorunumu çözmek için plan yaparken o planı yürütebileceğime güvenirim.	**1**	**2**	**3**	**4**	**5**	**6**
20	Belli bir çözüm planını uygulamaya koymadan önce nasıl bir sonuç vereceğini tahmin etmeye çalışırım.	**1**	**2**	**3**	**4**	**5**	**6**
21	Bir soruna yönelik olası çözüm yolları düşünürken çok fazla seçenek üretmem.	**1**	**2**	**3**	**4**	**5**	**6**
22	Bir sorunumu çözmeye çalışırken sıklıkla kullandığım bir yöntem, daha önce başıma gelmiş benzer sorunları düşünmektir.	**1**	**2**	**3**	**4**	**5**	**6**
23	Yeterince zamanım olur ve çaba gösterirsem karşılaştığım sorunların çoğunu çözebileceğime inanıyorum.	**1**	**2**	**3**	**4**	**5**	**6**
24	Yeni bir durumla karşılaştığımda ortaya çıkabilecek sorunları çözebileceğime inancım vardır.	**1**	**2**	**3**	**4**	**5**	**6**
25	Bazen bir sorunu çözmek için çabaladığım halde, bir türlü esas konuya giremediğim ve gereksiz ayrıntılarla uğraştığım duygusunu yaşarım.	**1**	**2**	**3**	**4**	**5**	**6**
26	Ani kararlar verir ve sonra pişmanlık duyarım.	**1**	**2**	**3**	**4**	**5**	**6**
27	Yeni ve zor sorunları çözebilme yeteneğime güveniyorum.	**1**	**2**	**3**	**4**	**5**	**6**
28	Elimdeki seçenekleri karşılaştırırken ve karar verirken kullandığım sistematik bir yöntem vardır.	**1**	**2**	**3**	**4**	**5**	**6**
29	Bir sorunla başa çıkma yollarını düşünürken çeşitli fikirleri birleştirmeye çalışmam.	**1**	**2**	**3**	**4**	**5**	**6**
30	Bir sorunla karşılaştığımda bu sorunun çıkmasında katkısı olabilecek benim dışımdaki etmenleri genellikle dikkate almam.	**1**	**2**	**3**	**4**	**5**	**6**
31	Bir konuyla karşılaştığımda, ilk yaptığım şeylerden biri, durumu gözden geçirmek ve konuyla ilgili olabilecek her türlü bilgiyi dikkate almaktır.	**1**	**2**	**3**	**4**	**5**	**6**
32	Bazen duygusal olarak öylesine etkilenirim ki, sorunumla başa çıkma yollarından pek çoğunu dikkate bile almam.	**1**	**2**	**3**	**4**	**5**	**6**
33	Bir karar verdikten sonra, ortaya çıkan sonuç genellikle benim beklediğim sonuca uyar.	**1**	**2**	**3**	**4**	**5**	**6**
34	Bir sorunla karşılaştığımda, o durumla başa çıkabileceğimden genellikle eminimdir.	**1**	**2**	**3**	**4**	**5**	**6**
35	Bir sorunun farkına vardığımda, ilk yaptığım şeylerden biri, sorunun tam olarak ne olduğunu anlamaya çalışmaktır.	**1**	**2**	**3**	**4**	**5**	**6**

EK 6. Motivasyon Ölçeği

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

Benim için kesinlikle yanlış 1 2 3 4 5 6 7 Benim için kesinlikle doğru

1	Bunun gibi bir derste beni gerçekten çalışmaya zorlayacağına inandığım ders materyallerini tercih ederim, bu sayede yeni şeyler öğrenebilirim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
2	Ancak uygun bir şekilde çalışırsam bu dersin konularını öğrenebilirim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
3	Sınavdayken diğer öğrencilerden daha yetersiz olduğumu düşünürüm.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
4	Bu derste öğrendiklerimi diğer derslerde de kullanabilirim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
5	Bu dersten çok iyi bir not alacağıma inanıyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
6	Bu derste okumam için verilecek en zor konuları bile anlayacağımdan eminim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
7	Benim için en tatmin edici şey sınıfta iyi bir not almaktır.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
8	Sınavda soruları çözerken, sınav kağıdının diğer bölümlerindeki yanıtlayamayacağım soruları düşünürüm.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
9	Eğer bu dersi öğrenemiyorsam bu benim kendi hatamdır.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
10	Bu derste verilen kaynakları (kaynak materyalleri) öğrenmek benim için önemlidir.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
11	Bu derste benim için en önemli şey, genel not ortalamamı yükseltmektir, yani bu dersteki asıl amacım iyi bir not almaktır.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
12	Bu derste anlatılan temel kavramları anlayabileceğim konusunda kendime güveniyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
13	Eğer yapabilirsem, bu sınıftaki diğer öğrencilerin hepsinden daha yüksek not almak isterim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
14	Sınavdayken başarısızlığı ve bunun doğuracağı sonuçları düşünürüm.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
15	Bu derste öğretmenin anlatacağı en zor konuyu bile anlayacağıma güveniyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
16	Bunun gibi bir derste, zor olsalar bile, bende merak uyandıran ders materyallerini tercih ederim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
17	Bu dersle ilgili konulara oldukça ilgi duyuyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
18	Yeterince çalışırsam dersi anlayabilirim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**

Benim için kesinlikle yanlış 1 2 3 4 5 6 7 Benim için kesinlikle doğru

19	Sınavdayken kendimi rahatsız ve morali bozuk hissederim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
20	Bu dersteki ödevleri ve sınavları mükemmel yapabileceğim konusunda kendime güveniyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
21	Bu derste başarılı olmayı bekliyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
22	Bu derste benim için en tatmin edici şey içeriği mümkün olduğunca çok anlayabilmektir.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
23	Bence bu derste kullanılan materyaller dersi öğrenmem için faydalıdır.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
24	Eğer olanak tanınırsa, iyi not almamı sağlamayacak olsa bile en iyi şekilde öğrenmemi sağlayacak ödevleri seçerim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
25	Dersi yeterince anlayamıyorsam, bu yeterince çalışmadığım içindir.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
26	Bu dersin konularını seviyorum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
27	Bu dersin konularını öğrenmek benim için çok önemlidir.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
28	Sınavdayken kalbimin hızla çarptığını hissederim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
29	Eminim ki bu derste öğretilen tüm becerileri ustalıkla yapabilirim.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
30	Sınıfta başarılı olmak isterim; çünkü yeteneğimi aileme, arkadaşlarıma, üstlerime ve diğerlerine göstermek benim için önemlidir.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
31	Dersin zorluğunu, öğretmeni ve becerilerimi dikkate aldığımda, bence bu derste başarılı olurum.	**1**	**2**	**3**	**4**	**5**	**6**	**7**

EK 7. Görüşme Formu

Soru 1. Alice programı ile programlama dilini öğrenmek, programlama dili öğreniminizi nasıl etkiledi?

Soru 2. Alice programı programlama dili öğrenme isteğinizi nasıl etkiledi?

Soru 3. Alice programının derse aktif katılımınızda etkisi oldu mu?

Soru 4. Alice programı ile programlama dili öğrenirken hisleriniz ne oldu?

Soru 5. Alice programı, programlama dili hakkında araştırma yapma, bilgi toplama isteğinizi nasıl etkiledi?

Soru 6. Alice programının kullanımını nasıl buldunuz?

Soru 7. Alice programında en sevdiğiniz özelikler neler oldu?

Soru 8. Alice programında en sevmediğiniz özelikler neler oldu?

Soru 9. Alice programını kullanmaya devam etmek ister misiniz? Neden?

EK 8. Deney Grubu Ders İçeriği

HAFTA 1

Konu : Veri tipleri ve değişkenler, operatörler

a. Veri Tipleri

i. Tam sayı veri tipleri

ii. Ondalıklı sayı veri tipleri

iii. Karaktersel veri tipleri

iv. Mantıksal veri tipleri

b. Değişken tanımlama

i. Değişken tanımlama kuralları

c. Operatörler

i. Aritmetik operatörler

Kazanımlar :

Farklı veri tiplerini kullanarak değişken tanımlar.

Operatörleri kullanarak atama işlemi yapar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Alice programında; tanıtılan metin değerlerini söyleyen, tanıtılan sayı değerleri doğrultusunda bir dikdörtgenin çevresi kadar hareket eden bir nesne hareketinin yapılması.

2. Girilen dört adet kesirli sayıyı okuyup ilk ikisinin çarpımını dördüncüsüne bölen ve üçüncüsüne ekleyerek sonucu veren Java programının yazılması.

Uygulamalar :

1. Alice programında bir dikdörtgenin klavyeden girilen uzun ve kısa kenarı değerleri doğrultusunda çevresi kadar ileri hareket eden, karesi kadar etrafında dönen bir nesnenin tasarlanması.
2. Alice programında klavyeden girilen iki sayının toplamını, büyük olan sayıdan küçük olan sayının çıkarımını, modunu, büyük olan değerleri, bölümünü, bölümünden kalan değeri söyleyen bir nesnenin tasarlanması.
3. Bir malın fiyatı 2000 TL'dir. Bu mal, %18'i kadar KDV'si alınarak satılacaktır. Bu malın KDV'sinin ve satılacağı fiyatın Java'da hesaplanması.
4. Bir dairenin alanını ve çevresini hesaplayan programın Java'da yazılması.

HAFTA 2

Konu : Kontrol yapıları

a. if yapısının kullanımı

i. if / else

ii. if / else if / else

b. switch yapısının kullanımı

i. switch / case

c. Operatörler

i. Mantıksal operatörler

Kazanımlar:

Kontrol yapılarının kullanımını kavrar.

Kontrol yapıları kullanarak program parçaları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Alice programında belirlenen bir şart doğrultusunda hareketine yön verilen bir nesnenin yapılması.
2. Alice programında random olarak atanan sıfır ile altı arasındaki değer dörtten küçükse kısa boylu nesnenin hareket etmesi, dörde eşit ve yukarı ise uzun boylu nesnenin hareket etmesini sağlayan programın yazılması.
3. Sürücü adayının direksiyon ve yazılı sınavından aldığı notlar klavyeden girilerek her iki notunda 50 ve üzeri olması durumunda ekrana *Geçti* aksi durumda *Kaldı* mesajını veren Java programının yazılması.
4. Girilen vize ve final notuna göre harf notunu hesaplayan Java programının yazılması.

Ortalama	**Harf Notu**
Ort< 29	FF
29 <= Ort <39	FD
39<= Ort <45	DD
45<= Ort <50	DC
50<= Ort <57	CC
57<= Ort <65	CB
65<= Ort <75	BB
75<= Ort <85	BA
Ort >=85	AA

Uygulamalar :

1. Alice programında random olarak atanan bir sayının tahmini kullanıcıya 3 hak verilerek yapılan ve program tarafından kullanıcıya geri dönüt veren bir yarışmanın tasarlanması.
2. Girilen sayının tek mi yoksa çift mi olduğunu bulan Java programının yazılması.
3. Bir mağazadan yapılan alışveriş fiyatına göre oluşacak kargo fiyatı değişmektedir. Buna göre kargo dahil oluşacak toplam fiyatı veren Java programının yazılması.

Alışveriş Fiyatı	Kargo Fiyatı
AF<100 TL	5 TL
100 <=AF<500	10 TL
500<=AF<1000	15 TL
AF>=1000	20 TL

HAFTA 3

Konu : Döngüler

a. For döngüsü

b. While döngüsü

c. Do / while

Kazanımlar:

Döngü yapılarının kullanımını kavrar.

Problem çözümünde ihtiyaç duyulan döngü yapısını oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Alice programında zıt yönde hareket eden iki nesnenin birbiriyle çarpışana kadar hareket etmesini sağlayan programın yazılması.

2. Alice programında her adımı random olarak belirlenen iki nesnenin yarışması ve belirlenen şart gerçekleştiğinde kazanının belli olmasını sağlayan programın yazılması.

3. Alice programında kullanıcın girdiği ilerleme ve dönme değerlerine göre döngü yapılarını kullanarak sırasıyla ilerleme ve dönme hareketlerinin gerçekleştirildiği programın yazılması.

4. Birden 500'e kadar olan sayıları ekrana farklı döngüler kullanarak yazdıran Java programının yazılması.

Uygulamalar :

1. Alice programında eklenen dört nesnenin döngü kullanılarak belirlenen bir sayıda sırayla hareket ettirilmesini sağlayan programın yazılması.
2. Alice programında boyları farklı iki nesnenin, kısa boylu olanının belirlenen bir değer doğrultusunda uzaması ve uzun boylu nesnenin boyunu geçince söylemesini sağlayan programın yazılması.
3. Bir ile 999 arasındaki tek tamsayıların toplamını veren Java programının yazılması.
4. Bir bakteri cinsi her dört dakikada bir ikiye bölünerek çoğalmaktadır. Başlangıçta bir bakteri olduğunu düşünerek 40 dakika sonra oluşacak bakteri sayısını bulan Java programının yazılması.

HAFTA 4

Konu : Diziler

a. Tek boyutlu dizi tanımlamaları

b. Çok boyutlu dizi tanımlamaları

Kazanımlar :

Dizi türündeki değişkenleri kullanır.

Tek boyutlu ve çok boyutlu dizi tanımlar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Alice programında eklenen aynı sınıftaki üç nesnenin dizi kullanılarak sırayla aynı etkinliklerin gerçekleştirilmesini sağlayan programın yazılması.
2. Alice programında dizi kullanılarak farklı üç sınıfa ait üç nesnenin tanımlanarak, belirlenen etkinliklerin her sınıf için sırasıyla yaptırılmasını sağlayan programın yazılması.

3. İki katlı bir binanın her katında dört daire bulunmaktadır. Klavyeden her dairede bulunan kişi sayısı girildikten sonra binada kaç kişi olduğunu hesaplayan ve ekrana yazdıra Java programının yazılması.

4. Eksi beş ile artı beş arasında rastgele atanan sekiz elemanlı bir dizinin elemanlarından negatif olanları bir diziye, pozitif olanları başka bir diziye atayan ve bu dizileri de ekranda gösteren Java programının yazılması.

Uygulamalar :

1. Alice programında dizi kullanılarak farklı iki sınıfta toplam dört nesnenin tanımlanmasını ve sıfır ile beş arası random atanan değere göre sınıfların etkinlik yapmasını sağlayan programın yazılması.

2. 20 öğrenciye ait vize ve final bilgileri random atandıktan sonra, her bir öğrencinin ortalamasını hesaplayacak, ekrana eğer öğrenci 70 ve üzeri not almış ise *Geçti* değilse *Kaldı* yazacak Java programının çok boyutlu dizi kullanarak yazılması.

HAFTA 5

Konu : Metotlar Kullanımı

a. Metotlar

i. Local ve Global değişken

Kazanımlar:

Metotların kullanım amacını kavrar.

Amaca göre uygun metot yapıları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Alice programında eklenen nesneye özgü metotların tanımlanması.

2. Alice programında bir nesnenin tüm özelliklerinin kullanımına imkân veren bir metodun oluşturulması.

3. Girilen uzun ve kısa kenara göre dikdörtgenin çevresini hesaplayan *Hesapla* adındaki metodun Java programının yazılması.

Uygulamalar :

1. Alice programında iki nesne ve bu nesnelere ait metotların kullanılmasıyla bir animasyonun tasarlanması.

2. *Döviz Kuru Çevirici* adında bir metot yapılmak istenmektedir. Döviz kur fiyatı ve elinizde bulunan döviz değeri gönderilerek sonucu hesaplayıp geri döndüren metodun Java programının yazılması.

HAFTA 6

Konu : Metot Kullanımı

a. Metotlar

i. Local ve Global değişken

Kazanımlar:

Metotların kullanım amacını kavrar.

Amaca göre uygun metot yapıları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Alice programında eklenen nesneye özgü metotların tanımlanması.

2. Parametre olarak double değerlerden oluşan bir diziyi alan ve bunların ortalamasını yine double bir değer olarak döndüren *ortalama* adlı static bir metodun Java programının yazılması.

Uygulamalar :

1. Alice programında random seçilen sayı beşten küçük ise A nesnesine, beşten büyük ise B nesnesine ait özelliklerin çalıştırılmasını sağlayan metodun yazılması.

2. *Takimlar* adlı dizi içerisinde 18 tane takım bulunmaktadır(Takım1, Takım2, vb.). Bu takımlar arasında yapılacak olan ilk maçlar için 18 tane takımı eşleştiren bir Java programının metot kullanılarak yazılması.

HAFTA 7

Konu : Sınıf yapıları, kalıtım ve nesneler

a. Sınıf yapısı

b. Kurucu (consturctor) metot

c. This deyimi

d. Sınıflarda kalıtım (inheritance)

e. Nesne kullanımı

Kazanımlar:

Sınıf yapısının kullanım amacını bilir.

Amaca göre sınıflar tanımlar.

Sınıf yapısının özelliklerini nesneye aktarır.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Alice programında bir sınıf kaydedilerek ona ait metotların oluşturulması.

2. Katı cisimlerin (küp, silindir vb.) özelliklerini (alan, çevre vb.) içeren ve bu özellikleri kapsayan metotları barındıran sınıfın Java programının yazılması.

Uygulamalar :

1. Alice programında önceden kaydedilmiş sınıfların yeni bir animasyonda çağırılarak sınıf metotları ile animasyon geliştirilmesi.

2. *Spor* isimli sınıfın oluşturularak spor alanındaki özellikler ile ilgili metotları içeren Java programının yazılması.

HAFTA 8

Konu : Sınıf yapıları, kalıtım ve nesneler

a. Sınıf yapısı

b. Kurucu (consturctor) metot

c. This deyimi

d. Sınıflarda kalıtım (inheritance)

e. Nesne kullanımı

Kazanımlar:

Sınıf yapısının kullanım amacını bilir.

Amaca göre sınıflar tanımlar.

Sınıf yapısının özelliklerini nesneye aktarır.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Alice programında bir sınıfa ait metotların oluşturulması ve oluşturulan metotların farklı animasyonlarda kullanılabilmesi için kaydedilmesi.

2. Bir *otomobil* sınıfını içeren Java programının oluşturulması. Bu sınıfta otomobilin markası, modeli, tipi, silindir hacmi, rengi, döşemeleri, motor numarası, şasi numarası, göstergeleri, vergisi, fiyatı, vb. bilgiler bulunacaktır. Ayrıca yürürlükte olan mali mevzuata göre otomobilin vergisini hesaplayan metotlar tanımlanacaktır.

Uygulamalar :

1. Alice programında önceden kaydedilmiş farklı sınıfların yeni bir animasyonda çağırılarak sınıf metotları ile animasyon geliştirilmesi.

2. *Kişi* sınıfını içeren Java programının oluşturulması. Bu sınıfta kişinin baba adı, anne adı, doğum yeri, doğum tarihi, cinsiyeti, yaşadığı şehir, adres, mail adresi, eğitim durumu, medeni hali vb. bilgiler bulunacaktır. Ayrıca kişinin yaşını hesaplayan bir metot tanımlanacaktır.

EK 9. Karşılaştırma Grubu Ders İçeriği

HAFTA 1

Konu : Veri tipleri ve değişkenler, operatörler

a. Veri Tipleri

i. Tam sayı veri tipleri

ii. Ondalıklı sayı veri tipleri

iii. Karaktersel veri tipleri

iv. Mantıksal veri tipleri

b. Değişken tanımlama

i. Değişken tanımlama kuralları

c. Operatörler

i. Aritmetik operatörler

Kazanımlar :

Farklı veri tiplerini kullanarak değişken tanımlar.

Operatörleri kullanarak atama işlemi yapar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Tanımlanan iki sayı üzerinde aşağıda verilen işlemleri uygulayarak sonuçları farklı değişkenlerde tutan Java programının yazılması.

- Toplama - Çarpma - Çıkarma
- Mod Alma - Bölme

2. Girilen dört kesirli sayıyı okuyup ilk ikisinin çarpımını dördüncüsüne bölen ve üçüncüsüne ekleyerek sonucu veren Java programının yazılması.

Uygulamalar :

1. Bir malın fiyatı 2000 TL'dir. Bu mal, %18'i kadar KDV'si alınarak satılacaktır.Bu malın KDV'sinin ve satılacağı fiyatın hesaplandığı Java programının hesaplanması.
2. Bir dairenin alanını ve çevresini hesaplayan Java programının yazılması.
3. Bir karenin kenar uzunluğunun klavyeden girilmesiyle alanı ile çevresini hesaplayan ve sonuçları ekrana yazdıran Java programının yazılması.

HAFTA 2

Konu : Kontrol yapıları

a. if yapısının kullanımı

- **i.** if / else
- **ii.** if / else if / else

b. switch yapısının kullanımı

- **i.** switch / case

c. Operatörler

- **i.** Mantıksal operatörler

Kazanımlar:

Kontrol yapılarının kullanımını kavrar.

Kontrol yapıları kullanarak program parçaları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma.

Örnekler :

1. Klavyeden girilecek boy bilgisine göre; kişinin boyu 170’e eşit veya küçükse ekrana *Kısa Boylusunuz*, 170’den büyükse ekrana *Uzun Boylusunuz* mesajını yazdıran Java programının yazılması.
2. Haftanın günü girildiğinde haftanın kaçıncı günü olduğu ekrana yazdıran Java programının yazılması.

3. Sürücü adayının direksiyon ve yazılı sınavından aldığı notlar klavyeden girilerek her iki notunda 50 ve üzeri olması durumunda ekrana *Geçti* aksi durumda *Kaldı* mesajını veren Java programının yazılması.

4. Girilen vize ve final notuna göre harf notunu hesaplayan Java programının yazılması.

Ortalama	Harf Notu
Ort< 29	FF
29 <= Ort <39	FD
39<= Ort <45	DD
45<= Ort <50	DC
50<= Ort <57	CC
57<= Ort <65	CB
65<= Ort <75	BB
75<= Ort <85	BA
Ort >=85	AA

Uygulamalar :

1. Girilen sayının tek mi yoksa çift mi olduğunu bulan Java programının yazılması.

2. Bir mağazadan yapılan alışveriş fiyatına göre, oluşacak kargo fiyatı değişmektedir. Buna göre kargo dahil oluşacak toplam fiyatı veren Java programının yazılması.

Alışveriş Fiyatı	Kargo Fiyatı
AF<100 TL	5 TL
100 <=AF<500	10 TL
500<=AF<1000	15 TL
AF>=1000	20 TL

3. Üç kenarı girilen bir üçgenin eşkenar, çeşitkenar veya ikizkenar olup olmadığını bulan Java programının yazılması.

HAFTA 3

Konu : Döngüler

a. For döngüsü

b. While döngüsü

c. Do / while

Kazanımlar:

Döngü yapılarının kullanımını kavrar.

Problem çözümünde ihtiyaç duyulan döngü yapısını oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Birden 500'e kadar olan sayıları ekrana yazdıran Java programının yazılması.

2. Çarpım tablosunu oluşturan Java programının yazılması.

3. Yıllık %20 faizle bankaya yatırılan paranın kaç yıl sonra 30 katına çıkacağını hesaplayan Java programının yazılması.

Uygulamalar :

1. Bir ile 999 arasındaki tek tamsayıların toplamını veren Java programının yazılması.

2. Bir bakteri cinsi her 4 dakikada bir ikiye bölünerek çoğalmaktadır. Başlangıçta bir bakteri olduğunu düşünerek 40 dakika sonra oluşacak bakteri sayısını bulan Java programının yazılması.

3. Bir satranç tahtasının birinci karesine bir buğday, ikinci karesine birinci karenin iki katı buğday, üçüncü karesine ikinci karenin iki katı buğday konuluyor. Satranç tahtasının yarısına gelindiğinde konulan toplam buğday sayısını bulan Java programının yazılması.

HAFTA 4

Konu : Diziler

a. Tek boyutlu dizi tanımlamaları

b. Çok boyutlu dizi tanımlamaları

Kazanımlar :

Dizi türündeki değişkenleri kullanır.

Tek boyutlu ve çok boyutlu dizi tanımlar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Gün içerisinde alınan 10 farklı sıcaklık verisini dizide tutarak günün en sıcak ve soğuk değerlerini ekrana yazdıran Java programının yazılması.

2. 10 elemanlı bir dizi değişkeni tanımlayan ve bu dizi değişkenine 100 ile 300 arasında rastgele sayılar üreterek aktaran Java programının yazılması.

3. İki katlı bir binanın her katında dört daire bulunmaktadır. Klavyeden her dairede bulunan kişi sayısını girildikten sonra binada kaç kişi olduğunu hesaplayan ve ekrana yazdıran Java programının yazılması.

Uygulamalar :

1. Eksi beş ile artı beş arasında rastgele atanan sekiz elemanlı bir dizinin elemanlarından negatif olanları bir diziye, pozitif olanları başka bir diziye atayan ve bu dizileri de ekranda gösteren Java programının yazılması.

2. Elemanları tam sayı olan 20 elemanlı bir A dizisine rastgele sayılar atayan, dizi elemanlarının toplamını, aritmetik ortalamasını hesaplayan ve elemanlardan kaç tanesinin hesaplanan ortalamadan büyük ve küçük olduğunu sayan ve ekrana yazdıran Java programının yazılması.

3. Üç öğrenciye ait vize ve final bilgileri random atandıktan sonra, her bir öğrencinin ortalamasını hesaplayacak ve ekrana eğer öğrencinin ortalaması 70 ve üzeri ise *Geçti* değilse *Kaldı* yazacak Java programının çok boyutlu dizi kullanarak yazılması.

HAFTA 5

Konu : Metot Kullanımı

a. Metotlar

i. Local ve Global değişken

Kazanımlar:

Metotların kullanım amacını kavrar.

Amaca göre uygun metot yapıları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Java'da klavyeden girilen bir cümlede bulunan kelime sayısını veren metodun yazılması.

2. Java'da girilen uzun ve kısa kenara göre dikdörtgenin çevresini hesaplayan *Hesapla* adındaki metodun yazılması.

Uygulamalar :

1. Java'da klavyeden girilen sayının tek veya çift olduğunu bulan metodun yazılması.

2. Java'da *Döviz Kuru Çevirici* adında bir metot tanımlanmak istenmektedir. Döviz kur fiyatı ve elde bulunan döviz değeri girilerek sonucu hesaplayıp geri döndüren metodun yazılması.

HAFTA 6

Konu : Metot Kullanımı

a. Metotlar

i. Local ve Global değişken

Kazanımlar:

Metotların kullanım amacını kavrar.

Amaca göre uygun metot yapıları oluşturur.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Girilen vize ve final notuna göre harf notunu hesaplayan Java programının metot kullanılarak yazılması.

2. Java'da parametre olarak double değerlerden oluşan bir diziyi alan ve bunların ortalamasını yine double bir değer olarak döndüren *ortalama* adlı static bir metodun yazılması.

Uygulamalar :

1. Java'da klavyeden girilen bir sayının pozitif, negatif veya sıfır olup olmadığını bulan metodun yazılması.

2. *Takimlar* adlı dizi içerisinde 18 tane takım bulunmaktadır (Takım1, Takım2, vb.). Bu takımlar arasında yapılacak olan ilk maçlar için 18 tane takımı eşleştiren bir Java programının metot kullanılarak yazılması.

HAFTA 7

Konu : Sınıf yapıları, kalıtım ve nesneler

a. Sınıf yapısı

b. Kurucu (consturctor) metot

c. This deyimi

d. Sınıflarda kalıtım (inheritance)

e. Nesne kullanımı

Kazanımlar:

Sınıf yapısının kullanım amacını bilir.

Amaca göre sınıflar tanımlar.

Sınıf yapısının özelliklerini nesneye aktarır.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Java'da katı cisimlerin (küp, silindir vb.) özelliklerini (alan, çevre vb.) içeren ve ilgili metotları barındıran sınıfın yazılması.

Uygulamalar :

1. Java'da *Spor* isimli sınıfın oluşturularak spor dalı ile ilgili özellikleri içeren metotların yazılması.
2. Java'da bilgisayarın özelliklerini içeren ve bu özellikleri kapsayan metotları barındıran sınıfın yazılması.

HAFTA 8

Konu : Sınıf yapıları, kalıtım ve nesneler

a. Sınıf yapısı

b. Kurucu (consturctor) metot

c. This deyimi

d. Sınıflarda kalıtım (inheritance)

e. Nesne kullanımı

Kazanımlar:

Sınıf yapısının kullanım amacını bilir.

Amaca göre sınıflar tanımlar.

Sınıf yapısının özelliklerini nesneye aktarır.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Java'da *otomobil* sınıfının oluşturulması. Bu sınıfta otomobilin markası, modeli, tipi, silindir hacmi, rengi, döşemeleri, motor numarası, şasi numarası, göstergeleri, vergisi, fiyatı, vb. bilgiler bulunacaktır. Ayrıca yürürlükte olan mali mevzuata göre otomobilin vergisini hesaplayan metotlar tanımlanacaktır.

Uygulamalar :

1. Java'da *Kişi* sınıfının oluşturulması. Bu sınıfta kişinin baba adı, anne adı, doğum yeri, doğum tarihi, cinsiyeti, yaşadığı şehir, adres, mail adresi, eğitim durumu, medeni hali vb. bilgiler bulunacaktır. Ayrıca kişinin yaşını hesaplayan metot tanımlanacaktır.
2. Java'da *Öğrenci* sınıfı oluşturularak öğrenci özelliklerini kapsayan metotların tanımlanması.

EK 10. Hafta 1 İçin Değerlendirme Örneğini İçeren Uzman Değerlendirme Formu

UZMAN DEĞERLENDİRME FORMU

Bu form Nesne Tabanlı Programlama I dersinde yürütülecek olan araştırma kapsamında sınıflarda uygulanacak ders içeriklerini değerlendirmek amacıyla hazırlanmıştır. Karşılaştırma olarak adlandırılan gruptaki örnekler ve uygulamalar Java programlama dili kullanılarak NetBeans derleyicisi üzerinde geliştirilecektir. Deney olarak adlandırılan grupta öncelikle Alice programı ile sonrasında Java programlama dili kullanılarak NetBeans derleyici üzerinde örnek ve uygulamalar geliştirilecektir. Grup ders içeriklerini inceleyerek öncelikle her grubu kendi içerisinde daha sonra ise grupları karşılaştırarak, 1 ile 5 arasında bir puanda (1 en düşük 5 en yüksek puanı temsil etmektedir) değerlendiriniz. Önerilerinizi (varsa) belirtiniz. Katkılarınız için teşekkür ederim.

Ceren BAŞTEMUR KAYA

Deney

HAFTA 1

Konu : Veri tipleri ve değişkenler, operatörler

a. Veri Tipleri
   i. Tam sayı veri tipleri
   ii. Ondalıklı sayı veri tipleri
   iii. Karaktersel veri tipleri
   iv. Mantıksal veri tipleri
b. Değişken tanımlama
   i. Değişken tanımlama kuralları
c. Operatörler
   i. Aritmetik operatörler
   ii. Mantıksal operatörler

Kazanımlar :

Veri tipine göre değişken tanımlar.
Aritmetik ve mantıksal operatörler arasındaki farkı bilir.
Atama işlemlerini yapar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Alice programında tanıtılan metin değerlerini söyleyen, tanıtılan sayı değeri doğrultusunda hareket eden bir nesne hareketinin yapılması.

Puan : 3	
Görüş : Karşılaştırma grubundaki eş değer örneğe göre biraz zayıf kalmıştır. İçerisinde +, -, * ve / operatörlerinin kullanılabileceği şeklinde örneğin yeniden düzenlenmesi faydalı olacaktır.	

2. Girilen dört kesirli sayıyı okuyup ilk ikisinin çarpımını dördüncüsüne bölen ve üçüncüsüne ekleyerek sonucu veren JAVA programının yazılması.

Puan : 5	
Görüş : Uygundur.	

Uygulamalar :

1. Alice programında random atanan bir değerle hareket eden ve dönen bir nesnenin tasarlanması.

Puan : 4	
Görüş : Operatörler kullanılarak zenginleştirilsin	

2. Alice programında kullanıcı tarafından girilen değerleri söyleyen bir nesnenin tasarlanması.

Puan :	4
Görüş :	Operatörleri kullanarak sonuçları değiştirmen daha yerinde olacaktı.

3. Bir malın fiyatı 2000 TL dir. Bu mal,%18 i kadar KDV si alınarak satılacaktır.Bu malın KDV sinin ve satılacağı fiyatın JAVA 'da hesaplanması.

Puan :	5
Görüş :	Konuya Uygundur

4. Bir dairenin alanını ve çevresini hesaplayan programın JAVA 'da yazılması.

Puan :	5
Görüş :	Konuya Uygundur

Değerlendirme : -

HAFTA 2
Ders : Nesne Tabanlı Programlama
Konu : Kontrol yapıları

a. if yapısının kullanımı
   i. if / else
   ii. if / else if / else
b. switch yapısının kullanımı
   i. switch / case

Kazanımlar:
Kontrol yapılarının kullanım alanını bilir.
İhtiyaca göre farklı kontrol yapıları kullanır.
if ve switch yapıları arasındaki farkı bilir.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma, grup çalışması
Örnekler :

1. Alice programında belirlenen bir şart doğrultusunda hareketine yön verilen bir nesnenin yapılması.

Puan :	5
Görüş :	Uygundur.

Karşılaştırma

HAFTA 1

Konu : Veri tipleri ve değişkenler, operatörler

a. Veri Tipleri
   i. Tam sayı veri tipleri
   ii. Ondalıklı sayı veri tipleri
   iii. Karaktersel veri tipleri
   iv. Mantıksal veri tipleri
b. Değişken tanımlama
   i. Değişken tanımlama kuralları
c. Operatörler
   i. Aritmetik operatörler
   ii. Mantıksal operatörler

Kazanımlar :

Veri tipine göre değişken tanımlar.
Aritmetik ve mantıksal operatörler arasındaki farkı bilir.
Atama işlemlerini yapar.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma

Örnekler :

1. Tanımlanan iki sayı üzerinde aşağıda verilen işlemleri uygulayarak sonuçları farklı değişkenlerde tutan JAVA programının yazılması.
   1) Toplama
   2) Çıkarma
   3) Bölme
   4) Çarpma
   5) Mod Alma
   6) Üs Alma

Puan : 5
Görüş : Operatörlerin ve değişkenlerin kullanımı açısında yeterlidir.

2. Girilen dört kesirli sayıyı okuyup ilk ikisinin çarpımını dördüncüsüne bölen ve üçüncüsüne ekleyerek sonucu veren JAVA programının yazılması.

Puan : 5
Görüş : Uygundur.

Uygulamalar :

1. Bir malın fiyatı 2000 TL dir. Bu mal,%18 i kadar KDV si alınarak satılacaktır. Bu malın KDV sinin ve satılacağı fiyatın hesaplanması.

Puan : 5
Görüş : Kapsaya Uygundur.

2. Bir dairenin alanını ve çevresini hesaplayan programın JAVA 'da yazılması.

Puan : 5
Görüş : Konuya Uygundur.

3. Bir koninin bir taban yarıçapı ve yüksekliğinin girilmesi ile Hacmini, Yanal Alanı, Toplam alanı hesaplayan ve ekrana yazdıran JAVA programını yazılması.

Puan : 4
Görüş : Sınıf matematikel ağırlık içerdiğinde biraz basitleştirilsin. Değişken olanı yetki olur.

Değerlendirme : -

HAFTA 2
Ders : Nesne Tabanlı Programlama
Konu : Kontrol yapıları

a. if yapısının kullanımı
   i. if / else
   ii. if / else if / else
b. switch yapısının kullanımı
   i. switch / case

Kazanımlar:
Kontrol yapılarının kullanım alanını bilir.
İhtiyaca göre farklı kontrol yapıları kullanır.
if ve switch yapıları arasındaki farkı bilir.

Öğretim Yöntemleri : Anlatım, alıştırma, soru-yanıt, gösterip yaptırma, grup çalışması
Örnekler :

1. Klavyeden girilecek boy bilgisine göre; 170'e eşit veya küçükse ekrana 'KISA BOYLUSUNUZ.', 170'den büyükse ekrana "UZUN BOYLUSUNUZ.' mesajını yazdıran programın yazılması.

Puan : Uygundur (5)
Görüş : Uygun.

	Hafta 1	Deney					Karşılaştırma				
		1	2	3	4	5	1	2	3	4	5
1	Örnekler konuya uygundur				X						X
2	Örnekler konunun anlaşılması için yeterlidir				X						X
3	Uygulamalar konuya uygundur					X					X
4	Uygulamalar örneklerle uyumludur					X					X
5	Örnekler ve uygulamaların birlikte kullanımı öğrencinin konuyu öğrenme potansiyelini arttırmaktadır					X					X
6	Öneriler : Deney grubundaki örnek ve uygulamalar konu içeriğine uygun olarak yeniden düzenlenmesi durumunda daha etkil olacaktır.										
7	Gruplarda kullanılan örnekler konunun anlaşılması için birbiriyle eş düzeydedir								X		
8	Deney..... grubu örneklerinin düzenlenmesi gereklidir Nedeni : Örnekler diğer gruba göre daha kolaydır. Öneriler: Operatör ve atamalı içeren ezber yapması uygun olacaktır.										
9	Gruplarda kullanılan uygulamalar konunun anlaşılması için birbiriyle eş düzeydedir								X		
10	Deney..... grubu uygulamalarının düzenlenmesi gereklidir Nedeni : Operatör ile ilgili kazanım yetersizdir. Öneriler: Sorların ilgili boşluklarına yapılan öneriler dikkate alınmalıdır.										

GAZİLİ OLMAK AYRICALIKTIR...