BLOK TABANLI GÖRSEL VE METİN TABANLI PROGRAMLAMA ÖĞRETİMLERİNİN ERİŞİ, MANTIKSAL DÜŞÜNME VE MOTİVASYONA ETKİLERİ

EFFECTS OF BLOCK-BASED VISUAL AND TEXT-BASED PROGRAMMING INSTRUCTION ON ACHIEVEMENT, LOGICAL THINKING AND MOTIVATION

Şenol SAYGINER

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı İçin Öngördüğü

YÜKSEK LİSANS TEZİ

Olarak Hazırlanmıştır.

2017

JÜRİ ONAY BİLDİRİMİ

Eğitim Bilimleri Enstitüsü Müdürlüğü'ne,

Şenol SAYGINER'in hazırladığı "Blok Tabanlı Görsel ve Metin Tabanlı Programlama Öğretimlerinin Erişi, Mantıksal Düşünme ve Motivasyona Etkileri" başlıklı bu çalışma jürimiz tarafından **Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı'nda Yüksek Lisans Tezi** olarak kabul edilmiştir.

Başkan	Prof. Dr. Arif ALTUN	
Üye (Danışman)	Doç. Dr. Hakan TÜZÜN	
Üye	Prof. Dr. Halil YURDUGÜL	
Üye	Doç. Dr. Hasan ÇAKIR	
Üye	Yrd. Doç. Dr. Cengiz Savaş AŞKUN	

ONAY

Bu tez Hacettepe Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin ilgili maddeleri uyarınca yukarıdaki jüri üyeleri tarafından ..... / ..... / ..... tarihinde uygun görülmüş ve Enstitü Yönetim Kurulunca ..... / ..... / ..... tarihinde kabul edilmiştir.

Prof. Dr. .....
Eğitim Bilimleri Enstitüsü Müdürü

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI BEYANI

Enstitü tarafından onaylanan lisansüstü tezimin/raporumun tamamını veya herhangi bir kısmını, basılı (kağıt) ve elektronik formatta arşivleme ve aşağıda verilen koşullarla kullanıma açma iznini Hacettepe Üniversitesine verdiğimi bildiririm. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanım hakları bana ait olacaktır.
Tezin kendi orijinal çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alınarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

☐ **Tezimin/Raporumun tamamı dünya çapında erişime açılabilir ve bir kısmı veya tamamının fotokopisi alınabilir.**
(Bu seçenekle teziniz arama motorlarında indekslenebilecek, daha sonra tezinizin erişim statüsünün değiştirilmesini talep etseniz ve kütüphane bu talebinizi yerine getirse bile, teziniz arama motorlarının önbelleklerinde kalmaya devam edebilecektir)

☒ **Tezimin/Raporumun 16.01.2019 tarihine kadar erişime açılmasını ve fotokopi alınmasını (İç Kapak, Özet, İçindekiler ve Kaynakça hariç) istemiyorum.**
(Bu sürenin sonunda uzatma için başvuruda bulunmadığım takdirde, tezimin/raporumun tamamı her yerden erişime açılabilir, kaynak gösterilmek şartıyla bir kısmı veya tamamının fotokopisi alınabilir).

☐ **Tezimin/Raporumun …….. tarihine kadar erişime açılmasını istemiyorum ancak kaynak gösterilmek şartıyla bir kısmı veya tamamının fotokopisinin alınmasını onaylıyorum.**

☐**Serbest Seçenek/Yazarın Seçimi:** ……
……

16 /01/2017

ETİK BEYANNAMESİ

Hacettepe Üniversitesi Eğitim Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında,

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversitede veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

Şenol SAYGINER

TEŞEKKÜR

Derin birikimleriyle ve çalışma azmiyle bana yol gösteren, çalışma disiplini, sabrı ve özverisiyle kendime örnek aldığım tez danışmanım Doç. Dr. Hakan TÜZÜN'e çalışmanın başından sonuna kadar bilgisini ve tecrübelerini benden esirgemediği, yolun başında olmam ve tecrübesizliğimden kaynaklanan hatalarıma hoşgörüyle yaklaştığı için teşekkürü bir borç bilirim.

Çalışmamı inceleyen ve değerli katkılarını eksik etmeyen hocalarım Prof. Dr. Arif ALTUN'a, Prof. Dr. Halil YURDUGÜL'e, Doç. Dr. Hasan ÇAKIR'a ve Yrd. Doç. Dr. Cengiz Savaş AŞKUN'a sonsuz teşekkürlerimi sunarım.

Veri toplama araçlarını dolduran katılımcılara, veri toplama sürecinde bana yardımcı olan ve değerli fikirlerini esirgemeyen Mustafa Kemal Üniversitesi Eğitim Fakültesi akademik personeline teker teker teşekkür ederim.

Eğitim yaşamımın en önemli aşamalarından birisi olan yüksek lisans sürecinde bilgileriyle beni aydınlatan ve alanda farklı bakış açıları geliştirmemi sağlayan Hacettepe Üniversitesi Eğitim Fakültesi BÖTE Bölümü Öğretim Elemanlarına teşekkürü bir borç bilirim.

Desteklerini hiçbir zaman esirgemeyen aileme sonsuz saygı, sevgi ve teşekkürlerimi sunarım.

Sevgili Aileme...

BLOK TABANLI GÖRSEL VE METİN TABANLI PROGRAMLAMA ÖĞRETİMLERİNİN ERİŞİ, MANTIKSAL DÜŞÜNME VE MOTİVASYONA ETKİLERİ

Şenol SAYGINER

ÖZ

Bu araştırmanın amacı, blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin öğrencilerin erişi, mantıksal düşünme ve programlamaya yönelik motivasyonlarına etkilerini belirlemektir.

Tek gruplu ön test son test deney modeline göre hazırlanan araştırmanın çalışma grubunu, 2015-2016 Eğitim-Öğretim yılının Bahar döneminde bir devlet üniversitesinin Fen Bilgisi Öğretmenliği Bölümü'nde öğrenim gören toplam 60 öğrenci oluşturmaktadır. Yapılan araştırmada iki farklı deneysel işlem gerçekleştirilmiş ve sonuçlar ayrı ayrı değerlendirilmiştir. 30 öğrenciden oluşan A şubesindeki öğrencilere 10 hafta süresince Scratch ile blok tabanlı görsel programlama öğretimi yapılmıştır. B şubesinde ise toplam 30 öğrenci çalışmaya dâhil edilmiş olup bu grupta da 10 hafta süresince Small Basic ile metin tabanlı bir programlama öğretimi gerçekleştirilmiştir.

Araştırma kapsamında veri toplama aracı olarak programlama erişi testi, mantıksal düşünme testi ve programlamaya yönelik motivasyon ölçeği kullanılmıştır.

Verilerin çözümlenmesinde bağımsız gruplar için t-testi, Mann-Whitney U testi ve eşleştirilmiş örneklem t-testi kullanılmıştır. Analizler sonucunda; blok tabanlı görsel programlama öğretiminin yapıldığı öğrencilerin erişi, mantıksal düşünme ve motivasyon ön test puanlarına kıyasla son test puanlarında istatistiksel olarak anlamlı bir artış elde edilmiş ve incelenen değişkenler için yüksek düzeyde bir etki büyüklüğü hesaplanmıştır. Metin tabanlı programlama öğretiminin yapıldığı grupta ise erişi ve mantıksal düşünme ön test puanları son testte istatistiksel olarak anlamlı bir artış göstermiştir. Erişi puanları için yüksek düzeyde, mantıksal düşünme içinse orta düzeyde bir etki büyüklüğü hesaplanmıştır. Metin tabanlı programlama öğretiminin yapıldığı grupta öğrencilerin motivasyon ön test puanları

eğitimlerin ardından uygulanan son testte istatistiksel olarak anlamlı bir artış oluşturmamıştır. Motivasyon puanları için hesaplanan etki büyüklüğü ise düşük düzeydedir.

Anahtar sözcükler: Blok tabanlı görsel programlama, scratch, small basic, erişi, mantıksal düşünme, motivasyon.

Danışman: Doç. Dr. Hakan TÜZÜN, Hacettepe Üniversitesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı.

EFFECTS OF BLOCK-BASED VISUAL AND TEXT-BASED PROGRAMMING INSTRUCTION ON ACHIEVEMENT, LOGICAL THINKING AND MOTIVATION

Şenol SAYGINER

ABSTRACT

This study aims to identify the effects of block-based visual and text-based programminginstructions on the students' achievement, logical thinking and their motivation for programming.

The study group of the research which was prepared based on one group pre test and post test model was composed of 60 students studying at the department of science education of a state university in the Spring semester of the academic year 2015-2016. Two different experimental procedures were carried out and the results were evaluated separately. The students in the A branch received training on block-based visual programming with Scratch for 10 weeks whereas the students in the B branch were trained on the text-based programming with Small Basic.

As data collection tool, the study employed programming achievement test, logical thinking test and motivation scale.

In the analysis of the data, the t-test, the Mann-Whitney U test and the paired sample t-test were used. As a result of the analyses; there was a statistically significant increase in the post test scores compared to the achievement, logical thinking and motivation pre test scores of the students with block-based visual programming instruction. In addition, a high effect size was calculated for the variables examined. There was a statistically significant increase in the post test scores compared to the achievement and logical thinking pre test scores of the students with text-based programming instruction. The effect size was calculated at a high level for achievement and at a medium level for logical thinking. In addition, the motivation pre test scores of students with text-based programming did not show a statistically significant increase in the final test following their training. The effect size calculated for motivation scores was low.

Keywords: Block-based visual programming, scratch, small basic, achievement, logical thinking, motivation.

Advisor: Assoc. Prof. Dr. Hakan TÜZÜN, Hacettepe University, Department of Computer Education and Instructional Technology.

İÇİNDEKİLER

TABLOLAR DİZİNİ

ŞEKİLLER DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

MDBT: Mantıksal Düşünme Becerisi Testi

BÖTE: Bilgisayar ve Öğretim Teknolojileri Eğitimi

1. GİRİŞ

Bu bölümde problem durumu, araştırmanın amacı, önemi, araştırma problemleri, sayıltıları ve sınırlılıkları yer almaktadır.

1.1. Problem Durumu

Bilgi ve iletişim teknolojilerinde yaşanan hızlı değişimler, bu dönemde yetişen bireylerden beklentilerin de değişmesine yol açmıştır. "*21.yy bireyleri*" ya da "*Dijital nesil*" gibi adlandırmaların yapıldığı bu dönemde problem çözebilen, mantıksal çözüm yolları üretebilen ve yaratıcılık becerisi gelişmiş bireyler yetiştirilmeye çalışılmaktadır (EARGED, 2011; Pinto ve Escudeiro, 2014). Bu becerilerin bireylere kazandırılmasının, programlama ve bilgisayar biliminin öğretilmesi ile mümkün olabileceği çeşitli araştırmacılar tarafından ifade edilmektedir (Monroy-Hernandez ve Resnick, 2008; Nam, Kim ve Lee, 2010; Lai ve Lai, 2012; Karabak ve Güneş, 2013; Kobsiripat, 2014; Giordano ve Mairona, 2015; Korkmaz, 2016). Ancak dünya genelinde yapılan çalışmalarda, programlama öğretiminde birtakım sorunların yaşandığı belirtilmektedir. Örneğin; Mccracken ve arkadaşları (2001) tarafından gerçekleştirilen bir araştırmada, öğrencilerin sadece %21'nin programlama derslerinde başarılı olabildikleri görülmüştür. Benzer bir çalışma Kinnunen ve Malmi (2008) tarafından yapılmış olup bu çalışmada da dünya genelindeki üniversitelerde programlama temelleri ve programlamaya giriş dersine kayıt olan öğrencilerin %20 - %40 oranında dersi veya bölümü daha ilk yılda bıraktıkları görülmüştür. Bir başka araştırmada ise eğitimin her kademesinde, özellikle de üniversite düzeyinde, öğrencilerin öğrenmekte en çok zorlandıkları derslerden birisinin bilgisayar programlama dersi olduğu belirtilmektedir (Jenkins, 2002).

Programlama öğretimi konusunda yapılmış çalışmalar değerlendirildiğinde bütün dünyada öğrencilerin büyük çoğunluğunun programlamayı öğrenme konusunda büyük sıkıntılar yaşadıkları görülmektedir. Programlama öğretiminde yaşanan sıkıntıları gidermek için proje tabanlı öğrenme ortamları, işbirlikli çalışma, oyun tabanlı öğrenme gibi farklı yaklaşım ve yöntemler önerilmektedir (Başer, 2013). Bu yaklaşımlardan birisi de blok tabanlı görsel programlama ortamlarıdır. Alanyazında blok tabanlı görsel programlama ortamlarından Scratch yazılımı sıklıkla karşımıza

çıkmaktadır. Örneğin Çatlak, Tekdal ve Baz (2015), Scratch yazılımıyla ilgili alanyazında yer alan 32 çalışmayı incelemiş ve Scratch yazılımının programlamada ve üst düzey düşünme becerilerinin (eleştirel düşünme, yaratıcılık, analitik düşünme vb.) gelişiminde etkili olduğunu belirtmişlerdir. Bunun yanında incelenen çalışmaların çoğunlukla ilköğretim düzeyinde gerçekleştirilmiş olduğunu belirtmişlerdir. Bu çalışmanın sonuçlarını destekler nitelikte diğer çalışmalar da alanyazında yer almaktadır (Nam, Kim ve Lee, 2010; Kukul ve Gökçearslan, 2014; Kobsiripat, 2014; Giordano ve Mairona, 2015; Korkmaz, 2016).

Günümüzde görsel programlama eğitiminin önemine inanan ülkeler, eğitim müfredatlarına bu dersi dâhil etmiş veya etmek üzere hazırlık yapmaktadırlar (Sayın ve Seferoğlu, 2016). Bu kapsamda özellikle Avrupa ülkelerinde programlama eğitiminin erken yaşlarda verilmesine yönelik bir eğilimin olduğu göze çarpmaktadır. Yapılacak bu eğitimlerle öğrencilerin "mantıksal düşünme ve problem çözme becerilerini geliştirmek" amaçlanmaktadır (Avrupa Okul Ağı, 2015). Nitekim benzer projeler pek çok ülkede gerçekleştirilmektedir. Ülkemiz açısından bakıldığında da son zamanlarda ilköğretim ve ortaöğretim müfredatına programlama dersleri dâhil edilmekte ve öğrencilerde "21.yy becerileri" olarak adlandırılan mantıksal düşünme, problem çözme, yaratıcılık, eleştirel düşünme gibi birtakım becerilerin geliştirilmesi hedeflenmektedir. Ancak gerek yurt dışı gerekse yurt içi perspektifinden bakıldığında öncelikle öğretmen veya öğretmen adaylarında bu becerilerin geliştirilmesi üzerine fazlaca yoğunlaşılmadığı belirtilebilir. Bunun bir sebebi olarak programlama eğitimlerinin ağırlıklı olarak Mühendislik veya Bilgisayar bölümlerinde verilmekte olması gösterilebilir. Bunun dışındaki fakültelerde programlama dersleri çok az sayıda veya hiç bulunmamaktadır.

Alanyazında görsel programlama eğitiminin etkililiği üzerine yapılan bilimsel çalışmaların da son zamanlarda yoğunluk kazandığı görülmektedir. Ancak yapılan bu çalışmalar çoğunlukla ilköğretim ve lise düzeylerinde gerçekleştirilmektedir. Üniversite düzeyinde ise az sayıda çalışma yapılmış ve bu çalışmaların da neredeyse tamamı programlama öğretimi üzerine odaklanmıştır. Bir başka ifadeyle, üniversite düzeyinde gerçekleştirilecek bir blok tabanlı görsel programlama eğitiminin mantıksal düşünme üzerindeki etkisini araştıran çalışmaların sınırlı sayıda olduğu görülmüştür.

1.2. Araştırmanın Amacı

Bu araştırmanın genel amacı, blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin, Fen Bilgisi Öğretmenliği 2. sınıf öğrencilerinin erişi, mantıksal düşünme ve programlamaya yönelik motivasyonlarına etkisini belirlemektir.

1.3. Araştırmanın Önemi

Blok tabanlı görsel programlama eğitimiyle ilgili yapılan çalışmalar incelendiğinde, birçok çalışmanın ilköğretim ve ortaöğretim düzeyinde yapıldığı ve bu düzeylerde programlama mantığının öğretimi, problem çözme, mantıksal düşünme ve yaratıcılık gibi beceriler üzerinde yoğunlaşıldığı görülmüştür. Üniversite düzeyinde ise özellikle temel programlama derslerinde yapılmış çalışmalar olmakla birlikte mantıksal düşünme ve problem çözme gibi beceriler üzerindeki etkisini araştıran çalışmaların sınırlı düzeyde olduğu ve yapılan çalışmaların neredeyse tamamının Bilgisayar Mühendisliği alanında öğrenim gören öğrencilerle gerçekleştirildiği görülmüştür. Üniversite düzeyinde Bilgisayar Mühendisliği dışında öğrenim gören öğrencilerle gerçekleştirilecek blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin erişi, mantıksal düşünme ve programlamaya yönelik motivasyon üzerindeki etkisini tespit etmek adına yararlı olacağı düşünülmektedir. Başka bir deyişle, önceki eğitim hayatında algoritma veya programlama gibi bir eğitim almamış ve üniversite düzeyinde ise ilk defa programlama eğitimi alacak olan bir grupta görsel programlama eğitiminin yaratacağı etkiyi ortaya koymak adına yararlı bir çalışma olacağı düşünülmektedir. Ayrıca bu çalışma ile 21. yy becerileri açısından öncelikle öğretmen adaylarının yetiştirilmesine dikkat çekilmeye çalışılmıştır. Ulaşılacak sonuçlardan hareketle programlama eğitiminin sadece bilgisayarla ilgili bölümlerde öğrenim gören öğrencilere değil, üniversite düzeyinde çeşitli bölümlerde de okutulması yönünde bir öneride bulunulabilir.

1.4. Problem Cümlesi

Blok tabanlı görsel ve metin tabanlı programlama eğitimleri alan öğrencilerin erişi testi, mantıksal düşünme testi ve programlamaya yönelik motivasyon ölçeğinden aldıkları puanların ortalamaları ön testlere kıyasla son testlerde istatistiksel olarak anlamlı bir artış göstermiş midir?

Araştırmanın problemi doğrultusunda aşağıdaki alt problemlere cevap aranmıştır.

1.4.1. Alt Problemler

Blok tabanlı görsel programlama eğitimi alan öğrencilerin;

1. Erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
2. Mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
3. Programlaya yönelik motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?

Metin tabanlı programlama eğitimi alan öğrencilerin;

4. Erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
5. Mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
6. Programlaya yönelik motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?

1.5. Sayıltılar ve Sınırlılıklar

Aşağıda bahsedilen durumlar, bu çalışmanın sayıltıları ve sınırlılıkları olarak kabul edilmiştir.

1. Gruplardaki öğrencilerin çalışmaya istekli ve gönüllü oldukları kabul edilmiştir.
2. Çalışmaya katılan öğrencilerin bilgisayar kullanım becerileri eşit kabul edilmistir.
3. Veri toplama amacıyla kullanılan ölçek ve testlerin araştırılan değişkenleri ölçtüğü kabul edilmiştir.
4. Çalışmaya katılan öğrencilerin veri toplama araçlarını samimiyetle yanıtladığı kabul edilmiştir.

1.6. Tanımlar

Bilgisayar Programlama: Mevcut bir problemin çözümü için üretilen çeşitli kod satırlarının belirli kurallar çerçevesinde dizilimi ile bilgisayar programları oluşturma sürecidir.

Programlama Erişisi: Verilen bir programlama görevini başarı ile yerine getirme, programı anlama, algoritma şemasını oluşturabilme, program hatalarını ayıklayabilme ve bir program yazabilme becerisidir.

Mantıksal Düşünme: Bireyin bir problemle karşılaştığında, o problemin çözümüne yönelik ortaya koyduğu zihinsel işlemlerdir.

Motivasyon: Bireylerin hedefe ulaşmasına yardım eden içsel bir enerji veya zihinsel bir güçtür.

Scratch: Bireylerin blokları bir yerden bir yere sürükle bırak etkinlikleriyle animasyonlar, oyunlar veya çeşitli yazılımlar geliştirebildikleri blok tabanlı görsel bir programlama dilidir.

Small Basic: BASIC programlama dilinin daha basit bir versiyonu olan Small Basic, kullanıcılarına kod yazarak çeşitli programlar geliştirme olanağı sunan metin tabanlı bir programlama dilidir.

2. İLGİLİ ALANYAZIN

2.1. Programlama ve Programlama Eğitimi

Programlama, en genel tanımı ile bilgisayara komutlar vererek birtakım işleri yaptırmaktır. Başka bir ifadeyle programlama, herhangi bir problemin çözümü için gerekli komutlar dizisinin, bilgisayarın anlayabileceği şekilde komutlara çevrilmesi, derlenmesi ve çalıştırılmasıdır (Kesici ve Kocabaş, 2001). Eryılmaz'a (2003) göre programlama, iyi bir şekilde analiz ve tasarımı yapılmış problemin çözümüne ait adımlar ile çözümün oluşturulup bir programlama dili ile bilgisayar ortamına aktarılması işidir. Alanyazında yer alan bu tanımlamalara bakıldığında programlama, mevcut bir problemin çözümü için üretilen çeşitli kod satırlarının belirli kurallar çerçevesinde dizilimi ile bilgisayar programları oluşturma süreci olarak tanımlanabilir.

Bilgisayar dili olarak adlandırılan özel kelime ve sembollerin bir araya gelerek oluşturdukları yapı ise programlama dilidir. Günümüzde çok sayıda programlama dilli geliştirilmiş ve her geçen gün yenileri de geliştirilmeye devam etmektedir. Her bir programlama dilinin kendine özgü kalıp ve yazım kuralları ile amaca uygun komutlar oluşturulur. Dile özgü komutların yazılması süreci kodlama ya da programlama, ortaya çıkan son ürün ise program ya da uygulama olarak adlandırılır (Ersoy, Madran ve Gülbahar, 2011).

Programlar, bir işin nasıl yapılacağını bilmeyen ancak işin yapılması için gerekli tüm donanıma sahip olan bilgisayarlara, adım adım işin nasıl yapılacağını anlatan yapılardır (Çağıltay ve Fal, 2014). Bu nedenle bir program yazılmadan önce, programı yazan kişinin problemi çok iyi anlaması ve adım adım süreci tasarlayabilmesi gerekmektedir. Problemin adım adım detaylı bir şekilde tanımlanması süreci ise algoritma olarak tanımlanabilir. Bir başka ifadeyle algoritma, bir görevi yerine getirmek veya bir problemi çözmek için sıralanan adımların bir kümesi olarak ifade edilebilir. Algoritma, özellikle programlama alanında kullanılan bir kavram olmasına rağmen, aslında günlük hayatta bilinçli ya da bilinçsiz olarak yaptığımız birçok iş bir algoritmaya göre şekillenmektedir. Örneğin evde yemek pişirirken, birine yol tarifi yaparken, trafik ışıklarında beklerken, birine telefon ederken hep bir sıralamayı takip ederiz. Bu işlemsel

sıralamalar birer algoritmadır. Benzer şekilde, bilgisayara işlerimizi yaptırtmak veya bir problemi çözdürtmek gerektiğinde, o işi hangi adımlarda yapması gerektiğini mantıklı bir sırada ve bilgisayarın anlayacağı bir dilde (programlama dili) vermek gerekir. Yani problemin çözüm algoritmasının oluşturulması gerekir.

Programlama öğrenimi karmaşık bir süreçtir (Kert ve Uğraş, 2009; Helminen ve Malmi, 2010; Çatlak, Tekdal ve Baz, 2015). Programlamanın problem çözme ve mantıksal düşünme gibi birtakım zihinsel becerileri kapsaması, programlama dillerinin karmaşık sözdizimi ve kullanıcı dostu olmayan arayüzleri gibi durumlar öğrencilerin programlamayı öğrenmelerini zorlaştıran sebepler olarak gösterilebilir. Alanyazında programlama öğrenimi süresince öğrencinin öğrenmesi gereken birbiri ile ilişkili üç tip programlama bilgisinin olduğu belirtilmektedir (Bayman ve Mayer, 1988). Bunlar, bir programlama diline ait yazım kurallarını kapsayan sözdizimsel (syntactic) bilgi, programlama kavramlarını ve prensiplerini kapsayan kavramsal (conceptual) bilgi ve programlamayla ilgili problem çözme sürecini kapsayan problem çözme–stratejik bilgidir (Bayman ve Mayer, 1988). Bu çeşitlilikten dolayı programlama öğretimi karmaşık bir süreç gerektirir. McGill ve Volet (1997) bu süreci öğrenim ve öğretim boyutuyla ele almış olup geliştirdikleri sistem Tablo 2.1'de sunulmuştur.

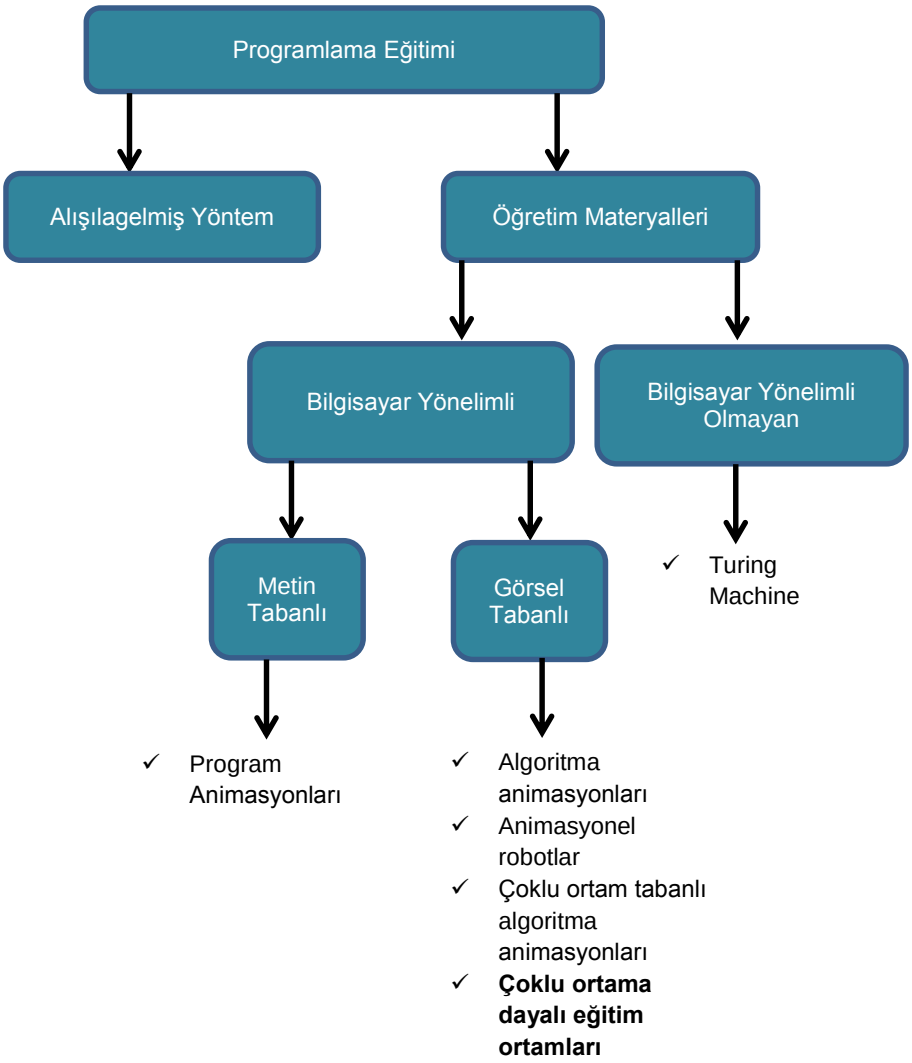
Tablo 2.1: McGill ve Volet'in (1997) Programlama Dilleri Öğrenim ve Öğretim Tablosu

	Bildirimsel Bilgi (Declarative Knowledge)	***İşlemsel Bilgi (Procedural Knowledge)***
Sözdizimsel (yazımsal) Bilgi (Syntactic Knowledge)	Programlama dilinin yazım kuralları bilgisi. Örneğin: Pascal programlama dilinde her komutun sonuna noktalı virgül işaretinin konulması gerektiğini bilme.	Program yazarken yazım kurallarına uyabilme. Örneğin: Small Basic programlama dilindeki If komutunu yazım kuralları açısından doğru olarak yazabilme.
Kavramsal Bilgi (Conceptual Knowledge)	Program çalıştırıldığında hangi mantıksal işlemlerin yapıldığını anlama ve açıklayabilme becerisi. Örneğin: Bir sözde kodun (Pseudocode) ne iş yaptığını açıklayabilme.	Programlamada karşılaşılan bir probleme yönelik çözüm üretebilme. Örneğin: Verilen sayıların ortalamalarını bulan bir prosedür oluşturabilme.

Stratejik / Durumsal Bilgi (Strategic / Conditional Knowledge)	Yeni bir problemle karşılaşıldığında buna ait bir çözüm üretmek için program tasarlama, kodlama ve test edebilme becerisi.

McGill ve Volet (1997) programlamada, öğrenenlerin kazanmaları gereken birbiri ile ilişkili üç bilgi türünden söz etmektedir. Birincisi sözdizimsel, ikincisi kavramsal, üçüncüsü ise stratejik/durumsal bilgidir. Sözdizimsel bilgi bir programlama diline ait yazım kuralları bilgisi ve bu kuralları kullanabilme olarak tanımlanmıştır. Kavramsal bilgi bilgisayar programcılığının yapıları ve prensipleri, stratejik bilgi ise problem çözme becerisi olarak tanımlanmıştır.

Programlama dili öğretimi için çeşitli öğretim materyalleri geliştirilmiştir (Örnek: Şekil 2.1).



Şekil 2.1: Programlama Dili Öğretim Materyallerinin Sınıflandırılması (Gültekin, 2006)

Geliştirilen bu öğretim materyallerinden bazıları üç boyutlu, bazıları ise iki boyutlu programlar yazma imkânı sunmaktadır. Bu tür ortamlar kullanıcılarına görselleştirilmiş yapılar sunmaktadır. Bilgisayar programlamayı görselleştiren bu

sistemlerin asıl amacı ise programların veya algoritmaların dinamik görsel sunumlarını hazırlamak veya sunmaktır (Gültekin, 2006). Ayrıca görsel programlama ortamları özellikle programlama ile yeni tanışan kullanıcılara karmaşık ilişkileri analiz edebilme, karşılaşılan problemleri küçük parçalara bölebilme, mantıksal çözüm yolları oluşturabilme ve herhangi bir programlama diline ait temel yapıları en etkili şekilde kullanabilme yeteneği kazandırmaktadır (Çağıltay ve Fal, 2014). Bu ortamlarla ilgili detaylı anlatımlar ilerleyen bölümlerde sunulmuştur.

2.2. Yurt Dışı ve Yurt İçinde Programlama Eğitimi

Günümüzde, bütün dünyada öğrencilerin programlamayı erken yaşlardan itibaren öğrenmesi gerektiği görüşü sıklıkla dile getirilmektedir. Çünkü 21. yüzyılda yetişen bireylerde bulunması istenen yaratıcılık, problem çözme ve üst düzey düşünme gibi birtakım becerilerin programlama ve bilgisayar biliminin öğretilmesi ile kazandırılabileceği görüşü savunulmaktadır (Monroy-Hernandez ve Resnick, 2008; Karabak ve Güneş, 2013; Shin, Park ve Bae, 2013). Bu kapsamda gerek yurt dışında gerek yurt içinde programlama eğitiminin öneminin farkına varılmış olup birçok ülke bu eğitime müfredat programlarında yer vermiştir.

2.2.1.Yurt Dışında Programlama Eğitimi

Programlama eğitimi tüm dünyada ağırlıklı olarak lisans düzeyinde yapılmaktadır (Karabak ve Güneş, 2013). Ancak son zamanlarda, özellikle gelişmiş ülkelerde, programlama sürecine öğrencileri daha erken yaşta başlatmak, yazılımı sevdirmek ve programlama eğitimini daha geniş bir alana yaymak amacıyla birtakım düzenlemeler yapılmaktadır. Örneğin; İngiltere'de 2013 yılı Kasım ayı itibariyle ilkokuldan itibaren okullarda bilgisayar programlama eğitimine başlanmış ve 2014 yılı tüm ülkede "*Kodlama Yılı*" (Year of Code) olarak ilan edilmiştir (Özdemir, 2015). Ayrıca okullardaki bilgi ve iletişim teknolojileri dersinin içeriğini "*çağa ayak uyduramadığı*" gerekçesi ile değiştirerek bu dersin içeriğine, zorunlu ve geniş bir programlama bölümü eklemişlerdir. Okullarda programlama öğretimini 5-6, 7-11 ve 12-14 yaş grubuna göre kademelere ayırarak her kademeye özel programlama eğitimleri planlamışlardır. Birinci basamakta (5-6 yaş grubu) algoritmanın ne olduğunun öğretilmesi, ikinci basamakta (7-11 yaş grubu) daha karmaşık programlar oluşturmaları ve hataları ayıklayabilecek düzeye gelmeleri

hedeflenmiştir (Öndeş, 2016). Üçüncü basamakta (12-14 yaş grubu) ise öğrencilerin iki ya da daha fazla programlama diline hâkim olmaları beklenmektedir (Öndeş, 2016).

Amerika Birleşik Devletleri; hükümet, sivil toplum kuruluşları, Microsoft ve Google gibi teknoloji ve yazılım şirketlerinin desteğiyle okullarda programlama eğitimi konusunda birçok çalışma yapmaktadır. Bu çalışmalar arasında 2013 yılında kurulan "*code.org*" platformu öne çıkmaktadır. ABD'de yaklaşık 6 milyon öğrencinin kullanmakta olduğu bu platform sayesinde öğrenciler programlama ile ilgili yüzlerce kavramı kullanarak kendi programlarını yazabilmektedirler (Öndeş, 2016).

Güney Kore Bilim ve Gelecek Planlama Bakanlığı, üstün yetenekli iş gücü sağlamak amacıyla ilkokuldan itibaren programlama derslerinin zorunlu olacağını açıklamıştır. Açıklamada, ilkokulların 2017, liselerin 2018 yılından itibaren kademeli olarak programlama eğitimine başlayacaklarını belirtmişlerdir (Özçakmak, 2014). Bu uygulama ile ilköğretim düzeyinde öğrencilerin bilgisayar işletmenliğinden çok algoritmaları anlamaya odaklanılması gerektiği üzerinde durulmuştur (Choi, Bell, Junve Lee, 2008).

Yazılım alanında önemli ilerlemeler göstermiş olan Hindistan'ın müfredat programına bakıldığında, hemen her kademede bilgisayar eğitiminin yer aldığı görülmektedir. Hindistan'da bir öğrenci ilkokulda (1 ve 4. sınıflar arası) algoritmanın temelleri ve LOGO programlama eğitimi almaktadır. Ortaokulda, BASIC programlama diline geçiş yapılmakta ve sabitler, değişkenler ve döngülerle ilgili derslere yer verilmektedir. Lisede ise dersler, daha ileri düzey programlar yazılacak şekilde düzenlenmiştir (SSRVM, 2007).

Estonya'da, 2012 yılında pilot bir program uygulamaya konulmuş ve ardından ilköğretim 1. sınıf itibariyle programlama eğitiminin verilmesine karar verilmiştir (Olson, 2012).

Avustralya Eğitim Bakanlığı tarafından 2015 yılı itibariyle okullarda birinci sınıftan (5 yaşından) itibaren 2 yıl boyunca temel programlama dilleri ve kod eğitimi verilmesi planlanmıştır. Dersler sonraki senelerde ileri düzey programlama derslerine dönüşecek ve ortalama 7 yaşında bir öğrenci temel programlama

mantığını çözmüş olacak şekilde bir düzenleme yapılmıştır (Öçalan, 2015; Kahraman, 2015).

Fransa'da 2015 yılı itibariyle programlama eğitiminin en erken kaç yaşında verilebileceği üzerine yapılan bir araştırma sonucunda temel programlama eğitiminin okul öncesi eğitim çağından itibaren verilebileceği belirtilmiştir. Bu düzeyde verilecek eğitimlerde amacın öğrencilerin sezgi gücünü ve görsel düşüncesini geliştirmek olduğu belirtilmektedir. Ayrıca çocuklara makineleri programlayanların insanlar olduğunu kavratmak amaçlanmaktadır (Euronews, 2015).

Macaristan'da "*programlamanın sadece erkek işi olmadığı*" iddiasını göstermek için "*Programcı Kızlar*" adıyla bir proje yürütülmektedir. Bu proje kapsamında ortaokulda öğrenim gören kız öğrencilere yönelik "*Processing*" programlama dili ile eğitim verilmektedir. Günden güne büyüyen bu projeyi sivil toplum kuruluşları ve farklı ülkeler de desteklemektedir (Euronews, 2015).

Çin'de ise temel kodlama eğitiminin daha erken yaşlarda verildiği gözlenmiştir. Temel kodlama eğitimi olarak adlandırılan derslerde kart oyunları ile eğitimler gerçekleştirilmekte olup bu eğitimler okul öncesi dönemde verilmektedir (Özkaya, 2016).

Yurtdışında programlama konusunda yapılan çalışmalar incelendiğinde ülkelerin programlama eğitimine verdiği önemin arttığı ve birçok ülkede bu eğitimlerin erken yaşlarda verilmesine yönelik bir eğilimin olduğu görülmektedir. Ayrıca programlama eğitimleri ülkelerin müfredat programlarında farklı başlıklar altında tanımlanmaktadır. Bilgisayar programlama, programlama eğitimi, kod eğitimi, kodlama, algoritma en fazla tercih edilen ders tanımlarıdır. Ülkeler arasında oluşan bu farklılığın, eğitimin verildiği sınıf düzeyi ve kullanılan programlama ortamıyla ilgili olduğu belirtilebilir. Örneğin Çin'de, temel kodlama eğitimi olarak tanımlamakta ve bu eğitim okul öncesi düzeyinde verilmekte iken Macaristan'da programlama olarak tanımlanmakta ve bu eğitim ortaokul düzeyinde verilmektedir. Amerika, "kodlama" olarak tanımlamakta ve öğrencilerine "code.org" platformu üzerinden blok tabanlı bir eğitim sunmakta iken Hindistan ortaokul düzeyinde "programlama eğitimi" kavramını kullanmakta ve "Basic" ortamında eğitim

sunmaktadır. Bu anlamda ülkemizdeki mevcut durumun incelenmesi, yurt dışında yapılan çalışmalarla karşılaştırma adına fayda sağlayacaktır.

2.2.2.Yurt İçinde Programlama Eğitimi

Yurt dışında olduğu gibi ülkemizde de programlama eğitimine verilen önem her geçen gün artmaktadır. Bu kapsamda ülkemizde 2012 yılına kadar "Bilgisayar" ve "Bilişim Teknolojileri" gibi isimlendirmelerin yapıldığı bilgisayar dersleri, 2012 yılında alınan bir kararla "Bilişim Teknolojileri ve Yazılım" olarak güncellenmiştir (BTE Derneği, 2013). Dersin adında yazılım kavramı ilk defa kullanılmış olup bu çerçevede müfredata algoritma ve programlama ile ilgili konular dâhil edilmiştir. Beşinci sınıftan itibaren öğrencilere temel düzeyde programlama eğitimleri verilmeye başlanmıştır (BTE Derneği, 2013). 2016 yılına gelindiğinde ise Milli Eğitim Bakanlığı (MEB) tarafından alınan bir karar ile ortaöğretim müfredatına "Bilgisayar Bilimi" adıyla yeni bir ders dâhil edilmesi kararlaştırılmıştır (MEB, 2016). Bu karar doğrultusunda güzel sanatlar ve spor liselerinde 2016-2017, diğer ortaöğretim kurumlarında ise 2017-2018 eğitim ve öğretim yılından itibaren "Kur-1" den başlamak üzere kademeli olarak "Bilgisayar Bilimleri" dersinin verilmesi planlanmıştır. Ders, Kur-1 ve Kur-2 olmak üzere iki kurdan oluşmaktadır. Her kur için 72 saatlik bir eğitim yapılacağı kararlaştırılmıştır. Yapılacak eğitimlerin içeriği incelendiğinde Kur-1'e ayrılan sürenin %93'ünün, Kur-2'nin ise tamamının programlama öğretimi üzerine planlanmış olması, ülkemizde de programlama alanında önemli adımlar atıldığının somut bir göstergesidir.

Ülkemizde programlama eğitimini yaygınlaştırmak için çeşitli sivil toplum kuruluşları, MEB, üniversiteler ve şirketler birtakım projeler yürütmektedir. Örneğin, Milli Eğitim Bakanlığı bünyesinde geliştirilmiş olan Eğitimde Bilişim Ağı (EBA) portalı üzerinden öğrenci ve öğretmenler özgün programlar yazabilmekte veya başkası tarafından yazılan bir programın kod satırlarına erişerek geliştirmeler yapabilmektedir. Ayrıca 2014 yılında Türkiye Bilişim Derneği (TBD) ve çeşitli üniversitelerin desteğiyle "Bilgisayar Programlama Çocuk Oyuncağı" adlı bir etkinlik organize edilmiştir. Bu etkinliğin amacı ilk, orta ve lise öğrencilerinin bilgisayar ve İnternet teknolojileriyle kendi programlarını yazabileceklerini ve bunun basit bir şey olduğunu fark etmelerini sağlamaktır (TBD, 2014).

Programlama eğitimini ülke genelinde yaygınlaştırmak amacıyla yürütülen diğer bir proje ise ülkemizde faaliyet gösteren mobil operatör şirketlerinden birisi tarafından gerçekleştirilmektedir. Proje, bireylerin teknolojiyi tüketen değil aynı zamanda üreten bireyler olarak yetişmesini sağlamak amacıyla oluşturulmuştur (Hatısaru, 2016). Ayrıca programlama alanında kız öğrencilerin sayısını arttırmak da belirtilen hedefler arasındadır (Yüzak, 2016). Projenin Milli Eğitim Bakanlığı'nın desteği ile okullarda 7-13 yaş aralığındaki öğrenciler için gerçekleştirileceği belirtilmektedir. Projenin 2016-2017 eğitim öğretim döneminde başlatılacağı hedeflenmiştir (Hatısaru, 2016).

Ülkemizde gerçekleştirilen bir diğer önemli proje ise Bilişim Garaj Akademisi tarafından 7-8, 9-12 ve 13-16 yaş gruplarına yönelik programlama, web tasarımı, 3-Boyutlu tasarım ve robot üretimi gibi eğitimlerdir. Bu eğitimler sayesinde öğrenciler, programlamanın bir problem çözme süreci olduğunun farkına varmaktadır (Demirer ve Sak, 2016). Yapılan bu çalışmalara bakıldığında programlama eğitiminin ülkemizde de değer kazandığını söylemek mümkündür.

2.3. Programlama Öğretiminde Yaşanan Zorluklar

Programlama, donanım ile yazılım arasındaki ilişkiyi oluşturan bir süreçtir. Bu sebeple bir donanımın nasıl davranacağını oluşturmak ve yönlendirmek problem çözme, mantık yürütme, karar verme gibi birtakım üst düzey düşünme becerilerini kullanmayı gerektirir (Monroy-Hernandez ve Resnick, 2008; Shin, Park ve Bae, 2013; Akpınar ve Altun, 2014). Bu beceriler ve programlamanın temel yapıları olan algoritma, koşullu ifadeler, döngüler, fonksiyonlar ve dizilerin öğrenimi ve gelişimi zor bir süreç olarak görülmektedir (Fesakis ve Serafeim, 2009). Bu durumun sebebi olarak programlama eğitimine yönelik öğrencilerin olumsuz bir tutuma sahip olmaları (Hongwarittorrn ve Krairit 2010; Başer, 2013), programlama diline ait kavramların soyutluğu ve karmaşıklığı (Esteves ve Mendes, 2004; Ozoran, Çağıltay ve Topallı, 2012) ve program yazarken yabancı dil kullanımı (Arabacıoğlu, Bülbül ve Filiz, 2007) gibi durumlar gösterilmektedir. Buna ilaveten alanyazında yer alan birçok çalışmada karşılaşılan sorunlar ve bu sorunların nasıl çözülebileceği etraflıca ortaya konulmaya çalışılmıştır.

Bayman ve Mayer (1983), programlama öğrenmeye yeni başlayanların en fazla programlama diline has kavramları anlamakta zorlandıklarını rapor etmişlerdir.

Aynı yıl yapılan başka bir çalışmada ise programlama öğrenmede yaşanan zorlukların planlama ve tasarım becerilerindeki eksikliklerden kaynaklı olduğu belirtilmiştir (Pea ve Kurland, 1983).

DuBoulay (1986) programlama öğrenmede yaşanan zorlukları 4 tema altında ele almıştır. Bunlar, programlamanın amacını ve problemin ne olduğunu anlama (orientation), programlama dilinin sözdizimini açıklama (notion), programlamada kullanılan standart yapıları anlayabilme (structure) ve program yazmak için gerekli olan ileri düzey becerilere sahip olma (pragmatics) olarak belirtilmiştir.

Linn and Clancy (1992), öğrencilerinin program yazarken karşılaştıkları problemleri tespit etmek üzere yapmış olduğu çalışmasında, öğrencilerin sadece programı yazmaya odaklandıklarını, mantıksal çıkarımlar yapmadıklarını belirtmiştir.

Byrne ve Lyons (2001) programlama alanında yaşanan zorluğun temel sebebinin geleneksel programlama öğretim yönteminden kaynaklı olduğunu belirtmektedir. Byrne ve Lyons'a göre, geleneksel programlama öğretim yöntemi kitaplarda kuralları vererek bu kuralların öğrenciler tarafından ezberlenmesini ister. Benzer durum Esteves ve Mendes'in (2004) çalışmalarında da görülmektedir.

Lahtinen, Ala-Mutka ve Jarvinen (2005), programlamada karşılaşılan zorluklarla ilgili olarak üniversite öğrencileri ve öğretmenlerle yürüttükleri anket çalışmasının sonucunda programlama yapısının anlaşılması ve belli bir görevi yerine getirecek bir programın nasıl tasarlanacağının anlaşılması gibi konularda zorluklar yaşandığını belirtmişlerdir.

Gomes ve Mendes (2007), programlama eğitiminde yaşanan zorlukları 5 kategoride ele almıştır. Bu kategoriler Tablo 2.2'de sunulmuştur.

Tablo 2.2: Gomes ve Mendes'e (2007) Göre Programlama Öğretiminde Yaşanan Zorluklar

1	**Öğretme yöntemleri**	• Öğretim kişiselleştirilmemektedir. • Öğretmenlerin öğretim stratejileri öğrencilerin öğrenme stillerine uymamaktadır. • Statik materyallerle dinamik kavramların öğretimi yapılmaktadır. • Öğretmenler problem çözmeyi geliştirmek yerine bir programlama dilini ve onun sözdizimini öğretmeye odaklanmaktadır.
2	**Çalışma yöntemleri**	• Öğrenciler doğru olmayan çalışma yöntemleri kullanmaktadır. • Öğrenciler programlama yeterliliklerini elde etmek için sıkı çalışmamaktadır.
3	**Öğrencilerin beceri ve davranışları**	• Öğrenciler nasıl problem çözüleceğini bilmemektedir. • Çoğu öğrencinin yeterli matematiksel ve mantıksal bilgisi bulunmamaktadır. • Öğrencilerin programlamaya özgü bilgileri eksiktir.
4	**Programlamanın doğası**	• Programlamanın soyut yapısı. • Programlamanın karmaşık sözdizimi.
5	**Psikolojik etkiler**	• Öğrencilerin motivasyonu düşüktür. • Öğrenciler programlamayı zor bir dönemde öğrenmek zorunda kalmaktadır.

Kinnunen ve Malmi (2008), 340 üniversite öğrencisi ile programlama dersinde en fazla zorlanılan konuları tespit etmek amacıyla bir çalışma yapmışlardır. Çalışmanın sonuçlarına göre, öğrencilerin en fazla hata bulma, problemin çözümü için algoritma oluşturma ve kod yazma konularında zorluk yaşadıklarını belirtmişlerdir.

İmal ve Eser (2009) programlama eğitiminde yapılan hataları 3 başlık altında ele almışlardır. Bunlar, aynı anda farklı dil öğrenme, ihtiyaç ve ezberci yaklaşımdır. Her programlama dilinin kendine has yazım kuralları ve kalıpları olmasından dolayı aynı anda birden fazla dil öğrenmeye kalkıldığında, diller arası karışıklık yaşandığını belirtmişlerdir. Programlama dili öğrenimine başlanmadan önce ihtiyaç veya ihtiyaçların belirlenmesi ve bu ihtiyaçları karşılayacak bir programlama dilinin öğrenilmesi gerektiği belirtilmektedir. Ayrıca problemler sürekli değiştiğinden dolayı yeni çözüm yolları üretmek gerektiğinden programlamada ezberci bir yaklaşımdan kaçınılmasının faydalı olacağı ifade edilmektedir.

Özmen ve Altun (2014), bir üniversitede İnternet Tabanlı Programlama dersinin laboratuvar etkinliklerini gözlemlemiş ve derse devam eden 12 öğrenci ile görüşmeler yaparak öğrencilerin programlama sürecinde yaşadıkları zorlukların programlama bilgisi, programlama becerisi, programın mantığını kavrama ve hata ayıklama olduğunu belirtmişlerdir. Öğrencilerin bu alandaki başarısızlıklarının en

büyük nedenlerinin ise pratik eksikliği, algoritma oluşturamama ve bilgi eksikliğinden kaynaklı olduğunu belirtmişlerdir.

Tablo 2.3'te programlama öğretiminde yaşanan zorluklar özetlenmiştir.

Tablo 2.3: Alanyazında Yer Alan Çalışmalara Göre Programlama Öğretiminde Yaşanan Zorluklar

	Programlama dilinin yapısı	*Programlama mantığının oluşturulamaması*	*Programlama öğretim yöntemi*	*Programlama öğrenim yöntemi*	*Program yazarken yabancı dil kullanımı*	*Programlama öğrenmeye yönelik tutum*	*Öğrenci özyeterlik düzeyi*	*Motivasyon*	*Aynı anda farklı programlama dillerini öğrenmeye çalışma*	*Pratik eksikliği*
Bayman ve Mayer (1983)						X				
Pea ve Kurland (1983)		X								
DuBoulay (1986)		X								
Linn ve Clancy (1992)		X								
Byrne ve Lyons (2001)			X							
Esteves ve Mendes (2004)	X		X							
Lahtinen, Ala-Mutka ve Jarvinen (2005)		X								
Gomes ve Mendes (2007)	X		X	X			X	X		
Arabacıoğlu, Bülbül ve Filiz (2007)					X					
Kinnunen ve Malmi (2008)		X								
İmal ve Eser (2009)				X					X	
Hongwarittorrn ve Krairit (2010)						X				
Ozoran, Çağıltay ve Topallı (2012)	X									
Başer (2013)						X				
Özmen ve Altun (2014)		X					X			X

Geçmişten günümüze doğru programlama eğitiminde yaşanan zorluklara bakıldığında, neredeyse her dönemde algoritma ve programlama mantığı ile ilgili unsurlar görülmektedir. Kullanılan programlama dilinden ya da öğretim yönteminden kaynaklı olarak öğrencilerin henüz sürecin başında birtakım zorluklarla karşılaşmasının motivasyonlarını düşürmekte olduğu ve zamanla programlamadan soğumalarına sebep olduğu söylenebilir. Programlamanın üst düzey düşünme becerileri gerektiren bir yapısının olması da bu eğitime yeni başlayanları tedirgin edebilmektedir. Bu becerilerle ilgili detaylar bir sonraki bölümde ele alınmıştır.

2.4. Programlama Öğretiminde Gerekli Olan Beceri ve Durumlar

Programlama, temelinde yoğun bir bilişsel süreci barındırdığı için öğrenimi sürecinde çeşitli beceriler gerektirmektedir. Problem çözme ve mantıksal düşünme bu becerilere örnek olarak gösterilebilir. Nitekim alanyazında problem çözme ile mantıksal düşünme arasında bir ilişki olduğu belirtilmektedir. Aşkar'a (1989) göre mantıksal düşünme, problem çözmenin alt aşamalarından birisidir. Bu ifade, mantıksal analiz veya muhakeme yapabilen bir kişinin karşılaştığı problemlere etkili çözümler ortaya koyabileceği şeklinde yorumlanabilir. Bu kapsamda bu çalışmada her iki becerinin de tanımlaması yapılmış ve programlamayla olan ilişkileri etraflıca ortaya konulmaya çalışılmıştır.

2.4.1. Problem Çözme Becerisi

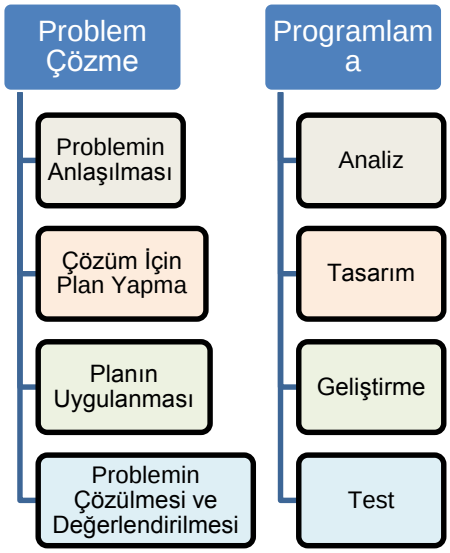
Günümüzde hızlı değişim gösteren teknolojiler, insan hayatını kolaylaştırmaya yönelik olsa da beraberinde aşılması zor birtakım problemleri de getirmektedir. Her insan, hayatı boyunca çeşitli problemlerle karşılaşmaktadır. İnsanlar karşılaştıkları bu problemleri en kısa zamanda en az çaba ile çözme eğilimindedirler. Alanyazında problem kavramıyla ilgili çeşitli tanımlamalar yapılmıştır. Kneeland (2001), problemi bir şeyin olması gereken durumu ile şu anda olan durumu arasındaki fark olarak tanımlamıştır. Türnüklü ve Yeşildere'ye (2005) göre problem; zihni karıştıran, kişide çözme isteği uyandıran ancak ilk defa karşılaşılması nedeniyle bir çözüm yolu bulunamayan, sadece çözmek için çalışan kişinin sahip olduğu bilginin doğru şekilde kullanılması sonucu çözülmesi mümkün olan sorundur. Karasar (2005), problemi birden çok çözüm yolu olasılığı bulunan ve bireyi fiziksel ya da düşünsel yönden rahatsız eden tüm durumlar şeklinde tanımlamış ve problemin 3 özelliğinin olduğunu belirtmiştir. Bunlar;

1. Problem, karşılaşılan kişi için bir güçlük, bir engeldir.
2. Kişi problemi çözmeye ihtiyaç duyar ve problemin çözümüne yönelik bir amaç belirler.
3. Kişi bu problemle daha önce karşılaşmamış olduğundan çözümle ilgili bir hazırlığı bulunmamaktadır. Bu durum, bireyi amacına ulaşmaya iten içsel bir gerginlik yaratır.

Bir problemle karşılaşıldığında problemi çözmek için ilk olarak durumun analiz edilmesi gerekir. Ardından gerekli bilgilerin toplanması, bunlardan çözüme yardımcı olanların seçilmesi ve seçilen bilgilerin uygun şekilde düzenlenerek kullanılması gerekir (Kaya, 2011). Bu süreç problem çözme süreci olarak ele alınabilir. D'Zurilla ve Goldfried (1971) problem çözmeyi, problemle başa çıkabilmek için etkili tepki seçeneklerini oluşturmayı ve bu seçeneklerin en etkili olanını seçebilmeyi içeren bilişsel ve davranışsal bir süreç olarak tanımlamıştır. Heppner ve Petersen (1982) problem çözmeyi, karşılaşılan bir problemle başa çıkma süreci olarak ele almaktadır. Altun (2004) ise problem çözmeyi, problemi çözme gayreti sırasındaki sürecin tümü olarak tanımlamaktadır. Problem çözme becerisi ise kişiyi çözüme götürecek kuralların edinilip kullanıma hazır kılınabilecek ölçüde birleştirilerek bir problemin çözümünde kullanılabilme düzeyidir (Bilen, 2002).

Problem çözmenin bilişsel, duygusal ve davranışsal etkinlikleri içeren karmaşık bir süreç olduğu söylenebilir. Problem çözmenin bilişsel yönü eleştirel, yansıtıcı ve yaratıcı düşünmeyi, analiz yapabilme ve sentezleme becerilerinin kullanımını gerektirir (Soylu ve Soylu, 2006).

Programlama birçok yönü ile problem çözme süreçlerini kapsamaktadır. Bu konuda Pea and Kurland (1987), programlamanın aşamalarıyla problem çözme sürecinin aşamalarını Şekil 2.2'deki gibi eşleştirmiştir.



Şekil 2.2: Problem Çözme ve Programlama Süreçleri (Pea ve Kurland, 1987)

Problemi anlama: Problem çözmede karşılaşılan önemli güçlüklerden birisi, problemin gereği gibi okunup anlaşılamamasından kaynaklıdır. Programlamada da bu durum sıklıkla yapılan bir hatadır. Yeni bir program yazarken ya da var olan programda değişiklik yapmaya gidildiğinde öncelikle problemin ne olduğu net bir şekilde ortaya konulmalıdır.

Çözüm için plan yapma: Bu aşama, problemi çözmeye yardımcı olabilecek bütün bilgilerin toplandığı ve bu bilgiler ışığında değişkenler arasındaki ilişkilerin kurulduğu, hangi değişkenlerin ilişkili ya da ilişkisiz olduğunun belirlendiği ve olası bütün çözüm yollarının ortaya konulduğu aşamadır. Program yazarken de problemin anlaşılmasının ardından olası bütün çözüm yolları belirlenmeye ve çözüm için bir yol haritası oluşturulmaya çalışılır.

Planın uygulanması: Bir önceki bölümde bir problemi çözmek için belirlenen en uygun yöntemin uygulamaya konulduğu aşamadır. İnsanlar bir problemle karşılaştıklarında bu problemi, en az zaman ve çaba ile çözmek isterler. Benzer bir durum programlama için de geçerlidir. Yazılan kodun en uygun komutlardan seçilerek programın olabildiğince az satırdan oluşması programlamada tercih edilen bir durumdur.

Problemin çözülmesi ve değerlendirilmesi: Problemin çözümü için seçilen yöntemin işe konulmasının ardından başarıya ulaşılıp ulaşılmadığının değerlendirildiği aşamadır. Bu durum programlamada programın yazıldıktan sonra doğru çalışıp çalışmadığının kontrol edilmesi şeklindedir.

Problem çözme ile programlamayı ilişkilendirmekte olan bazı araştırmalar da programlama öğretiminin öğrencilerde problem çözme becerilerini geliştirmeye katkı sağladığını ortaya koymaktadır. Çetin (2012), programlama eğitiminin öğrencilerin problem çözme becerileri üzerine etkisinin olup olmadığını araştırmak için ortaokulda öğrenim gören 17 öğrenci ile deneysel bir çalışma gerçekleştirmiştir. Bu çalışma sonucuna göre, programlama eğitiminin öğrencilerin problem çözme becerilerine olumlu yönde katkı sağladığı belirlenmiştir. Alanyazında bu çalışmanın sonuçlarıyla benzerlik gösteren çalışmalar bulunmaktadır (Miller, Kelly ve Kelly, 1988; Taylor, Harlow ve Forret, 2010; Kim,

Chung ve Yu, 2011; Lai ve Yang, 2011; Kalelioğlu ve Gülbahar, 2014; Hooshyar, Ahmad, Yousefi, Yusop ve Horng, 2015).

Alanyazında programlama öğreniminin problem çözme becerisini geliştirdiğine yönelik çalışmaların yanında bu durumun tam tersi olan problem çözme becerisinin programla öğrenmeyi geliştirdiğini ifade eden çalışmalar da yer almaktadır. Başka bir deyişle, problem çözme becerisi yüksek bireylerde programlama öğrenimi daha hızlı veya kolay olabilmektedir. Çakıroğlu, Sarı ve Akkan (2011) tarafından üstün yetenekli öğrenciler ile programlama öğretiminin incelendiği bir çalışmada, programlamanın matematiksel düşünmeyi geliştirdiğini, alternatif çözüm yolları üretmeyi sağladığını ve öğrencilerin bilişsel gelişimlerinde önemli bir rol oynadığını belirtmişlerdir. Bu durumun sebebi olarak da üstün yetenekli öğrencilerin problem çözme yeteneklerinin ve zekâ düzeylerinin diğer öğrencilerden daha gelişmiş olması gösterilmektedir.

Diğer bir çalışma ise Hung (2008) tarafından yapılmış olup, bu çalışmada da problem çözme eğitiminin programlama öğrenimine etkisi araştırılmıştır. Çalışma, bilgisayar mühendisliği bölümünde öğrenim gören 48 öğrenci ile gerçekleştirilmiş olup çalışma sonucunda programlama performansı bakımından problem çözme eğitiminin verildiği deney grubu lehine anlamlı bir farklılığın olduğu ortaya çıkmıştır. Bir diğer ifade ile problem çözme becerisi arttıkça öğrencilerin programlama erişi düzeylerinde de bir artışın olduğu rapor edilmiştir.

2.4.2. Mantıksal Düşünme Becerisi

Mantıksal düşünme, kişinin bir problemle karşılaştığında kullandığı zihinsel işlemlerdir (Karplus, 1977). Bireyin sayıları etkili kullanma, karşılaştığı problemlere bilimsel çözümler üretme, kavramlar arasındaki ilişkileri ayırt etme, sınıflama, genelleme yapma, matematiksel bir formülle ifade etme gibi davranışları gösterme yeteneğidir (Demirel, 2003). Mantıksal düşünme bir sonuca varmak için kararlı biçimde düşünmeyi gerektirir (Bozdoğan, 2007). Bu düşünme modelinin temelinde ardışık düşünme vardır (Sebetci ve Aksu, 2014). Bu işlev problemle ilgili tüm fikirleri, gerçekleri ve sonuçları almak ve onları zincirleme biçimde düzene koymak demektir (Bozdoğan, 2007). Ayrıca mantıksal düşünme, problem çözmenin alt aşamalarından birisidir (Aşkar, 1989).

Yapılan tanımlamalara bakıldığında mantıksal düşünmenin karşılaşılan problemlere etkili çözüm yolları oluşturma süreci olduğu söylenebilir. Burada problemden kastedilen, sadece günlük yaşamda karşılaşılan durumlar değil, bilgisayar ortamında da karşılaşılan problemleri kapsamaktadır. Kapsamındaki bu geniş alandan dolayı öğrencilerde bu becerilerin geliştirilmesi önemli görülmektedir (Yaman ve Karamustafaoğlu, 2006; Şentürk, 2009). Bu becerinin, genetik olmayan ve sonradan öğrenilerek gelişen bir süreç olduğu belirtilebilir. Alanyazında, bu becerinin gelişmesine katkı sağlamak üzere bilgisayar programlama eğitimi üzerinde durulmaktadır (Mains, 1997; Swain, 2013).

Papert (1980) tarafından, bilgisayar programlama eğitiminin öğrencilerin mantıksal düşünme becerilerine etkisinin araştırıldığı bir çalışmada Logo programlama dili kullanılmış olup, öğrencilerin mantıksal düşünme becerilerinde anlamlı bir artış olduğu görülmüştür. Alanyazında bu sonuçla benzerlik gösteren çalışmalar da yer almaktadır (Pea ve Kurland, 1983; Dalbey ve Linn, 1985; Many, Lockard, Abrams ve Friker, 1988).

Clements ve Gullo (1984), Logo programlama dilini kullanarak 18 öğrenci ile bilgisayar programlamanın düşünme becerileri üzerindeki etkisini araştırdıkları çalışma neticesinde programlamayla uğraşan öğrencilerin bilişsel becerilerinin, programlamayla uğraşmayan öğrencilerden daha yüksek olduğunu belirtmişlerdir.

Mains'in (1997) 15 üniversite öğrencisi ile gerçekleştirdiği "programlama öğretiminin mantıksal düşünme becerisine etkisi" konulu çalışmasında QBasic programlama dilini kullanarak 14 haftalık bir eğitim yapmıştır. Süreç içerisinde derslerdeki davranışları gözlemlenen öğrencilerin, mantıksal düşünme becerilerini kullanım düzeylerinde bir artış olduğunu belirtmiştir.

21. yy becerileri olarak nitelendirilen sistematik düşünme, problem çözebilme, olaylar arasındaki ilişkileri görebilme, yaratıcı ve mantıksal düşünebilme gibi yetiler programlama ile kazandırılabilmektedir (Balım,2013; Taylan, 2013; Küçükoğlu, 2014).

Sebetci ve Aksu (2014) tarafından 142 üniversite öğrencisi ile gerçekleştirilen bir çalışma kapsamında öğrencilerin mantıksal düşünme becerileri ile programlama başarıları arasındaki ilişki incelenmiştir. Çalışmada mantıksal düşünme becerisi ile

programla başarısı arasında pozitif yönde, orta düzeyde ve istatistiksel açıdan anlamlı bir ilişki bulunmuştur. Ayrıca mantıksal düşünme becerileri geliştirildiğinde öğrencilerin programlama başarılarının da artacağı belirtilmektedir.

Yapılan çalışmalarda mantıksal düşünme becerisi ile programlama arasında anlamlı bir ilişkinin olduğu görülmektedir. Tablo 2.4'te bu ilişkinin dayandığı temeller ortaya konulmuştur.

Tablo 2.4: Mantıksal Düşünme Becerisi ve Programlama İlişkisi

Mantıksal düşünme becerisi,	***Programlama,***
Problemin çözümünü etkileyecek olası bütün faktörlerin adım adım hesaplanmasını gerektirir.	Karşılaşılan bir problemi adım adım detaylı bir şekilde tanımlamayı gerektirir. Bu süreç algoritma olarak tanımlanmaktadır.
Problemin çözümünde tüm faktörleri kontrol etmeyi ve bir faktör değiştirildiğinde diğer faktörlerin bundan nasıl etkilenebileceğini bilmeyi gerektirir.	Bir problemin çözümü için üretilen çeşitli kod satırlarının bir bütünlük oluşturacak şekilde dizilimidir. Kodların herhangi bir yerinde oluşacak bir hata, programının genelinin çalışmasını etkileyeceğinden, kodların birbirini nasıl etkileyeceği bilinmelidir.
Değişkenler arasındaki ilişkileri neden–sonuç bağlamında karşılaştırabilmeyi gerektirir.	Bir komut çalıştığında diğer komutu veya komutları nasıl etkileyeceğini bilmeyi gerektirir.
Bir olayın başlangıcından sonucuna kadar bütün aşamalarda her türlü olası durumları düşünebilmeyi gerektirir.	Mevcut bir problemin çözümüne yönelik olası bütün çözüm yollarını hesaplamayı gerektiren bir süreçtir.

2.4.3. Programlama Öğrenimine Yönelik Motivasyon

Öğrencilerin bazılarının karşılaştığı bir probleme çözüm üretmede istekli oldukları gözlenirken bazı öğrencilerin ise çözüm üretmek veya sorunla mücadele etmek yerine daha çok kaçmayı tercih ettiği sıklıkla görülen bir durumdur. Öğrenciler arasında bu farklılığın oluşumuna etki eden faktörlerin başında motivasyon gelmektedir (Akbaba, 2006). Motivasyon, bireyin hedefe ulaşmasında yardımcı olan içsel bir enerji veya zihinsel bir güç olarak tanımlanmaktadır (Sternberg ve Williams, 2002). Bireylerin hareketini tetikleyen, davranışın düzeyini belirleyen, davranışa yön veren ve davranışın devamını sağlayan bir durumdur (Deci, Kostner ve Ryan, 2001). Yani motivasyon hedefe odaklı davranışı teşvik eder ve sürekli kılar (Pintrich ve Schunk, 2002). Yapılan tanımlamalara bakıldığında, kişinin hedefe ulaşmasında tetikleyici bir mekanizmanın olduğundan bahsedilmektedir. Bu tetikleyici güç yani motivasyon iki şekilde oluşmaktadır. Bunlar içsel motivasyon

ve dışsal motivasyondur (Akbaba, 2006). İçsel motivasyon, bireyin kendi isteğiyle hareket edip çalışması şeklinde tanımlanabilir. Başka bir deyişle içsel motivasyon, bireyin kendisinde öğrenme ihtiyacı ve arzusunun oluşması durumu şeklinde tanımlanabilir. Öğrenenlerin öğrenmeye ve başarmaya karşı tutumu, dikkat düzeyi, ilgisi, hoşlanma durumu, haz duyması ve kişilik özellikleri içsel motivasyona örnek olarak verilebilir. Dışsal motivasyon ise ödül, rekabet, cezadan sakınma ve arkadaş baskısı gibi öğrenme isteğinin çevrenin etkisiyle oluşması durumudur (Dede ve Argün, 2004).

Programlamanın karmaşık yapısı ve yoğun bilişsel bir süreç gerektirmesi sebebiyle öğrencilerde zamanla başarma duygusunun azaldığı ve bu durumun da öğrencilerin içsel motivasyonlarını düşürdüğü söylenebilir. Deci, Kostner ve Ryan (2001) öğrenenlerin yeteneklerini sergileyebildiği, merak uyandıran, eğlenceli ve hedefler ile iyi ilişkilendirilmiş bir öğrenme ortamında içsel motivasyonun sağlanabileceğini vurgulamaktadır. Bu durumda programlama ortamlarının merak uyandıran ve eğlenceli bir yapıya sahip olması içsel motivasyonu arttırabilir. Ayrıca programlama öğrenirken öğrencinin üzerinde bir baskı oluşturulmaması ve öğrenenlerin pekiştirilmesi ile öğrenenlerde dışsal motivasyonun ve başarının arttırılabileceği yorumunda bulunulabilir.

Alanyazında motivasyonun başarı ile ilişkili olduğunu ortaya koyan çalışmalar (Yılmaz ve Çavaş, 2007; Yazıcı ve Altun, 2013) motivasyonu daha yüksek olan öğrencilerin derslerinde daha başarılı olduklarını belirtmektedir (Schunk, 2003; Bayraktar, 2015). Özellikle programlama gibi yoğun işlem becerileri gerektiren bir derste öğrencilerin uzun süre bilgisayar başında çalışmaları, programlamanın karmaşık yapısı, programlama kavramlarının soyutluğu ve sürekli çaba gerektirmesi gibi durumlar, öğrencilerin zamanla programlamadan soğumalarına yol açmaktadır (Ersoy, Madran ve Gülbahar, 2011). Bu durumun bir sebebi olarak, motivasyonun programlama performansını etkileyen faktörlerden birisi olması gösterilebilir.

Programlama öğretiminde yaşanan sıkıntıları gidermek ve öğrencilerin programlama dersindeki motivasyonlarını arttırmak için proje tabanlı öğrenme ortamları, işbirlikli çalışma ve oyun tabanlı öğrenme gibi farklı yaklaşımlar önerilmektedir (Başer, 2013). Bu yaklaşımlardan birisi de görsel programlama

ortamlarıdır. Görsel programlama ortamları ile kullanıcılar öğrenmede zorluk yaşadıkları soyut kod ve komutları adım adım somutlaştırarak öğrenebilmekte ve anlık dönütler alabilmektedir. Ayrıca bu ortamların öğrencilere kod yazmayı sevdirdiğini ve program yazmaya istek oluşturduğunu da söylemek mümkündür.

Yapılan bilimsel çalışmalarda elde edilen sonuçlar, görsel programlama ortamlarında öğrenci motivasyonunun daha yüksek olduğuna işaret etmektedir (Genç ve Karakuş, 2011; Ozoran, Çağıltay ve Toplallı, 2012; Erol, 2015). Ruf, Mühling ve Hubwieser (2014) tarafından yapılan bir çalışma kapsamında bir grup öğrenciye metin tabanlı programlama eğitimi verilirken diğer öğrencilere görsel programlama eğitimi verilmiştir. Her iki gruptaki öğrencilerin eğitim öncesi ve sonrası motivasyon düzeyleri ölçülmüş olup, eğitim öncesinde motivasyon yönünden bir farklılığın olmadığı belirtilmiştir. Son testlerde ise görsel programlama eğitimi alan grup lehine bir farklılığın oluştuğu yani deney grubunun motivasyon düzeyinin daha yüksek olduğu belirtilmiştir. Jiau, Chen ve Ssu (2009), programlama derslerinde öğrencilerin motivasyonlarını arttırmak için dersleri simulasyon temelli eğitsel bir oyun şekline dönüştürmüşlerdir. Yapılan eğitim sonrası öğrencilerin başarı puanları ile motivasyon puanlarını ilişkilendirerek, öğrencilerin motivasyonları arttığında daha başarılı olduklarını belirtmişlerdir.

Programlama ortamlarında öğrencilerin motivasyonlarını arttırmak için çeşitli programlama çevreleri geliştirilmektedir. İzleyen bölümde bu programlama çevreleri kısaca tanıtılmaktadır.

2.5. Görsel Programlama Çevreleri

Programlama öğretiminde kullanılmak üzere çeşitli görselleştirme programları geliştirilmiştir. Bu programların bazıları programlama ortamını görselleştirirken, bazıları da algoritmayı görselleştirmektedir. Bergin ve arkadaşları (1996) tarafından yapılan bir çalışmada programlamanın 3 farklı şekilde görselleştirilebileceği belirtilmiştir. Bunlar, program görselleştirme, algoritma görselleştirme (algoritma animasyonları) ve veri görselleştirme olarak belirtilmiştir. Cooper, Powers, McNally, Goldman, Proulx ve Carlisle (2006) ise görselleştiricileri 4 başlık altında ele almıştır. Bunlar, programlama çıktılarını görselleştiren araçlar, akış şeması modelleme araçları, görsel programlama araçları ve algoritma hikâyeleştirme araçlarıdır.

Bu çalışmada görsel programlama çevreleri iki başlık altında ele alınmıştır. Bunlar, programlama ortamını görselleştiren ve kod bloğunu bir yerden bir yere sürükle bırak mantığıyla çalışan “Blok tabanlı görsel programlama” ortamları ile kod yazmayı gerektiren ancak ekrana sonucu grafiksel olarak da yansıtabilen “Metin tabanlı programlama” ortamları olarak ele alınmıştır.

2.5.1. Blok Tabanlı Görsel Programlama

Günümüzde programlama öğrenimini kolaylaştırmak, programlamayı daha zevkli bir hale getirmek ve programlamaya yeni başlayanlara programlamayı sevdirmek için proje tabanlı öğrenme, oyun tabanlı öğrenme veya bulut tabanlı programlama gibi çeşitli yöntemler kullanılmaktadır. Bu yöntemlerden birisi de blok tabanlı görsel programlama ortamlarıdır. Bu ortamlar ses, resim ve müzik gibi birtakım medya araçlarını biraraya getirerek soyut yapıdaki programlama kavramlarını animasyon veya simulasyonlarla destekleyerek somutlaştırabilmektedir. Herhangi bir kod yazmayı gerektirmeyen arayüzleri sayesinde kullanıcılar sürükle bırak etkinlikleriyle projeler oluşturabilmektedir. Bu ortamlarda kodların bloklar halinde görsel bir yapıda olması, program yazarken kodu unutma veya bildiklerini koda dökme (Lahtinen, Ala-Mutka ve Jarvinen, 2005; Kinnunen ve Malmi, 2008), kodların veya kuralların mantığını öğrenmek yerine ezberlenmeye çalışılması (Byrne ve Lyons, 2001; İmal ve Eser, 2009) ya da programın algoritmasını oluşturmada güçlük çekilmesi gibi sorunların önüne geçilebileceği söylenebilir. Özellikle programlama öğrenmeye yeni başlayan gençlerin programlamaya olan merakını arttırarak onları araştırmaya sevk etmekte, sistematik ve mantıksal düşünme becerilerini geliştirmekte ve özgüvenlerini arttırmaktadır (Genç ve Karakuş, 2011). Ayrıca blok tabanlı görsel programlama, öğrenme ortamına dâhil edilen duyu sayısını da arttırmaktadır. Örneğin; değişken, döngü ve dizi gibi öğrenilmesinde zorluk yaşanan konuların, çeşitli görsel ve işitsel araçlar ile somutlaştırılarak öğretimi, konuların kolay bir şekilde öğrenimini sağlamaktadır (Erol, 2015). Bu bilgi “*somut bilgi daha kısa sürede kazanılır ve daha kolay öğrenilir*” ilkesiyle de örtüşmektedir (Aykaç, 2014).

Blok tabanlı görsel programlama çevrelerinden bazıları Tablo 2.5’te sunulmuştur.

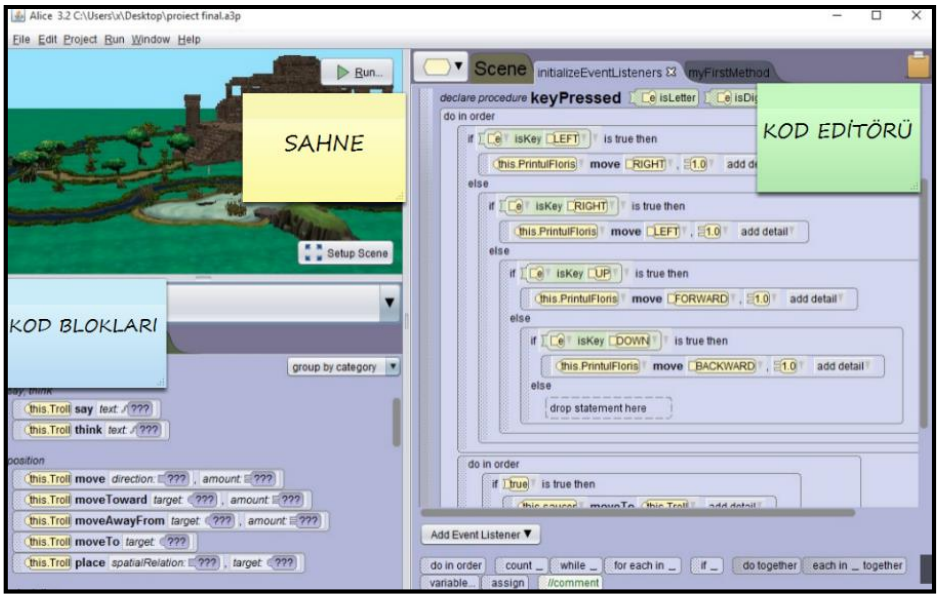
Tablo 2.5: Blok Tabanlı Görsel Programlama Ortamları

Geliştirenler	*Programlama Ortamı*	*Adresi*
Kelleher (1998)	Alice	http://www.alice.org/
Friedman ve Abelson (2009)	MIT App Inventor	http://appinventor.mit.edu/
Resnick vd. (2003)	Scratch	https://scratch.mit.edu/

Tablo 2.5'te akademik projeler tarafından desteklenen Alice, Scratch ve MIT App Inventor gibi blok tabanlı görsel programlama ortamlarının verildiği görülmektedir. Bu blok tabanlı görsel programlama ortamları ilerleyen bölümde kısaca tanıtılmaktadır.

2.5.1.1. Alice

Carnegie Mellon Üniversitesi tarafından 1998 yılında ilk versiyonu yayınlanan Alice yazılımı Bilgisayar Bilimleri dalında bir doktora çalışması kapsamında lise öğrencilerine temel programlama kavramlarını öğretmek maksadıyla geliştirilmiştir (Şendağ ve Erol, 2012). Yazılım, kullanıcılarına herhangi bir kod yazmayı gerektirmeden tamamen sürükle bırak mantığıyla 3-Boyutlu animasyon yapabilme imkânı sağlayan nesne yönelimli bir programlama dilidir. Blok tabanlı yapısı gereği, yazılan bir programın nasıl çalıştığı hemen test edilebilmekte ve kod yazımından kaynaklı hataların (sözdizimi hataları) önüne geçilebilmektedir. Ayrıca, kullanıcılarına temel programlama kavramları ile çalışabilme olanağı sunmaktadır (Kelleher, 2006). Yazılan bir programda nesneler hareket ettirilebilmekte, ses, resim, müzik ve animasyon gibi çeşitli medya araçları kullanılabilmekte, klavye veya fare ile yapılan girdilere tepkiler verdirtilebilmektedir. Yazılımın bu özellikleri sayesinde fonksiyon, döngü ve dizi gibi programlamanın karmaşık yapıları gerçek bir nesne üzerine aktarılarak somut bir öğrenme ortamı oluşturulabilmektedir (Wang, Mei, Lin, Chiu ve Lin, 2009). Şekil 2.3'te Alice yazılımının arayüzü sunulmuştur.

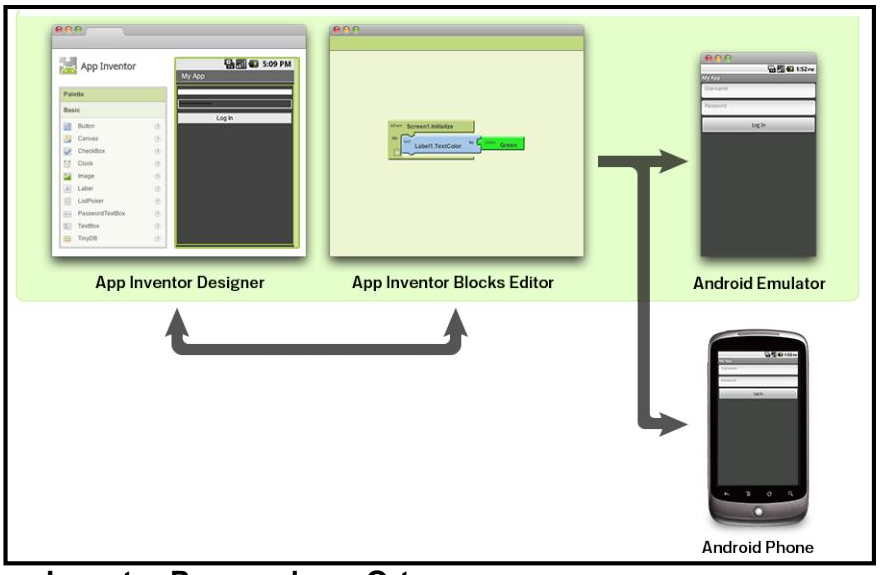


Şekil 2.3: Alice Programlama Ortamı

Tamamen ücretsiz olan Alice yazılımı, Linux, Mac veya Windows işletim sistemlerinde kullanılabilmektedir (Alice, 2016). Ayrıca yazılımın hemen her yaştan kullanıcıya hitap ettiği de belirtilebilir.

2.5.1.2. MIT App Inventor

App Inventor, Google ve MIT (Massachusetts Teknoloji Enstitüsü) işbirliğiyle geliştirilen ve Android işletim sistemli mobil cihazlara yönelik uygulamalar geliştirilmesini mümkün kılan blok tabanlı görsel bir yazılımdır (Cuiumju, 2013). Alice yazılımına benzer bir şekilde App Inventor'da da herhangi bir kod yazımı olmadan, blokları bir yerden bir yere sürükleyerek çeşitli yazılımlar geliştirilebilmektedir. Alice yazılımından farkı ise sadece Android destekli mobil cihazlara yönelik programlar geliştirilmesi (Morelli, Lanerolle, Lake, Limardo, Tamotsu ve Uche, 2011) ve bulut tabanlı program geliştirme arayüzünün (IDEs) olmasıdır (App Inventor, 2016). Kullanıcılar yazılımı kurmaksızın, İnternet bağlantısının olduğu herhangi bir yerden gmail hesabıyla oturum açarak projelerine devam edebilmektedir. Geliştirilen bir yazılımın önizlemesi, mobil cihaz ekranında ya da web tarayıcıda yapılabilmektedir. Ücretsiz olan yazılımın hemen her yaştan kullanıcıya hitap ettiği belirtilebilir. Şekil 2.4'te App Inventor yazılımının arayüzü sunulmuştur.



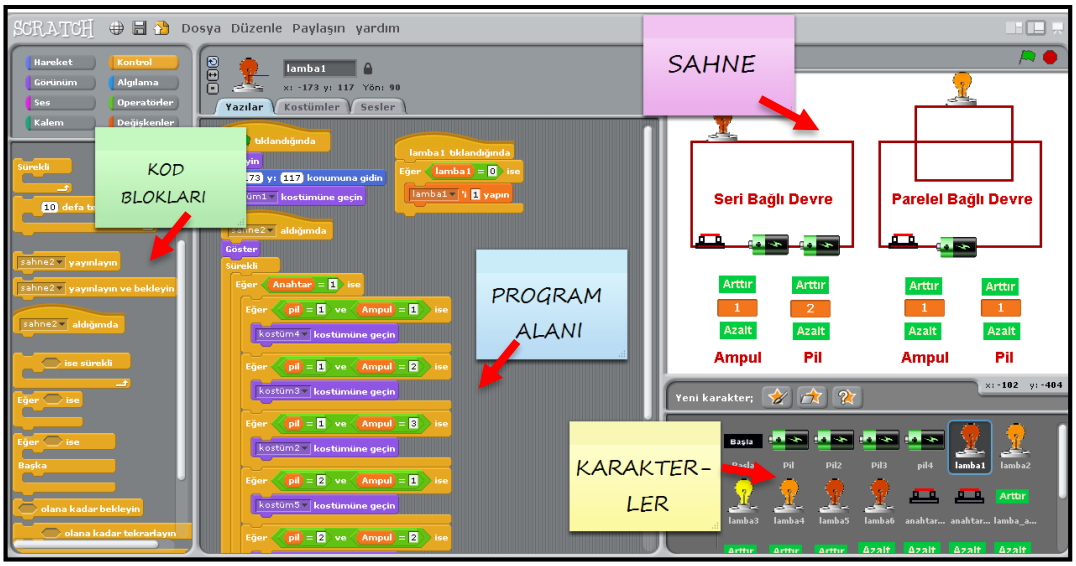
Şeki 2.4: App Inventor Programlama Ortamı

2.5.1.3. Scratch

Scratch, gençlere ve ilk defa programlama öğrenecek olan programcılara programlama öğretme maksadıyla Massachusetts Teknoloji Enstitüsü (MIT) Medya Laboratuvarı tarafından geliştirilen blok tabanlı görsel bir programlama dilidir (Yorulmaz, 2008; Scratch, 2016; Yükseltürk ve Altıok, 2016). Türkçe de dâhil olmak üzere 40'ın üzerinde dil desteğiyle Scratch, kullanıcılarına ücretsiz olarak sunulmaktadır. İlerleyen kısımda Scratch yazılımının özellikleri başlıklar halinde sunulmuştur.

Scratch'ın Genel Yapısı ve Arayüzü

Scratch yazılımı, hemen her yaştan kullanıcıya hitap eden görsel bir programlama ortamıdır (Ünal, 2015). Alice ve App Inventor yazılımlarında olduğu gibi Scratch ortamında da program yazımı, kodların bir yerden bir yere sürüklenip bırakılmasıyla gerçekleşmektedir. Birbirleriyle ilişkili olmayan bir kod diğer kod ile birleşememektedir. Bu özellik sayesinde programlamada en büyük sorunlardan birisi olan sözdizimi hataları ortadan kaldırılmış ve komut ezberlemenin de önüne geçilmiştir (Maloney, Burd, Kafai, Rusk, Silverman ve Resnick, 2004; Ismail, Ngah ve Umar, 2010; Karabak ve Güneş, 2013). Şekil 2.5'te Scratch yazılımının arayüzü sunulmuştur.



Şekil 2.5: Scratch Programlama Ortamı

Scratch yazılımının arayüzü 4 bölümden oluşmaktadır (Şekil 2.5). Ekranın sağında sahne ve sahnenin altında projedeki bütün karakterleri gösteren bir alan bulunmaktadır. Ortadaki pencere, kullanıcının seçilen karaktere kod yazdığı alandır. Soldaki pencere ise kodlama alanına sürüklenebilen kod bloklarını içermektedir.

Scratch yazılımında kod blokları kullanılarak karakterlerin görünümleri (kostümleri), sesleri, renkleri veya davranışları kontrol edilebilmektedir. Bunun yanında değişkenler, koordinatlar ve rasgele sayılar gibi birtakım matematiksel işlemler yapılabilmektedir. Ayrıca tekrarlama, koşullu yapılar ve döngüler gibi temel bilgisayımsal işlemler de yapılabilmektedir. Tüm bu işlemler, Scratch yazılımı içerisinde 8 başlık altında toplanmıştır. Bu bloklar farklı renklerde tasarlanmış olup her renk farklı kod bloklarını temsil etmektedir. Şekil 2.6'da örnek bir blok yapısı sunulmuştur.

Sarı renkli bloklar:	Kontrol menüsü
Turuncu renkli bloklar:	Değişkenler menüsü
Koyu mavi renkli bloklar:	Hareket menüsü
Mor renkli bloklar:	Görünüm menüsü
Pembe renkli bloklar:	Ses menüsü
Koyu yeşil renkli bloklar:	Kalem menüsü
Açık yeşil renkli bloklar:	Operatörler menüsü
Açık mavi renkli bloklar:	Algılama menüsü

Şekil 2.6: Scratch Menüleri

Kod bloklarının farklı renklerde olması ve uyumsuz blokların birbiriyle birleşmemesi, kod yazarken karışıklıkları (syntax, hatalı kod yazımı, noktalama işaretleri vs.) azaltmaktadır. Ayrıca adım adım çalıştır seçeneği sayesinde yeni bir blok eklendiğinde program test edilebilmektedir. Bu durum, program içerisinde hata bulmayı da kolaylaştırmaktadır.

Çoklu Ortam Desteği

Scratch yazılımı kullanıcılarına çoklu ortam destekli animasyonlar, oyunlar, simulasyonlar, müzik ve video destekli projeler geliştirme olanağı sunmaktadır (Peppler ve Kafai, 2007; Maloney, Resnick, Rusk, Silverman ve Eastmond, 2010). Çoklu ortama dayalı pogram yazmak öğrencilerin programlamaya olan ilgi ve motivasyonlarını arttırmaktadır (Osman, Loke, Zakaria ve Downe, 2012; Ruf, Mühling ve Hubwieser, 2014).

Etkileşim

Kod blokları penceresi kullanılarak nesneler hareket ettirilebilmekte, klavye veya fare ile yapılan girdilere tepki verdirtilebilmektedir. Algılama menüsü altında bulunan sensörler aracılığıyla elektronik bir devre (picoboard) programlanabilmektedir. Örneğin, ses sensörü kullanılarak, gerçek ortamda bir ses olduğunda karakterin tepkilerinde değişiklik yapılabilir. Yine ışık sensörü kullanılarak, devrenin üzerindeki lambanın yanması sağlanabilir. Klavye yön tuşları kullanılarak, gerçek ortamdaki bir nesnenin (tekerlek, araba vs.) hareket etmesi sağlanabilir. Bu özelliği ile Scratch yazılımının kullanıcılarını üreten bireyler olmaya ve yaratıcı düşünmeye teşvik ettiğini söylemek mümkündür.

Paylaşım ve İşbirlikli Çalışma

Scratch yazılımı hem bilgisayara kurulabilme hem de İnternet üzerinden çalışabilme kolaylığı sağlamaktadır. Scratch ile yapılan bir proje İnternet üzerinden paylaşılabilmektedir. Bu şekilde kullanıcılar birbirleriyle fikir alışverişinde bulunabilmekte ya da işbirlikli çalışma gerçekleştirebilmektedir. İnternet sitesine yüklenen projeler diğer kullanıcılar tarafından incelenebilmekte, ekleme veya düzeltmeler yapılarak yeniden paylaşılabilmektedir. Bu sayede kullanıcıların programlamanın nasıl yapıldığını keşfetmelerine (Romeike, 2007) ve hataları kendilerinin bulup düzeltmelerine olanak sağlanmaktadır (Kert ve Uğraş, 2009).

Farklı Derslerde Kullanımı

Scratch yazılımı ile çoklu ortam destekli projeler, oyunlar, simulasyon ya da animasyonlar oluşturulabilmektedir. Bu bakımdan Scratch, Algoritma ve Programlama derslerinin haricinde; Yabancı Dil, Matematik ve Fen Bilgisi gibi derslerde de kullanılabilir. Bu dersler çerçevesinde öğreniminde zorluk çekilen birtakım konular Scratch projeleri ile eğlenceli hale getirilip bu konular öğrencilere oyun oynayarak öğretilebilir. Örneğin, Scratch ile yapılan ve öğrencinin işlem becerisini geliştirmeyi amaçlayan bir matematik dersi materyali geliştirilebilir. Bu materyal sayesinde öğrenciye eğitsel bir oyun sunulmuş olacaktır.

Diğer Programlama Ortamlarına Uygunluğu

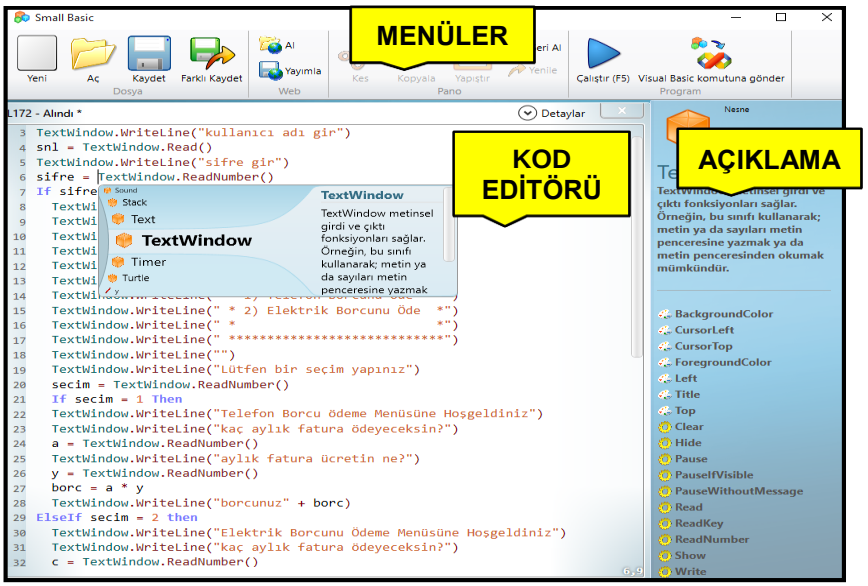
Scratch yazılımında kodlar yerine bloklar kullanılsa da bu bloklar programlama dillerinde kullanılan yapıları içinde barındırmaktadır (değişken, döngü, koşul yapıları vs.). Diğer programlama dillerinden farklı olarak öğrenciler Scratch yazılımında kod yazmak yerine kod bloklarını sürükle bırak yöntemiyle birleştirerek programlar oluşturmaktadır. Bunun dışında kod ya da komutların çalışma mantığı diğer programlama dilleriyle benzerlik göstermektedir. MIT, Harvard Üniversitesi, Berkeley Üniversitesi ve California Üniversitesi gibi üniversitelerde programlamaya giriş dersinde Scratch yazılımının kullanılıyor olması (Resnick, Maloney, Monroy-Hernandez, Rusk, Eastmond, Brennan ve Kafai, 2009) da "*Scratch'ın diğer programlama dilleriyle uyumlu olduğu*" ifadesini desteklemektedir.

2.5.2. Metin Tabanlı Programlama

2.5.2.1. Small Basic

Microsoft firması tarafından programlamaya yeni başlayanlara yönelik olarak geliştirilen Small Basic yazılımı, açık kaynak kod özelliğine ve oldukça işlevsel bir içeriğe sahiptir (Kocaman, 2010). Yazılım, BASIC programlama dilinin basitleştirilmiş bir versiyonudur (Akçay, 2009). Small Basic ile geliştirilen bir programa ait kodları Visual Basic yazılımına aktarmak ve çalışmaya orada devam etmek mümkündür. Bunun yanısıra yazılan bir programın kodlarını web ortamında yayınlamak ve farklı insanların yayınladıkları program kodlarını yine web ortamından programa dâhil etmek mümkündür. Ücretsiz olan yazılımın hemen her yaştan kullanıcıya hitap ettiği de belirtilebilir.

Small Basic yazılımı, blok tabanlı programlama ortamlarından farklı olarak kod yazımını gerektirmektedir. Blok tabanlı bir yapıya sahip olmayan bu yazılım ile kullanıcıların program geliştirebilmek için çeşitli kod veya komutları doğru bir şekilde yazmaları gerekmektedir. Aksi halde sözdizimi hataları oluşabilmektedir. Small Basic yazılımına ait bir görünüm Şekil 2.7'de sunulmuştur.

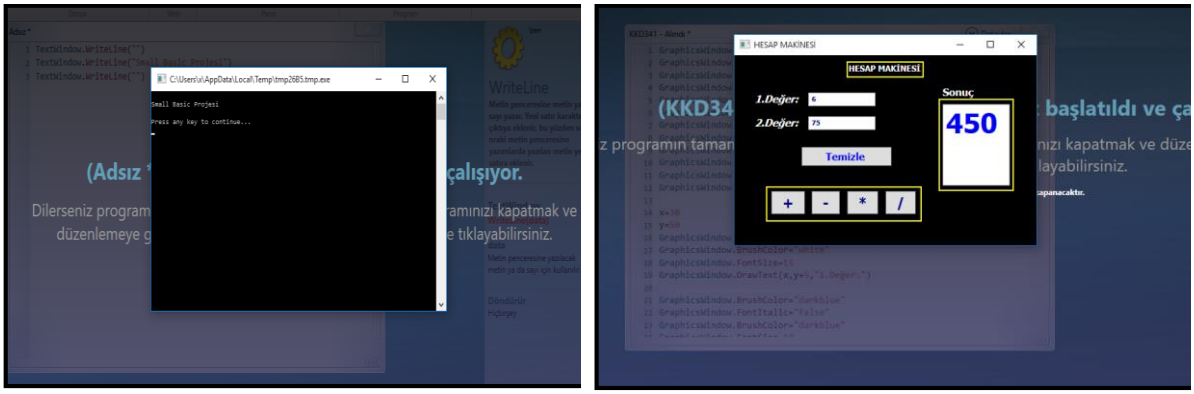


Şekil 2.7: Small Basic Programlama Ortamı

Small Basic yazılımı 3 kısımdan oluşmaktadır. Kod editörü paneli kullanılarak program kodları metinsel olarak yazılır. Kod yazarken yazılım, açılır bir menü ile kod önerisinde bulunmaktadır. Bu özelliğin kod unutmayı engellediğinden bahsedilebilir. Açıklama paneli seçilen bir komutun ne işe yaradığının ve hangi

komutlarla birlikte kullanılabileceğinin gösterildiği yardımcı kısımdır. Ekranın üstünde ise menülerin olduğu bölüm yer alır. Yazılan kodlar bu bölüm aracılığı ile İnternet üzerinden paylaşılabilir, bir programa ait kodlar İnternet'ten editöre aktarılabilir veya kodlar Visual Basic programlama ortamına taşınabilir.

Small Basic ile yazılan kodların önizlemesi komut satırında ya da grafik arayüzünde yapılabilmektedir. Bu yönüyle Small Basic, hem grafik ekrana hem de komut satırı penceresine çıktı veren bir yazılımdır. Şekil 2.8'de Small Basic yazılımına ait çıktı ekranları sunulmuştur.



Şekil 2.8: Small Basic Çıktı Ekranları

2.6. Blok Tabanlı Görsel Programlama (Scratch) Öğretiminin Katkıları

Bu bölümde blok tabanlı görsel programlama (Scratch) öğretiminin katkılarıyla ilgili ilköğretim, ortaöğretim ve üniversite düzeylerinde yapılan bilimsel çalışmalar incelenmiştir. Araştırmanın çalışma grubunu oluşturan öğrenciler üniversite öğrencileri olduğundan, ilköğretim ve ortaöğretim düzeylerinde yapılan çalışmalar özetlenerek tablolar halinde sunulmuştur.

2.6.1. İlköğretim Düzeyinde Yapılan Çalışmalar

Scratch yazılımı kullanılarak ilköğretim düzeyinde gerçekleştirilen çalışmalarda araştırılan değişkenler ve ulaşılan sonuçlar Tablo 2.6'da sunulmuştur.

Tablo 2.6: İlköğretim Düzeyinde Yapılan Çalışmalar ve Ulaşılan Sonuçlar

		Programlama öğrenimi (erişi)	*Problem çözme becerisi*	*Motivasyon*	*Matematiksel düşünme becerisi*	*Yaratıcı Düşünme Becerisi*	*Tutum*	*Mantıksal düşünme becerisi*	*Fen öğrenimi (science learning)*	*Öz düzenleme becerisi*	*Dil eğitimi (ingilizce)*	*SONUÇ*
1	Gülmez (2009)	X		X								+-
2	Nam, Kim ve Lee (2010)		X									+
3	Lai ve Yang (2011)		X					X				+-
4	Lai ve Lai (2012)		X					X	X			+++
5	Ferrer-Mico, Prats-Fernandez ve Redo-Sanchez (2012)									X		+
6	Wilson, Hainey ve Connolly (2012)	X										+
7	Brown, Mongan, Kusic vd. (2013)		X				X					++
8	Fessakis, Gouli ve Mavroudi (2013)		X									+
9	Nikiforos, Kontomaris ve Chorianopoulos (2013)						X					+
10	Ke (2014)				X							+
11	Su, Yang vd. (2014)	X										+
12	Tekerek ve Altan (2014)	X										-
13	Wyffles, Martens ve Lemmens (2014)						X					+
14	Kalelioğlu ve Gülbahar (2014)		X									-
15	Shin ve Park (2014)		X									+
16	Kobsiripat (2014)					X						+
17	Wang, Huang ve Hwang (2014)	X	X	X			X					+---
18	Ruf, Mühling ve Hubwieser (2014)	X		X								++
19	Kukul ve Gökçearslan (2014)		X									-
20	Ersoy ve Aydın (2015)	X										+
21	Su, Huang, Yang ve Hsieh (2015)	X										+
22	Calao, Moreno-Leon, Correa ve Robles (2015)		X		X							++
23	Moreno-Leon ve Robles (2015)										X	+
24	Akpınar ve Aslan (2015)	X										+
25	Korkmaz (2016)	X	X					X				+++
26	Ortiz-Colon ve Romo (2016)	X		X								++

(**+,-** : Scratch yazılımı incelenen değişken(ler) bakımından anlamlı bir farklılık yaratmıştır / farklılık yaratmamıştır)

Tablo 2.6 incelendiğinde, Scratch yazılımının çeşitli değişkenler üzerindeki etkisini araştıran çalışmaların son 10 yıl içerisinde ve farklı farklı değişkenler üzerinde yoğunlaştığı söylenebilir. Bunun yanında ilköğretim düzeyinde incelenen 26 çalışmaya ve ulaşılan sonuçlara genel çerçevede bakıldığında;

1. Scratch yazılımının programlamanın temel yapıları ve programlama mantığını kazandırmada etkili bir yazılım olduğu,

2. Problem çözme, mantıksal düşünme ve yaratıcılık gibi üst düzey düşünme becerileri üzerinde anlamlı bir etkiye sahip olduğu,

3. Eğlenceli, basit ve anlaşılır arayüzü ve çoklu ortam desteği sayesinde öğrencilerin tutum ve motivasyonlarını arttırdığı,
4. Scratch yazılımının sadece programlama öğretimi amacıyla değil, diğer derslerde de kullanılabileceği şeklinde yorumlarda bulunulabilir.

2.6.2. Ortaöğretim Düzeyinde Yapılan Çalışmalar

Scratch yazılımı kullanılarak ortaöğretim düzeyinde gerçekleştirilen çalışmalarda araştırılan değişkenler ve ulaşılan sonuçlar Tablo 2.7'de sunulmuştur.

Tablo 2.7: Ortaöğretim Düzeyinde Yapılan Çalışmalar ve Ulaşılan Sonuçlar

		Programlama öğrenimi	*Problem çözme becerisi*	*Motivasyon*	*Transfer becerisi*	*Mantıksal düşünme becerisi*	*Yabancı dil öğrenimi*	*SONUÇ*
1	Maloney, Peppler, Kafai, Resnick ve Rusk (2008)	**X**						**+**
2	Meerbaum-Salant, Armonia ve Ben-Aria (2010)	**X**						**+**
3	Wang ve Zhou (2011)	**X**			**X**	**X**		**+++**
4	Osman, Zakaria, Loke ve Downe (2012)	**X**		**X**				**++**
5	Nikou ve Economides (2014)			**X**				**+**
6	Sanjanaashree, Anand ve Soman (2014)						**X**	**+**
7	Giordano ve Mairona (2014)	**X**						**+**
8	Giordano ve Mairona (2015)	**X**	**X**					**-+**
9	Fields, Vasudevan ve Kafai (2015)	**X**						**+**
10	Koorsse, Ciliers ve Calitz (2015)	**X**						**-**
11	Ouahbi, Kaddari, Darhmaoui, Elachqar ve Lahmine (2015)	**X**		**X**				**++**
12	Konak (2016)	**X**						**+**

(**+**,**-** : Scratch yazılımı incelenen değişken(ler) bakımından anlamlı bir farklılık yaratmıştır / farklılık yaratmamıştır)

Scratch yazılımı kullanılarak ortaöğretim düzeyinde gerçekleştirilen çalışmalarda, ilköğretimde olduğu gibi yapılan çalışmaların son 10 yılda yoğunlaştığı görülmektedir. Yapılan 12 çalışma neticesinde ulaşılan sonuçlara bakılarak;

1. İlköğretimde olduğu gibi ortaöğretim düzeyinde de Scratch yazılımının programlama öğrenimi üzerinde olumlu bir etkiye sahip olduğu,
2. Öğrencilerin programlamaya yönelik motivasyonlarını arttırdığı,
3. Mantıksal düşünme ve problem çözme becerilerinin gelişimine katkı sağladığı,
4. Scratch ile programlama eğitimi alan öğrencilerin diğer programlama dillerini daha kolay öğrenebildiği,

5. Farklı derslerde de kullanımının olduğu şeklinde yorumları yapmak mümkündür.

2.6.3. Üniversite Düzeyinde Yapılan Çalışmalar

Alanyazında Scratch yazılımının çeşitli beceriler üzerinde etkililiğini araştıran çok sayıda çalışma olmasına rağmen üniversite düzeyinde yapılan çalışmalarda daha çok programlama derslerinde kullanımına yoğunlaşılmıştır. Bu kapsamda üniversite düzeyinde yapılan çalışmalardan örnekler aşağıda sunulmuştur.

Malan ve Leither (2007) tarafından yapılan bir çalışmada, Scratch yazılımı yükseköğretimde Bilgisayar Bilimleri dersi kapsamında kullanılmıştır. Scratch ile giriş yapılıp Java programlama diline geçiş yapılmıştır. Dönem başında öğrencilerin programlama ön bilgi düzeyleri ölçülmüş ve 25 katılımcıdan %52'sinin herhangi bir programlama ön bilgisine sahip olmadığı belirtilmiştir. Buna ilaveten öğrencilerden %32'sinin orta düzeyde ön bilgiye sahip olduğu, %16'sının ise iyi düzeyde programlama bilgisine sahip olduğu belirtilmiştir. Dönem sonunda Scratch ile yapılan eğitimin programlama ön bilgisine sahip olan ve olmayan öğrencilerin programlama başarılarını nasıl etkilediği araştırılmıştır. Araştırma sonucunda Scratch yazılımının 25 katılımcıdan %76'sına pozitif bir etkisi olduğu belirtilmiştir. Scratch, katılımcıların %16'sında herhangi bir etki oluşturmazken, %2'sinde de negatif bir etki oluşturduğu raporlanmıştır.

Kereki (2008) tarafından üniversite öğrencileri ile yapılan bir çalışmada deney ve kontrol grupları oluşturularak deney grubundaki öğrencilerle Scratch ile programlama etkinlikleri yapılmıştır. Kontrol grubunda ise programlama dersi, müfredatta tanımlı olan şekilde işlenmiştir. Bilgisayar Bilimleri (CS1) dersinde yapılan bu çalışmada öğrencilerin programlama deneyimleri ve motivasyon düzeyleri ölçülmüştür. Çalışmanın sonuçlarına bakıldığında ön test ve son test puanları bakımından gruplar arasında anlamlı bir farklılık çıkmazken Scratch ile eğitim alan deney grubunun motivasyon düzeyi kontrol grubuna kıyasla daha yüksek çıkmıştır.

Fesakis ve Serafeim (2009), Scratch yazılımının "Bilgisayar Bilimine Giriş" dersi kapsamında kullanımına yönelik olarak üniversite öğrencilerinin tutum ve görüşlerini tespit etmeyi amaçlamıştır. Bu amaç doğrultusunda çalışma 35 öğrenci ile gerçekleştirilmiştir. Öğrencilere verilen Scratch eğitimlerinin ardından dönem

sonu tutum ve görüş anketleriyle veriler toplanmıştır. Sonuçlara bakıldığında öğrencilerin yaklaşık %80'i Scratch programlamaya yönelik olumlu bir görüşe sahip olduğunu belirtmiştir. Ayrıca Scratch'ın bilgisayar programlama ve bilişim eğitiminde pozitif bir etkiye sahip olduğu düşünülmektedir.

Rizvi, Humphries, Major, Lauzun ve Jones (2011) tarafından üniversite birinci sınıfta öğrenim gören 35 öğrenciyle gerçekleştirilen bir çalışmada matematik başarısı yüksek olmayan öğrencilerle Bilgisayar Bilimleri (CS0) dersinde Scratch ile programlama etkinlikleri yapılmıştır. Araştırmacılar dönem sonunda öğrencilerin programlama özyeterlilikleri ile programlamaya yönelik tutumlarındaki değişimi incelemiştir. İlerleyen yılda ise CS0 dersini alan Scratch grubu (23 öğrenci) ve CS0 dersini almayan diğer bir grup (42 öğrenci) ile Bilgisayar Bilimleri (CS1) dersinde programlama eğitimine devam etmişlerdir. Ardından her iki grupta da öğrencilerin programlama başarıları ölçülmüştür. İki yıl süren çalışmanın sonunda, Scratch'ın CS0 dersinde öğrencilerin programlamaya karşı tutumunu ve programlama öz yeterliliğini anlamlı şekilde arttırdığı belirtilmiştir. Programlama başarısına bakıldığında ise Scratch eğitimi alan grubun %74'ü CS1 dersinde başarılı olurken, Scratch eğitimi almayan öğrencilerden sadece %39'u başarı göstermiştir.

Genç ve Karakuş (2011) tarafından "Eğitimde Bilgisayar Oyunları Tasarımı" dersi kapsamında öğrencilerin deneyimlerini ve görüşlerini belirlemek amacıyla Scratch etkinlikleriyle desteklenen bir çalışma gerçekleştirilmiştir. Çalışma, ikinci sınıfta öğrenim gören 109 üniversite (BÖTE) öğrencisi ile gerçekleştirilmiştir. Çalışma kapsamında elde edilen nitel veriler araştırmacıların bloglarındaki anketlerden, nicel veriler ise öğrenci blog girdilerinden toplanmıştır. Toplanan veriler sonucunda öğrencilerin %79'u Scratch kullanımını kolay bulduklarını, program yazarken kendilerini rahat hissettiklerini ve Scratch'ın eğlenceli bir arayüze sahip olduğunu belirtmişlerdir. Öğrencilerin yaklaşık %20'si ise Scratch ile programlama deneyiminden hoşlanmadıklarını rapor etmişlerdir. Öğrencilerin %73'ü Scratch ile programlama yapılarının öğreniminin ve anlaşılmasının diğer programlama dillerine göre daha kolay olduğunu ve Scratch'ın eğitsel bir araç olarak kullanılabileceğini belirtmişlerdir.

Ozoran, Çağıltay ve Topallı (2012) mühendislik öğrencileri ile gerçekleştirdikleri bir çalışmada, programlama derslerinde C programına paralel olarak Scratch

yazılımını kullanmışlardır. Öğrenciler iki dönem boyunca ders esnasında C programlama dilini öğrenirken, laboratuvar etkinliklerinde Scratch yazılımını kullanmışlardır. Çalışma kapsamında bir yıl önce (2011) programlama dersini alan farklı öğrencilerin final sınavı başarıları ile uygulamanın yapıldığı yılda (2012) bu dersi alan öğrencilerin başarılarını karşılaştırmak amaçlanmıştır. Bu amaç doğrultusunda, her iki yılda da aynı sınav soruları kullanılmıştır. Yapılan karşılaştırma sonucunda 2012 yılında programlama dersinde başarısız olan öğrenci sayısında önceki yıla göre %19 oranında bir azalma görülmüştür.

Erol (2015) tarafından gerçekleştirilen doktora çalışmasında Bilişim Teknolojileri öğretmen adaylarına yönelik olarak yapılan programlama öğretiminde Scratch kullanımının motivasyon ve başarıya olan etkisi araştırılmıştır. Araştırmaya üniversite ikinci sınıfta öğrenim gören 52 öğrenci katılmıştır. Bu öğrencilerden yansız atama ile deney ve kontrol grupları oluşturulmuştur. Eğitimin ilk yedi haftalık bölümünde programlama mantığının kazandırılması ve temel programlama yapılarının öğretilmesi amacıyla deney grubundaki katılımcılar Scratch ile oyun tasarımı etkinlikleri, kontrol grubunda yer alan katılımcılar ise mevcut ders programındaki haliyle problem çözme etkinlikleri yapmıştır. Uygulamanın ikinci yedi haftalık bölümde ise her iki grupta da aynı yöntem kullanılarak C# programlama diline geçiş yapılmıştır. Araştırmada veri toplama araçları olarak başarı ve motivasyon testleri ile odak grup görüşmesi formu kullanılmıştır. Araştırma sonucu elde edilen bulgulara göre, katılımcıların motivasyon puanlarının ön testte her iki grupta da benzer olduğu, son testlerde ise deney grubu lehine anlamlı bir farklılık oluştuğu belirtilmiştir. Benzer şekilde katılımcıların programlama başarı puanlarında da ön testte her iki grup puanlarının benzer olduğu, son testlerde ise deney grubu lehine anlamlı bir farklılık oluştuğu raporlanmıştır. Araştırmanın nitel verileri incelendiğinde ise deney grubunda yer alan katılımcılar Scratch ile oyun tasarımı etkinliklerinin eğlenceli ve kolay olduğunu, ders süresince yapılan etkinliklerin programlama mantığını kazandırmada ve motivasyonu arttırmada etkili olduğunu, ancak bazı temel yapılar için yetersiz olduğunu dile getirmişlerdir. Kontrol grubunda yer alan katılımcılar ise akış diyagramları ile problem çözme sürecinin zor ve sıkıcı olduğunu, ders süresince yapılan etkinliklerde aktif olamama ve uygulamanın olmaması gibi sınırlılıkların motivasyonlarını düşürdüğünü belirtmişlerdir. Ayrıca deney grubunda

yer alan katılımcılar Scratch ile oyun tasarımı sürecinde edindikleri deneyimin, C# programlama öğrenme sürecinde programlama başarılarına katkısının olduğunu belirtmişlerdir.

Demir (2015) tarafından gerçekleştirilen bir çalışmada eğitsel programlama dili olarak Scratch yazılımı kullanımının öğrencilerin programlama başarısı ve kaygı düzeylerine olan etkisi araştırılmıştır. Bu amaç doğrultusunda bir devlet üniversitesinin programlama bölümünde öğrenim gören 87 öğrenci, çalışma öncesinde 3 gruba ayrılmıştır. Birinci grupta eğitsel programlama dili (Scratch) dersin uygulama kısmına, ikinci grupta kuramsal kısmına, üçüncü grupta ise hem kuramsal hem de uygulama kısmına entegre edilerek gruplar oluşturulmuştur. Veriler, programlama kaygı ölçeği ve başarı testi kullanılarak toplanmıştır. Programlamaya yönelik kaygı ve başarı açısından ön test ve son test sonuçlarına bakıldığında, eğitsel programlama dilinin hem kuramsal derse hem de uygulamaya entegre grup lehine anlamlı bir farklılık oluşturduğu görülmüştür. Bu bulgular, eğitsel programlama dilinin dersin hem kuramsal hem de uygulama kısmına entegre edilmesinin bilişsel ve üst düzey bilişsel beceriler açısından öğrencilerin programlamaya yönelik akademik başarılarını ve bilgisayar programlamaya yönelik kaygılarını olumlu yönde etkilediğini göstermektedir. Akademik başarıyı arttırmak ve bilgisayar programlamaya yönelik kaygıyı azaltmak için Scratch yazılımının, dersin hem kuramsal hem de uygulama kısmına entegre edilerek kullanılabileceği de belirtilmektedir.

Scratch yazılımıyla ilgili üniversite düzeyinde yapılan çalışmalarda ağırlıklı olarak programlama derslerinde kullanımı üzerine yoğunlaşılmıştır. İncelenen çalışmalarda Scratch ile programlama öğrenmenin programlama başarısı ve programlamaya yönelik motivasyon üzerinde pozitif bir etkiye sahip olduğu sonuçlarına ulaşılmıştır. Ayrıca öğrencilerin Scratch'ın programlama derslerinde kullanılmasına yönelik olarak olumlu görüşlere sahip olduğu da ulaşılan diğer sonuçlar arasındadır.

3. YÖNTEM

Bu bölümde araştırma yöntemi, değişkenler, çalışma grubu, veri toplama araçları, uygulama süreci, deneysel çerçeve, verilerin analizi ile çalışmanın iç ve dış geçerliliği açıklanmaktadır.

3.1. Araştırmanın Yöntemi

Bu çalışmada yöntem olarak nicel araştırma yaklaşımlarından tek gruplu ön test son test deney modeli kullanılmıştır (Fraenkel ve Wallen, 2003). Tek gruplu ön test son test deneysel desende deneysel işlemin etkisini test etmek için bir grupla çalışılır ve katılımcıların bağımlı değişkene ilişkin ölçümleri uygulama öncesinde ön test, uygulama sonrasında son test olarak aynı katılımcılardan elde edilir (Karasar, 2010).

Yapılan araştırmada iki farklı deneysel işlem gerçekleştirilmiş ve sonuçlar ayrı ayrı değerlendirilmiştir. 30 öğrenciden oluşan A şubesinde blok tabanlı görsel programlama (Scratch) öğretimi yapılmış olup incelenen değişkenlere ait ön test puanlarının son testlerde anlamlı bir artış gösterip göstermediği araştırılmıştır.

Diğer deneysel işlem ise 30 öğrencinin yer aldığı B şubesinde gerçekleştirilmiştir. Bu grupta metin tabanlı programlama (Small Basic) öğretimi yapılmış olup incelenen değişkenlere ait ön test puanlarının son testlerde anlamlı bir artış gösterip göstermediği araştırılmıştır.

Blok tabanlı görsel programlama öğretiminin yapıldığı grupta kullanılan yöntemin simgesel gösterimi Tablo 3.1'de, metin tabanlı programlama öğretiminin yapıldığı grupta kullanılan yöntemin simgesel gösterimi ise Tablo 3.2'de sunulmuştur.

Tablo 3.1: Tek Gruplu Ön Test Son Test Deseni (Blok Tabanlı Görsel Programlama)

Grup	*Ön test*	*Uygulama*	*Son test*
A Şubesi - Blok tabanlı görsel programlama (30 öğrenci)	O_1	Blok tabanlı görsel programlama öğretimi	O_2

Tablo 3.2: Tek Gruplu Ön Test Son Test Deseni (Metin Tabanlı Programlama)

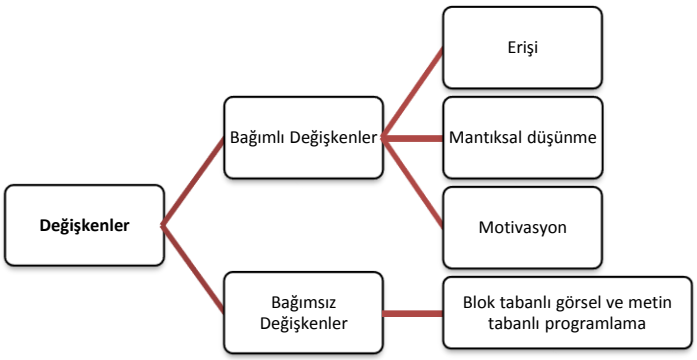
Grup	*Ön test*	*Uygulama*	*Son test*
B Şubesi - Metin tabanlı programlama (30 öğrenci)	O_1	Metin tabanlı programlama öğretimi	O_2

O_1: Ön test

O_2: Son test

3.2. Değişkenler

Araştırmanın bağımsız değişkenleri, blok tabanlı görsel programlama ve metin tabanlı programlama öğretimleridir. Bağımlı değişkenler ise erişi, mantıksal düşünme ve motivasyondur. Değişkenlere ilişkin şekilsel gösterim Şekil 3.1.'de sunulmuştur.



Şekil 3.1: Bağımlı ve Bağımsız Değişkenler

3.3. Çalışma Grupları

Araştırmanın çalışma grubunu 2015-2016 Eğitim-Öğretim yılının Bahar döneminde Akdeniz Bölgesi'nde yer alan bir devlet üniversitesinin Fen Bilgisi Öğretmenliği Bölümü'nde öğrenim gören 2. sınıf öğrencileri oluşturmaktadır.

3.3.1. Çalışma Gruplarının Özellikleri

Araştırmaya toplam 70 öğrenci katılmıştır. Ancak süreç sonunda her iki grupta toplam 10 öğrenci devamsızlık ve testleri eksik doldurma gibi sebeplerle değerlendirme dışında tutulmuştur. Analiz için her iki grupta 30'ar öğrencinin verileri kullanılmıştır. Tablo 3.3'te çalışma gruplarının özellikleri sunulmuştur.

Tablo 3.3: Çalışma Gruplarında Yer Alan Öğrencilerle İlgili Bilgiler

		A Şubesi - Blok Tabanlı Görsel Programlama		*B Şubesi - Metin Tabanlı Programlama*	
	Özellik	***n***	***%***	***n***	***%***
Cinsiyet	Kadın	22	73	20	67
	Erkek	8	27	10	33
	Toplam	**30**	**100**	**30**	**100**
Mezun olunan okul türü	Genel Lise	19	64	18	60
	Anadolu Lisesi	10	33	12	40
	Fen Lisesi	-	-	-	-
	Meslek Liseleri	-	-	-	-
	Anadolu Öğret. Lisesi	1	3	-	-
	Toplam	**30**	**100**	**30**	**100**
Daha önce kodlama, algoritma veya programlama ile ilgili bir eğitim alındı mı?	Evet	0	0	0	0
	Hayır	30	100	30	100
	Toplam	**30**	**100**	**30**	**100**

Tablo 3.3 incelendiğinde, blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin yapıldığı sınıflardaki öğrencilerin cinsiyet açısından birbirlerine benzer oldukları görülmektedir. Öğrencilerin mezun oldukları lise türlerine bakıldığında, benzer liselerden mezun oldukları ve Fen lisesi ya da Meslek liselerinden herhangi bir öğrencinin bulunmadığı görülmektedir. Bu durumdan hareketle, grupların lisede benzer eğitimleri aldıkları ve eğitim düzeyleri açısından benzer özelliklere sahip oldukları görülmektedir. Lise ya da daha önceki eğitim dönemlerinde veya kişisel gelişim anlamında kodlama, algoritma ya da programlama gibi bir eğitimi alıp almadıklarına yönelik verilen dağılıma bakıldığında ise tüm öğrencilerin programlamayla ilgili bir eğitimlerinin olmadığı görülmektedir. Bu durum, çalışmanın daha önce herhangi bir zamanda programlama eğitimi almamış öğrencilerle yürütüldüğünü göstermektedir.

3.4. Veri Toplama Araçları

Araştırma kapsamında öğrencilerden veri toplamak amacıyla kullanılan ölçme araçları şunlardır:

1. Öğrencilerin kişisel özellikleri ve programlamaya ait yaşantıları hakkında bilgi almak amacıyla "*Kişisel bilgi formu*",
2. Öğrencilerin algoritma ve programlama erişi düzeylerini ölçmek için *"Programlama erişi testi",*

3. Öğrencilerin mantıksal düşünme düzeylerini ölçmek için *"Mantıksal düşünme testi",*
4. Öğrencilerin programlamaya yönelik motivasyonlarını ölçmek için "*Motivasyon ölçeği"*.

Aşağıda veri toplama araçlarının geliştirilmesi ve uygulanmasıyla ilgili bilgiler verilmiştir.

3.4.1. Kişisel Bilgi Formu

Araştırmacı tarafından hazırlanan kişisel bilgi formu; katılımcıların cinsiyetini, mezun oldukları okul türünü, daha önce kodlama, algoritma veya programlama ile ilgili bir eğitim alıp almadıklarını belirlemeye yönelik sorulardan oluşmaktadır.

3.4.2. Programlama Erişi Testi

Çalışmada kullanılan "Programlama Erişi Testi" araştırmacı tarafından geliştirilmiştir. Bu süreçte alanyazından yararlanılarak klasik ve çoktan seçmeli olmak üzere toplam 30 soruluk bir havuz oluşturulmuştur.

Oluşturulan sorular, iki üniversitenin BÖTE bölümlerinde programlama eğitimi veren 4 akademisyene ve dil açısından bir akademisyene sunularak dönüt ve düzeltmeler alınmıştır. Bu aşamada "benzer başka soru olması", "soruların düzeye uygun olmayışı" ve "seçenek oluşturulamaması" gibi nedenlerle bazı sorular elenmiş ve test 20 soruya (4 klasik, 16 test) düşürülmüştür.

Bu aşamadan sonra oluşturulan test, çalışmanın yapıldığı Fen Bilgisi Öğretmenliği programında öğrenim gören ve daha önce programlama eğitimi almış 174 öğrenciye uygulanmıştır. Öğrenciler testten aldıkları puanlara göre sıralanmış, madde analizi yapmak için %27'lik alt ve %27'lik üst grup seçilerek gerekli işlemler yapılmıştır. Bir sorunun madde güçlük indeksi 1'e yakın ise kolay, 0,50 civarında ise orta, 0,00'a yakın ise zor olarak kabul edilir (Bayrakçeken, 2007). Maddenin ayırıcılık indeksi 0,40 ve daha yüksek değerde olan maddeler için çok iyi, 0,30 ile 0,39 değerleri arasında olan maddeler için oldukça iyi, 0,20 ile 0,29 değerleri arasında olan maddeler için düzeltilmesi ve geliştirilmesi gerektiği, 0,19 ve daha düşük değerde olan maddeler için çok zayıf ve testten çıkarılması gerektiği şeklinde değerlendirilebilir. Bu doğrultuda 7. soru (0,11) ve 14. soru (-0,02)

elenerek testteki soru sayısı 18'e düşürülmüştür. Testin son haline ait ayırıcılık indeksi değerleri ve madde güçlük indeksi değerleri Tablo 3.4'te sunulmuştur.

Tablo 3.4: Geliştirilen Erişi Testinin Pilot ve Son Haline Ait Madde İstatistikleri

Pilot Çalışma			*Testin Son Hali*				
Madde No	***Güçlük indeksi***	***Ayırıcılık indeksi***	***Madde No***	***Güçlük indeksi***	***Ayırıcılık indeksi***	***Varyans***	***Standart sapma***
1	0,34	0,55	**1**	0,33	0,53	0,22	0,47
2	0,33	0,66	**2**	0,32	0,64	0,22	0,47
3	0,26	0,51	**3**	0,26	0,51	0,19	0,44
4	0,38	0,64	**4**	0,37	0,62	0,24	0,49
5	0,66	0,47	**5**	0,66	0,43	0,23	0,48
6	0,64	0,47	**6**	0,65	0,49	0,23	0,48
7	0,82	**0,11**	**-**				
8	0,63	0,62	**7**	0,62	0,64	0,24	0,49
9	0,61	0,70	**8**	0,61	0,66	0,24	0,49
10	0,47	0,55	**9**	0,46	0,57	0,25	0,50
11	0,30	0,43	**10**	0,29	0,40	0,20	0,45
12	0,48	0,62	**11**	0,47	0,64	0,25	0,50
13	0,62	0,30	**12**	0,64	0,34	0,23	0,48
14	0,22	**-0,02**	**-**				
15	0,62	0,47	**13**	0,64	0,51	0,23	0,48
16	0,39	0,70	**14**	0,40	0,72	0,24	0,49
17	0,37	0,57	**15**	0,39	0,62	0,24	0,49
18	0,29	0,40	**16**	0,29	0,40	0,20	0,45
19	0,29	0,36	**17**	0,30	0,38	0,21	0,46
20	0,50	0,79	**18**	0,50	0,79	0,25	0,50

N=174, %27=47 öğrencidir.

Erişi testinin son haline ait betimsel istatistikler Tablo 3.5'te sunulmuştur.

Tablo 3.5: Erişi Testinin Son Haline Ait Betimsel İstatistikler

N		94
KR-20		0,80
Ortalama güçlük indeksi		0,45
Ortalama ayırıcılık indeksi		0,55
X		8,18
Medyan		8
Mod		4
Standart sapma		5,24
Varyans		27,44
Max		18
Min		0
Ranj		18
Çarpıklık katsayısı	Çarpıklık	0,14
	Çarpıklık st. hata	0,25
Basıklık katsayısı	Basıklık	-1,54
	Basıklık st. hata	0,50

174 öğrenciye uygulanan erişi testinden güçlük ve ayırıcılık indeksi uygun olmayan maddeler atıldıktan sonra nihai testin güvenirlik katsayısı hesaplanmış ve KR-20 değeri 0,80 olarak bulunmuştur. Bu değer testin güvenirliğinin yüksek olduğunu göstermektedir.

Testin ortalama güçlük indeksi 0,45 olarak hesaplanmıştır. Bu değer testin orta güçlükte olduğunu göstermektedir. Testin ortalama ayırıcılık indeksi ise 0,55 olarak hesaplanmıştır. İstenen ayırıcılık gücü olan 0,50'ye yakın olduğu için testin oldukça ayırt edici olduğu görülmektedir. Çarpıklık katsayısının standart hatasına bölünmesi ile elde edilecek z istatistiğinin α=0,05 için 1,96'dan küçük çıkması dağılımın normalden aşırı bir sapma göstermediği şeklinde yorumlanmaktadır (Büyüköztürk, 2013). Bu verinin 1,96'dan küçük olması (0,14/0,25=0,56) doğrultusunda normal dağılım gösteren bir erişi testinin geliştirildiği görülmektedir.

3.4.3. Mantıksal Düşünme Becerisi Testi (MDBT)

Mantıksal Düşünme Becerisi Testi (MDBT), Tobin ve Capie (1981) tarafından geliştirilmiş olup bu test değişkenleri kontrol etme, orantısal düşünme, olasılıklı düşünme, ilişkisel düşünme ve birleşik düşünme olmak üzere beş mantıksal işlemi ölçen 10 adet iki aşamalı sorudan oluşmaktadır. Testte yer alan sorular, öncelikli olarak bir dizi seçenek arasından bir cevabın seçilmesini ve sonrasında cevabın açıklamasının verilen seçenekler arasından seçilmesini gerektirmektedir. Bir sorunun doğru kabul edilebilmesi için, her iki aşamanın da işaretlenmiş olması ve her iki aşamaya da doğru cevap verilmiş olması gerekmektedir. Bu durumun, sorulara cevap vermede şans faktörünü en aza indirebileceği ve testin güvenirliğini yükseltebileceği düşünülebilir. Testteki maddelerin alt boyutlarına göre dağılımı Tablo 3.6'da sunulmuştur.

Tablo 3.6: Mantıksal Düşünme Becerisi Testinde Yer Alan Soruların Alt Boyutları

Alt Boyutlar	*Maddeler*
Değişkenlerin Kontrol Edilmesi	1. ve 2. maddeler
Orantısal Düşünme	3. ve 4. maddeler
İlişkisel Düşünme	5. ve 6. maddeler
Olasılıkla İlgili Düşünme	7. ve 8. maddeler
Birleşik Düşünme	9. ve 10. maddeler

Tobin ve Capie (1981) tarafından geliştirilen bu testin güvenirlik katsayısı 0,85 olarak rapor edilmiştir. Testin Türkçeye çevirisi ve uyarlanması Geban, Aşkar ve Özkan (1992) tarafından yapılmış ve Cronbach Alfa güvenirlik katsayısı 0,77 olarak bulunmuştur. Bu testten alınabilecek en yüksek puan 10'dur. Testten 0-3 aralığında puan alan öğrencinin düşük, 4-6 aralığında puan alan öğrencinin orta ve 7-10 aralığında puan alan öğrencinin yüksek düzeyde mantıksal düşünme becerisine sahip olduğu belirtilmektedir (Oliva, 2003).

3.4.4. Motivasyon Ölçeği

Araştırmada Pintrich, Smith, Garcia ve McKeachie (1991) tarafından Motivated Strategies for Learning Questionnaire (MSLQ) adıyla geliştirilen ve Büyüköztürk, Akgün, Özkahveci ve Demirel (2004) tarafından Türkçe'ye uyarlama çalışması yapılan Motivasyon Ölçeği, öğrenmeye ilişkin güdülenmeyi belirlemek amacıyla uygulanmıştır. Ölçeğin Türkçe uyarlamasında geçerlik ve güvenirlik çalışmaları ikinci, üçüncü ve son sınıfta okuyan 852 üniversite öğrencisinden elde edilen veriler kullanılarak gerçekleştirilmiştir. Yapılan uyarlama çalışması sonucunda ölçekteki madde sayıları ve bu maddelerin faktörlere göre dağılımları Tablo 3.7'de sunulmuştur.

Tablo 3.7: Ölçeğin Türkçe Uyarlaması Sonucu Oluşan Alt Faktörleri ve Cronbach Alfa Katsayıları

Faktör Adı	*Madde Sayısı*	*Maddeler*	*Cronbach Alfa*
İçsel hedef düzenleme faktörü	4	1, 16, 22, 24	0,59
Dışsal hedef düzenleme faktörü	4	7, 11, 13, 30	0,63
Görev değeri faktörü	6	4, 10, 17, 23, 26, 27	0,80
Öğrenmeye ilişkin kontrol inancı faktörü	4	2, 9, 18, 25	0,52
Öğrenme ve performansla ilgili öz yeterlik faktörü	8	5, 6, 12, 15, 20, 21, 29, 31	0,86
Sınav kaygısı faktörü	5	3, 8, 14, 19, 28	0,69

Türkçe uyarlamasında ölçeğin bütünü için Cronbach Alpha değeri 0,83 olarak hesaplanmıştır. Bu araştırma kapsamında ölçeğin bütünü için Cronbach Alfa güvenirlik katsayısı 0,86 olarak hesaplanmıştır. Ölçeğin uygulama süresi 20-30 dakika arasında değişmektedir. Ölçekteki maddelere katılma düzeyleri, 7'li Likert türü dereceleme ölçeğine göre "Benim için kesinlikle yanlış (1)" ile "Benim için kesinlikle doğru (7)" arasında değişmektedir.

3.5. Uygulama Süreci

3.5.1. Araştırmacının Rolü

A ve B şubelerinde 10 hafta boyunca tüm ders anlatımları ve 2 hafta süren veri toplama süreçleri araştırmacı tarafından gerçekleştirilmiştir. Bu kapsamda dersin planlaması, derslerin işlenmesi ve katılımcıların araştırma sürecinde takibi araştırmacı tarafından yapılmıştır.

3.5.2. Uygulama Süreci

Çalışmanın uygulama süreci 2015-2016 Eğitim-Öğretim yılı Bahar döneminde, Bilgisayar-II dersinde gerçekleştirilmiştir. Bilgisayar-II dersi, içeriğinde programlama öğretiminin de yer aldığı haftalık 4 saat olarak planlanan bir derstir. Bu doğrultuda dersin 3 saatinde programlama öğretimi gerçekleştirilmiş olup kalan bir saatinde ise müfredatta yer alan diğer konuların anlatımı yapılmıştır. Ders anlatımları her grup için haftada 3 ders saati olmak üzere 10 hafta boyunca devam etmiştir. Ön test ve son test verilerini toplamak için ise 2 haftalık bir süre ayrılmıştır. Uygulamada bir gruba (A şubesi) blok tabanlı görsel programlama eğitimi (Scratch), diğer gruba (B şubesi) ise metin tabanlı programlama öğretimi (Small Basic) yapılmıştır. Uygulama süreci tüm ayrıntıları ile Tablo 3.8'de sunulmuştur.

Tablo 3.8: Uygulama Süreci

UYGULAMA ÖNCESİ			
Hafta 1 ÖN TESTLER	***Veri Toplama Araçları***		***Süre***
	Kişisel bilgi formu		10 dakika
	Erişi testi		1 saat
	Mantıksal düşünme testi		1 saat
	Motivasyon ölçeği		25 dakika
SÜREÇ-1 (Süre: 3 Hafta)			
	Etkinlik (A ve B Şubeleri)		***Süre***
Hafta 2	Akış şemaları ve Metinsel Gösterimler		3 ders
Hafta 3	Akış şemaları ve Metinsel Gösterimler		3 ders
Hafta 4	Akış şemaları ve Metinsel Gösterimler		3 ders
Amaç: Algoritma mantığını yani değişken oluşturmayı, işlemleri ve akışı kavratma, programlamanın temel kavramlarını öğretme, gerçek bir problemi ortaya koyup o probleme ait işlem adımlarını sistematik bir şekilde oluşturabilmelerini sağlama.			
SÜREÇ-2 (Süre: 7 Hafta)			
	Etkinlik		***Süre***
	A Şubesi	***B Şubesi***	
Hafta 5	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 6	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 7	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 8	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 9	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 10	Blok Tabanlı Görsel Prog. (Scratch)	Metin Tabanlı Prog. (Small Basic)	3 ders
Hafta 11	Blok Tabanlı Görsel Prog. (Scratch))	Metin Tabanlı Prog. (Small Basic)	3 ders
Amaç: A şubesinde yer alan öğrencilerden sürükle bırak etkinlikleriyle verilen problemlerin çözümünün yapımı beklenirken, aynı işlemin B şubesinde kod yazmayı gerektiren Small Basic yazılımı ile gerçekleştirilmesi beklenmiştir.			
UYGULAMA SONRASI			
Hafta 12 SON TESTLER	***Veri Toplama Araçları***		***Süre***
	Erişi testi		1 saat
	Mantıksal düşünme testi		1 saat
	Motivasyon ölçeği		25 dakika

Eğitim sürecinde anlatılacak konuların ve derslerde çözülecek örneklerin belirlenmesinde ise BÖTE Bölümlerinde Programlama Dilleri dersleri veren

öğretim elemanlarının görüşleri alınmış ve programlama dili öğretimi ile ilgili ders kitapları incelenmiştir. Bu süreçte anlatımı yapılan konular ve çözülen örneklerle ilgili bilgiler Tablo 3.9'da sunulmuştur.

Tablo 3.9: Uygulama Sürecinde İşlenen Konular

<table>
<tr><td colspan="3">A ve B Şubeleri</td></tr>
<tr><td>Hafta 2
(3 Ders)
Hafta 3
(3 Ders)
Hafta 4
(3 Ders)</td><td colspan="2">Uygulamanın ilk 3 haftası boyunca hem A hem de B şubelerinde akış şemaları ve metinsel gösterimler kullanılarak bir programın algoritmasını oluşturabilme üzerine eğitim verilmiştir. Bu kapsamda; algoritma mantığını yani değişken oluşturmayı, işlemleri ve akışı kavratma, programlamanın temel kavramlarını öğretme, gerçek bir problemi ortaya koyup o probleme ait işlem adımlarını sistematik bir şekilde oluşturabilmelerini sağlama gibi konuların anlatımı gerçekleştirilmiştir.</td></tr>
<tr><td></td><td>A Şubesi</td><td>B Şubesi</td></tr>
<tr><td rowspan="3">Hafta 5
(3 Ders)</td><td>Blok Tabanlı Görsel Programlama (Scratch)</td><td>Metin Tabanlı Programlama (Small Basic)</td></tr>
<tr><td colspan="2">Kazanım: Algoritma oluşturma, ilgili programları kurma, açma, kapatma, yeni bir proje oluşturma, ara yüzü tanıma. İlk proje: Klavyeden bir değer girme, yazı rengi, arka plan rengi değişiklikleri.</td></tr>
<tr><td colspan="2">Uygulamalar: Verilen bir örnek uygulamanın arka planında çalışan algoritmaları tespit etme.</td></tr>
<tr><td rowspan="3">Hafta 6
(3 Ders)</td><td colspan="2">Kazanım: Değişken tanımlama. Matematiksel ifadelerle işlemler yapma (toplama, çıkarma, karekök, tam sayıya yuvarlama, mod).</td></tr>
<tr><td colspan="2">Uygulamalar:
Kütle numarası hesaplama,
Hesap Makinesi Örneği,
Alan hesaplama
ve
Sıcaklık Dönüştürücüsü (Matematiksel işlemler)
$°C = \frac{5(°F-32)}{9}$</td></tr>
<tr><td colspan="2">Ödev: 2 Vize, 1 Final ve 2 Sözlü notunu klavyeden girdikten sonra öğrencinin alacağı notu hesaplayan programı yazınız (Vize1 = %20, Vize2 = %20, Final = %40, Sözlü1 = %10, Sözlü2 = %10).</td></tr>
<tr><td>Hafta 7</td><td colspan="2">Kazanım: Koşul ifadeleri ve mantıksal operatörleri kullanarak bir program yazabilme.</td></tr>
</table>

(3 Ders)	**Uygulamalar:** Girilen bir sayının tek mi çift mi olduğunu bulma, Saat aralıklarına göre kullanıcıya dönüt veren program (Ör: Saat: 06.00-10.00 ise "*Günaydın*"), Su hangi sıcaklıklarda katı, sıvı ve gaz haline geçer? Verilen bir programdaki hatanın tespitini yapma ve düzeltme.
	Ödev: Yapılacak işlemi seçerek direnç, akım ve gerilim değerlerini hesaplayan program.
Hafta 8 (3 Ders)	**Kazanım**: Döngü ve sayaç kullanarak bir program yazabilme.
	Uygulamalar: 1 ile 10 arasındaki çift sayıların ortalamasını bulduarak sonucu ekranda gösterin. Rastgele girilen 10 sayının en büyük veen küçük olanını bulan programı yazma. N adet öğrencinin bulunduğu bir sınavın standart sapmasını hesaplayan programı yapınız (Standart Sapma formülü veriliyor).
	Ödev: Bilgisayara rastgele girilen 5 notun ortalamasını bulup bu ortalamaların harf notu karşılığını gösteren programı yazma.
Hafta 9 (3 Ders)	Sayı Tahmin Oyunu
Hafta 10 (3 Ders)	Uygulama (proje)
Hafta 11 (3 Ders)	Uygulama (proje)

3.6. Deneysel Çerçeve

Tablo 3.8'den de anlaşılacağı üzere kişisel bilgi formunun, erişi testinin, mantıksal düşünme testinin ve motivasyon ölçeğinin ön test olarak A ve B şubelerine uygulanması ile süreç başlamıştır. Her iki sınıfta da üç hafta boyunca algoritma mantığını yani değişken oluşturmayı, işlemleri ve akışı kavratma, programlamanın temel kavramlarını öğretme, gerçek bir problemi ortaya koyup o probleme ait işlem adımlarını sistematik bir şekilde oluşturabilmelerini sağlama gibi konuların anlatımı yapılarak öğrencilerin sürece ve derse alışmaları sağlanmaya çalışılmıştır.

Üç haftalık temel programlama eğitiminin ardından, A şubesindeki öğrenciler blok tabanlı görsel programlama ortamında, B şubesindeki öğrenciler ise metin tabanlı programlama ortamında eğitim almışlardır. Bu süreç 7 hafta sürmüştür. Bu süreç

içerisinde her iki şubede de dersler anlatıma ve problem çözme etkinliklerine dayalı olarak bilgisayar laboratuvarında gerçekleştirilmiştir. Anlaşılmayan noktalarda gerekli tekrarlar ve düzeltmeler yapılmış, dönütler verilmiştir. Öğrencilerin takıldıkları noktalara hızlı müdahale edebilmek ve bilgisayarda ders dışı işlerle ilgilenmelerin önüne geçebilmek için sınıf yönetimi yazılımlarından Net Support School kullanılmıştır. Bu yazılım ile öğrencilerin bilgisayarda kayıtlı olan diğer programlara erişimleri de sınırlandırılmıştır. Her hafta derslerin bitiminde öğrenciler ile soru cevap etkinliği yapılarak konunun anlaşılıp anlaşılmadığı tespit edilmeye çalışılmıştır.

Her iki sınıfta da dersler için aynı bilgisayar laboratuvarı kullanılmış olup, eğitim öncesinde bu laboratuvar yazılım ve donanım açısından eğitime uygun hale getirilmiştir. Şekil 3.2'de uygulama süreçleriyle ilgili görseller sunulmuştur.

Şekil 3.2: A (sol) ve B (sağ) şubelerine ait uygulama sürecinden bir görünüm

On haftalık eğitimin ardından erişi testinin, mantıksal düşünme testinin ve motivasyon ölçeğinin son test olarak her iki sınıfa uygulanması ile süreç sonlandırılmıştır.

3.7. Verilerin Analizi

Verilerin analizinde parametrik test varsayımlarının karşılanması durumunda parametrik istatistikler, karşılanmaması durumunda ise parametrik olmayan istatistikler kullanılmıştır. Bu doğrultuda uygulanan test ve ölçeklerden elde edilen verilerin çözümlenmesi için aritmetik ortalama, standart sapma ve t-Testi analizleri yapılmıştır. Ayrıca etki büyüklüğü (η^2) korelasyon katsayısı hesaplanmıştır. Analizler SPSS-20 paket programı ile gerçekleştirilmiş ve analizlerin anlamlılık düzeyi 0,05 olarak kabul edilmiştir. Hangi testte hangi analizin kullanıldığıyla ilgili detaylı bilgiler Tablo 3.10 ve Tablo 3.11'de sunulmuştur.

Tablo 3.10: A Şubesine İlişkin Verilerin Analizinde Kullanılan İstatistiksel Teknikler

Şube	*Testler*	*Yapılan Analizler*	*Erişi Testi*	*Mantıksal Düşünme Testi*	*Motivasyon Ölçeği*
Blok Tabanlı Görsel Programlama Öğretimi (A Şubesi)	**ÖN TEST**	Bağımsız Gruplar İçin t-Testi	✓	✓	
		Mann-Whitney U Testi			✓
	FARK (Son test - Ön test)	Eşleştirilmiş Örneklem t-Testi	✓	✓	✓

Blok tabanlı görsel programlama öğretiminin yapıldığı A şubesine ait ön test puanları üzerinde bağımsız gruplar t-testi ve Mann-Whitney U testi analizleri yapılmıştır. Ardından ön test puanlarına göre son testlerde anlamlı bir artışın gerçekleşip gerçekleşmediğine bakılmak üzere eşleştirilmiş örneklem t-testi (paired samples t-test) yapılmıştır.

Tablo 3.11: B Şubesine İlişkin Verilerin Analizinde Kullanılan İstatistiksel Teknikler

Şube	*Testler*	*Yapılan Analizler*	*Erişi Testi*	*Mantıksal Düşünme Testi*	*Motivasyon Ölçeği*
Metin Tabanlı Programlama Öğretimi (B Şubesi)	**ÖN TEST**	Bağımsız Gruplar İçin t-Testi	✓	✓	
		Mann-Whitney U Testi			✓
	FARK (Son test - Ön test)	Eşleştirilmiş Örneklem t-Testi	✓	✓	✓

Metin tabanlı programlama öğretiminin yapıldığı B şubesinde de ön test puanları üzerinde benzer analizler yapılmıştır. Ön test puanlarına göre son testlerde anlamlı bir artışın gerçekleşip gerçekleşmediğine bakılmak üzere ise yine eşleştirilmiş örneklem t-testi (paired samples t-test) yapılmıştır.

3.8. Araştırmanın İç ve Dış Geçerliği

Bir araştırmanın bilimsel ölçütlere uygun olarak gerçekleştirildiğini gösteren en önemli unsulardan birisi olan geçerlik, bir ölçme aracının ölçmeyi amaçladığı özelliği, bir başka özellikle karıştırmadan doğru olarak ölçebilme derecesi olarak tanımlanmaktadır (Alpar, 2014).

Geçerlik, iki başlık altında ele alınabilir. Bunlar iç ve dış geçerliktir. Bir araştırmada, bağımlı değişken üzerindeki değişimin, bağımsız değişkenlerdeki değişim ile açıklanabilme derecesi iç geçerlik, bu sonucun araştırma için seçilen grupların özelliklerine sahip daha büyük gruplara genellenebilme derecesi ise dış geçerlik olarak tanımlanmaktadır (Köklü, 2002; Fraenkel ve Wallen, 2003; Büyüköztürk, Çakmak, Akgün, Karadeniz ve Demirel, 2015). Bu araştırma kapsamında iç ve dış geçerliği sağlayabilmek için alınan önlemler Tablo 3.12'de sunulmuştur.

Tablo 3.12. İç ve Dış Geçerlik Tehditlerine Yönelik Olarak Alınan Önlemler

	Olası tehditler	*Önlemler*
İç geçerlik	Araştırmaya katılanların seçimi	Her iki şubede bulunan öğrencilere çalışma öncesinde kişisel bilgi formu uygulanarak çeşitli değişkenler açısından (kişisel özellikler, liseden mezun olduğu okul türü, cinsiyet) benzerlikler ortaya konulmaya çalışılmıştır.
	Katılımcıların ayrılması (katılımcı kaybı)	Katılımcı kaybının araştırmanın sonuçları üzerindeki etkisini en aza indirebilmek için her iki sınıfta da fazladan öğrenciler yer almıştır. Çalışma sonunda testleri eksik dolduran, çalışmayı tamamlayamayan ve eğitime tam katılamayan öğrenciler belirlenerek araştırmanın kapsamı dışında tutulmuştur.
	Beklenti etkisi	Tüm öğrencilere uygulanacak testler, araştırmanın koşulları ve ayrıntıları hakkında bilgi verilmemiştir.
	Ön test etkisi	Ön test ile son test sorularının aynı olmasından dolayı, öğrencilerin soruları hatırlamasını zorlaştırmak için ikinci ölçümler 10 haftalık bir zaman dilimi sonrasında gerçekleştirilmiştir.
	Veri toplama aracı	Veri toplama araçlarından kaynaklı tehditlerin önüne geçebilmek için her iki gruba da aynı ölçme aracı aynı zaman diliminde ve aynı kişi tarafından uygulanmıştır. Ayrıca her iki gruba da eşit süreler tanınmıştır.
	Veri toplayıcı farklılığı	Veri toplayıcının farklılaşmasından kaynaklı tehditleri engelleyebilmek için çalışma kapsamında tüm süreç araştırmacı tarafından yönetilmiştir. Her iki sınıftta da 10 hafta ders anlatımları, tüm verilerin toplanması araştırmacı tarafından gerçekleştirilmiştir.
	Verinin toplandığı mekânın farklaşması	Mekâna bağlı olarak ortaya çıkabilecek iç geçerlik tehditlerini engelleyebilmek için her iki sınıfta da veriler aynı ortamda toplanmıştır. Her iki sınıfta da tüm veriler derslerin işlendiği bilgisayar laboratuvarında toplanmıştır.
Dış geçerlik	Örnekleme (temsil edebilme) etkisi	Bu araştırmada, çalışma grubunun yapısı nedeniyle sınırlı bir genelleme söz konusudur. Çalışma bir devlet üniversitesinin Eğitim Fakültesi Fen Bilgisi Öğretmenliği Anabilim Dalı 2. sınıfında öğrenim görmekte olan öğrencilerle gerçekleştirilmiştir. Elde edilen sonuçlar sadece benzer özelliklere sahip gruplar için genellenebilir.

4. BULGULAR VE YORUMLAR

Bu bölümde blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin yapıldığı öğrencilerin erişi, mantıksal düşünme ve programlamaya yönelik motivasyon durumları sırasıyla incelenmiştir. Verilerin istatistiksel çözümlemesi sonucunda elde edilen bulgular tablolar halinde sunulmuştur.

4.1. Araştırmanın Alt Problemleri

Blok tabanlı görsel programlama eğitimi alan öğrencilerin;

1. Erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
2. Mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
3. Programlaya yönelik motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?

Metin tabanlı programlama eğitimi alan öğrencilerin;

4. Erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
5. Mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?
6. Programlaya yönelik motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?

4.2. Blok Tabanlı Görsel Programlama Öğretimine İlişkin Bulgular

Bu bölümde A şubesine uygulanan blok tabanlı görsel programlama eğitimi sonucu öğrencilerin ön testlerdeki puanlarının son testlerde anlamlı bir artış gösterip göstermediği araştırılmıştır.

4.2.1. Erişi Puanlarına İlişkin Bulgular

Birinci alt problemde "*Blok tabanlı görsel programlama eğitimi alan öğrencilerin erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için erişi ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmış ve etki büyüklüğü hesaplanmıştır. Ulaşılan sonuçlar Tablo 4.1'de sunulmuştur.

Tablo 4.1: Blok Tabanlı Görsel Programlama Eğitimi Alan Öğrencilerin Erişi Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Erişi Testi	Ön test	30	4,47	1,96	29	12,388	,000	2,26
	Son test	30	9,97	2,51				

Tablo 4.1'de görüldüğü gibi öğrencilerin erişi testi son test puan ortalaması ($\bar{x}$=9,97), ön test puan ortalamasına ($\bar{x}$=4,47) göre daha yüksek olup bu fark istatistiksel olarak anlamlıdır (t_{29}=12,388, p<0,05). Test sonucu hesaplanan etki büyüklüğü (η^2=2,26) bu farkın oldukça yüksek düzeyde olduğunu göstermektedir. Çünkü etki büyüklüğü 1'in üzeri için çok büyük olarak yorumlanırken, 0,8 büyük, 0,5 orta, 0,2 ise küçük (az) etki olarak değerlendirilir (Gren ve Salkind, 2005). Bu durum, blok tabanlı görsel programlama öğretiminin erişi puanları üzerinde anlamlı bir etkisinin olduğunu göstermektedir. Ayrıca bu bulgu, blok tabanlı görsel bir programlama ortamının kullanımı ile erişi arasında anlamlı bir ilişkinin olduğu şeklinde de yorumlanabilir.

Programlama öğretiminde Scratch yazılımının kullanımı ile ilgili erişi değişkenini ele alan başka araştırmalar da bulunmaktadır. Örneğin Fesakis ve Serafeim (2009), görsel programlama eğitiminin bilgisayar bilimine giriş ve programlama uygulamalarında pozitif bir etkiye sahip olduğunu; Kaucic ve Asic (2011), eğitsel programlama dili gibi basite indirgenmiş bir ortamın programlama öğretiminde kullanımının programlama öğrenimini olumlu yönde etkilediğini belirtmiştir. Ayrıca programlama ders konularının alışılagelmiş yöntemler yerine blok tabanlı görsel programlama dilleriyle verilmesinin öğrencilerin erişilerini olumlu yönde etkilediğini belirten çalışmalar da bulunmaktadır (Örneğin Klassen, 2006; Arabacıoğlu, Bülbül ve Filiz, 2007; Malan ve Leither, 2007; Wang ve Zhou, 2011; Ozoran, Çağıltay ve Topallı, 2012; Giordano ve Mairona, 2014; Demir, 2015; Erol, 2015).

Bu sonuçla benzerlik göstermeyen çalışmalar incelendiğinde blok tabanlı görsel programlama eğitiminin erişi yönünden farklılık yaratmadığı ancak öğrencilerin bu

ortamları kullanarak program yazmayı sevdikleri üzerinde durulmaktadır. Örneğin Kereki (2008) tarafından yapılan bir çalışmada Bilgisayar Bilimleri dersi kapsamında blok tabanlı görsel programlama eğitimi yapılarak öğrencilerin erişi ve motivasyon düzeyleri ölçülmüştür. Çalışmanın sonuçlarına bakıldığında ön test ve son test puanları bakımından gruplar arasında anlamlı bir farklılık çıkmazken, Scratch ile eğitim alan deney grubunun motivasyon düzeyi kontrol grubuna göre daha yüksek çıkmıştır. Benzer sonuçlara Giordano ve Mairona (2015) ile Koorsse, Ciliers ve Calitz'in (2015) araştırmalarında da ulaşılmıştır. Tekerek ve Altan (2014) ise kontrol gruplu ön test son test deseninin kullanıldığı çalışmada deney ve kontrol gruplarının son test puanları arasında anlamlı bir fark oluşmadığını ve görsel programlama eğitiminin öğrenci erişisi üzerine olumlu bir etkisinin olmadığını belirtmiştir.

4.2.2. Mantıksal Düşünme Testi Puanlarına İlişkin Bulgular

İkinci alt problemde "*Blok tabanlı görsel programlama eğitimi alan öğrencilerin mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için mantıksal düşünme ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmıştır. Ulaşılan sonuçlar Tablo 4.2'de sunulmuştur.

Tablo 4.2: Blok Tabanlı Görsel Programlama Eğitimi Alan Öğrencilerin Mantıksal Düşünme Testi Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Mantıksal Düşünme	Ön test	30	4,00	2,39	29	7,628	,000	1,39
	Son test	30	6,07	1,68				

Tablo 4.2'de son test puan ortalamasının ($\bar{x}$=6,07), ön test puan ortalamasına ($\bar{x}$=4,00) göre daha yüksek olduğu görülmektedir. Ortalamalar arasındaki farka bakılarak blok tabanlı görsel programlama eğitiminin mantıksal düşünme düzeyini artırdığı belirtilebilir. Çünkü mantıksal düşünme testinden 0-3 arası puan alan öğrencinin düşük, 4-6 arası orta ve 7-10 arası puan alan öğrencinin ise yüksek düzeyde mantıksal düşünme yetisine sahip olduğu belirtilmektedir (Oliva, 2003). Son testte ortalamanın yüksek düzeye yaklaştığı görülmektedir. Bu farklılık

istatistiksel olarak da anlamlıdır (t_{29}=7,628, p<0,05). Hesaplanan etki büyüklüğüne (η^2=1,39) bakıldığında bu farkın oldukça yüksek düzeyde olduğu görülmektedir.

Ulaşılan sonuçlar, blok tabanlı görsel programlama eğitimi ile mantıksal düşünme becerisi arasında anlamlı bir ilişkinin olduğunu ortaya koymaktadır. Yenilmez, Sungur ve Tekkaya (2005) bu ilişkinin mantıksal düşünmenin, programlamanın temelinde yer alan problem çözme becerisinin önemli bir yordayıcısı olmasından kaynaklandığını belirtmiştir. Robbins (2011), mantıksal düşünmenin problem çözmede gerekli bir bileşen olduğunu savunmaktadır. Sebetci ve Aksu (2014) ise öğrencilerin mantıksal düşünme becerileri ile programlama erişileri arasında pozitif yönde, orta düzeyde ve anlamlı bir ilişki olduğunu, mantıksal düşünme becerileri geliştirildiğinde programlama erişilerinin de artacağını belirtmektedir.

Öğrencilerin mantıksal düşünme puanlarının son testlerde artış göstermesinin diğer bir sebebinin Scratch'ın ilgi çekici yapısından kaynaklı olduğu düşünülmektedir. Çünkü bir programlama ortamının ilgi çekici yapıda olması öğrencilerin bu ortamda daha fazla zaman geçirmesine, dolayısıyla da programlama deneyimlerinin gelişmesine yol açtığı görüşü savunulabilir. Sebetci ve Aksu'nun (2014) belirttiği gibi öğrencilerin programlama deneyimlerinin artmasının bir sonucu olarak mantıksal düşünme becerilerinin de arttığı düşünülmektedir.

4.2.3. Motivasyon Puanlarına İlişkin Bulgular

Üçüncü alt problemde "*Blok tabanlı görsel programlama eğitimi alan öğrencilerin motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için motivasyon ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmıştır. Ulaşılan sonuçlar Tablo 4.3'te sunulmuştur.

Tablo 4.3: Blok Tabanlı Görsel Programlama Eğitimi Alan Öğrencilerin Motivasyon Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Motivasyon	Ön test	30	136,97	24,07	29	4,642	,000	0,85
	Son test	30	152,17	16,48				

Tablo 4.3 incelendiğinde, öğrencilerin son test puan ortalaması ($\bar{x}$=152,17), ön test puan ortalamasına ($\bar{x}$=136,97) göre daha yüksek olup bu fark istatistiksel olarak anlamlıdır (t_{29}=4,642, p<0,05). Hesaplanan etki büyüklüğüne (η^2=0,85) bakıldığında bu farkın yüksek etki büyüklüğünde olduğu görülmektedir. Motivasyon son testinde görülen bu artışın temel sebebinin blok tabanlı görsel programlama eğitimi olduğu düşünülmektedir. Çünkü yazılımlarda grafik ve animasyon kullanımı öğrencilerin derslere olan ilgisini arttırmada oldukça etkili bir yöntemdir (Brusilovsky ve Spring, 2004; Kelleher, Pausch ve Kiesler, 2007).

Son testte görülen artışın bir diğer sebebinin ise Scratch yazılımının keyifli bir programlama ortamına sahip olması gösterilebilir. Nitekim alanyazında merak uyandıran, eğlenceli ve hedefler ile iyi ilişkilendirilmiş öğrenme ortamlarında içsel motivasyonun artabileceği belirtilmektedir (Deci, Kostner ve Ryan, 2001). Genç ve Karakuş (2011) tarafından "Eğitimde Bilgisayar Oyunları Tasarımı" dersi kapsamında öğrencilerin deneyimlerini ve görüşlerini belirlemek amacıyla Scratch etkinlikleriyle desteklenen bir çalışma gerçekleştirilmiştir. Yapılan çalışma sonucunda, öğrencilerin %79'u Scratch kullanımını basit ve kolay bulduklarını ve kendilerini rahat hissettiklerini, Scratch'ın eğlenceli ve keyifli bir ortam olduğunu ve genel olarak Scratch kullanımından hoşlandıklarını belirtmişlerdir. Öğrencilerin yaklaşık %20'si ise Scratch ile programlama deneyiminden hoşlanmadıklarını rapor etmişlerdir. Ayrıca öğrencilerin %73'ü Scratch ile programlama yapılarının öğrenilmesinin ve anlaşılmasının diğer programlama dillerine göre daha kolay olduğunu ve Scratch'ın eğitsel bir araç olarak kullanılabileceğini belirtmişlerdir.

Alanyazına bakıldığında da blok tabanlı görsel ortamların, programlamaya yönelik motivasyonu arttırdığıyla ilgili sonuçlar görülmektedir. Örneğin Erol (2015) tarafından gerçekleştirilen bir çalışmada, Bilişim Teknolojileri öğretmen adaylarına yönelik olarak yapılan programlama öğretiminde Scratch kullanımının motivasyon ve erişiye olan etkisi araştırılmıştır. Araştırma sonucu elde edilen bulgulara göre

katılımcıların motivasyon puanlarının ön testte her iki grupta da benzer olduğu, son testte ise deney grubu lehine anlamlı bir farklılık oluştuğu belirtilmiştir.

4.3. Metin Tabanlı Programlama Öğretimine İlişkin Bulgular

Bu bölümde B şubesine uygulanan metin tabanlı programlama eğitimi sonucu öğrencilerin ön testlerdeki puanlarının son testlerde anlamlı bir artış gösterip göstermediği araştırılmıştır.

4.3.1. Erişi Puanlarına İlişkin Bulgular

Birinci alt problemde "*Metin tabanlı programlama eğitimi alan öğrencilerin erişi ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için erişi ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmıştır. Ulaşılan sonuçlar Tablo 4.4'te sunulmuştur.

Tablo 4.4: Metin Tabanlı Programlama Eğitimi Alan Öğrencilerin Erişi Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Erişi Testi	Ön test	30	4,27	1,82	29	12,155	,000	2,22
	Son test	30	10,67	2,99				

Tablo 4.4'te görüldüğü gibi son test puan ortalaması ($\bar{x}$=10,67), ön test puan ortalamasına ($\bar{x}$=4,27) göre daha yüksek olup bu fark istatistiksel olarak anlamlıdır (t_{29}=12,155, p<0,05). Test sonucu hesaplanan etki büyüklüğü (η^2=2,22) bu farkın oldukça yüksek etki büyüklüğünde olduğunu göstermektedir. Small Basic ile yapılan metin tabanlı programlama eğitimi öğrencilere programlama konusunda yeni bilgiler kazandırmıştır. Alanyazına bakıldığında da metin tabanlı programlama ortamlarında erişi puanlarının son testlerde arttırdığıyla ilgili sonuçlar görülmektedir (Ruf, Mühling ve Hubwieser, 2014; Erol, 2015).

4.3.2. Mantıksal Düşünme Testi Puanlarına İlişkin Bulgular

İkinci alt problemde "*Metin tabanlı programlama eğitimi alan öğrencilerin mantıksal düşünme ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için mantıksal düşünme ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmıştır. Ulaşılan sonuçlar Tablo 4.5'te sunulmuştur.

Tablo 4.5: Metin Tabanlı Programlama Eğitimi Alan Öğrencilerin Mantıksal Düşünme Testi Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Mantıksal Düşünme	Ön test	30	3,87	2,39	29	3,477	,002	0,63
	Son test	30	4,93	2,45				

Tablo 4.5'te görüldüğü gibi son test puan ortalaması ($\bar{x}$=4,93), ön test puan ortalamasına ($\bar{x}$=3,87) göre daha yüksek olup bu fark istatistiksel olarak anlamlıdır (t_{29}=3,477, $p<0,05$). Metin tabanlı programlama öğretiminin yapıldığı grubun mantıksal düşünme puan ortalaması ön testte orta düzeyin altında iken yapılan eğitimle birlikte orta düzeye yükselmiştir. Nitekim test sonucu hesaplanan etki büyüklüğü de (η^2=0,63) bu farkın orta düzeyde bir etki büyüklüğüne sahip olduğunu göstermektedir.

Alışılagelmiş bir yöntem olan metin tabanlı programlama öğretimi, mantıksal düşünme üzerinde orta düzeyde anlamlı bir farklılık oluşturmuştur. Bu farklılığın bir sebebinin, mantıksal düşünmenin problem çözme sürecinin önemli bir bileşeni olmasından, dolayısıyla da programlama öğrenimi ile gelişebilecek bir beceri olmasından kaynaklı olduğu düşünülmektedir. Alanyazında da programlama eğitiminin problem çözme becerisinin gelişimine yani mantıksal düşünmeye katkılarıyla ilgili bulgular yer almaktadır (Dalton ve Goodrum, 1991; Lai ve Yang, 2011; Fessakis, Gouli ve Mavroudi, 2013; Hooshyar, Ahmad, Shamshirband, Yousefi ve Horng, 2015). Liao ve Bright (1991) tarafından gerçekleştirilen bir meta analiz çalışmasında da programlamanın problem çözme ve mantıksal düşünme becerilerini geliştirdiğine yönelik çalışmaların ağırlıkta olduğu belirtilmiştir.

4.3.3. Motivasyon Puanlarına İlişkin Bulgular

Üçüncü alt problemde "*Metin tabanlı programlama eğitimi alan öğrencilerin motivasyon ön test puanları ile son test puanları arasında istatistiksel olarak anlamlı bir farklılık var mıdır?*" sorusuna cevap aranmıştır.

Bu soruya cevap bulmak için motivasyon ön test ve son test puanları üzerinde eşleştirilmiş örneklem t-testi (paired samples t-test) analizi yapılmıştır. Ulaşılan sonuçlar Tablo 4.6'da sunulmuştur.

Tablo 4.6: Metin Tabanlı Programlama Eğitimi Alan Öğrencilerin Motivasyon Puanlarına (Son Test - Ön Test) İlişkin Eşleştirilmiş Örneklem t-Testi Sonuçları

	Testler	*N*	$\bar{x}$	*ss*	*sd*	*t*	*p*	η^2
Motivasyon	Ön test	30	150.43	34.42	29	-0,951	,349	0,17
	Son test	30	144.80	20.72				

Tablo 4.6 incelendiğinde, öğrencilerin son test puan ortalaması ($\bar{x}$=144,80), ön test puan ortalamasına ($\bar{x}$=150,43) göre daha düşük olup bu farkın istatistiksel olarak anlamlı olmadığı görülmektedir (t_{29}=-0,951, p>0,05). Test sonucu hesaplanan etki büyüklüğü (η^2=0,17) bu farkın düşük düzeyde bir etki büyüklüğüne sahip olduğunu göstermektedir. Metin tabanlı programlamanın kod yazımı gerektiren karmaşık yapısı, problemin çözümünde yapılan hatalar (syntax, hatalı kod yazımı, hatalı noktalama işaretleri vs.) ve bünyesinde yoğun bir bilişsel süreci barındırması gibi durumların öğrencilerde zamanla içsel motivasyonun düşmesine yol açtığı belirtilebilir. Bu durum, bu çalışma sonucunda testler arasında oluşan motivasyon farklılığının bir sebebi olarak düşünülmektedir. Çünkü motivasyon, programlama öğretimi amaçlı kullanılan ortamdan etkilenmektedir (Calder, 2010).

5. SONUÇLAR VE ÖNERİLER

Bu bölüm, blok tabanlı görsel ve metin tabanlı programlama öğretimi sonucu elde edilen sonuçlar üzerine odaklanmaktadır. Ulaşılan sonuçlar doğrultusunda birtakım önerilerde bulunulmuştur.

5.1. Sonuçlar

Blok tabanlı görsel programlama eğitimi alan öğrencilerin;

Programlama erişi düzeylerinde ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış yaşanmıştır. Hesaplanan etki büyüklüğünün ise oldukça yüksek olduğu sonucuna ulaşılmıştır.

Mantıksal düşünme düzeylerinde ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış yaşanmış olup etki büyüklüğünün oldukça yüksek düzeyde olduğu görülmüştür.

Programlamaya yönelik motivasyon düzeyleri ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış göstermiştir. Hesaplanan etki büyüklüğünün ise yine oldukça yüksek düzeyde olduğu sonucuna ulaşılmıştır.

Metin tabanlı programlama eğitimi alan öğrencilerin;

Programlama erişi düzeylerinde ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış yaşanmıştır. Hesaplanan etki büyüklüğünün ise oldukça yüksek olduğu sonucuna ulaşılmıştır.

Mantıksal düşünme düzeylerinde ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış yaşanmış olup etki büyüklüğünün ise orta düzeyde olduğu görülmüştür.

Programlamaya yönelik motivasyon düzeylerinde ön test puan ortalamalarına kıyasla son testte istatistiksel olarak anlamlı bir artış görülmemiştir. Hesaplanan etki büyüklüğü ise düşük düzeydedir.

5.2. Öneriler

Yapılan araştırma neticesinde elde edilen sonuçlara dayanarak uygulamaya ve ileride yapılacak araştırmalara ilişkin önerilerde bulunulmuştur.

5.2.1. Uygulamalara İlişkin Öneriler

Öğrencilerin takıldıkları noktalara hızlı müdahale edebilmek ve bilgisayarda ders dışı işlerle ilgilenmelerinin önüne geçebilmek için uygulamalar esnasında sınıf yönetimi yazılımlarının (NetSupport School, Netup School vs.) kullanımı yerinde olacaktır.

Etkili bir uygulama için bilgisayarların donanım ve yazılım açısından programları kaldırabilecek düzeyde özelliklere sahip olması gerekmektedir. Gerekli kontroller sağlanmadan uygulamaya geçilmemelidir.

Daha önce algoritma veya programlama gibi bir eğitim almamış bir grupla çalışılacaksa, programlama öğretimine en temel bilgiler verilerek başlanılmasının gerektiği göz önünde bulundurulmalıdır.

Etkili bir uygulama süreci için öğrencilerin öğrenim gördükleri alanla ilişkili olan örnekler oluşturulmaya çalışılmalıdır. Örneğin, programlama eğitimiyle ilgili bir çalışma fizik bölümü öğrencileriyle yürütülecekse yazılacak programların olabildiğince fizikle ilgili konulardan seçilmesi öğrencilerin farkındalığını arttıracaktır.

5.2.2. Araştırmalara İlişkin Öneriler

Çalışma kapsamında geliştirilen erişi testi yalnızca metin tabanlı programlamaya yönelik sorulardan oluştuğu için gruplar arasında bir karşılaştırma yapılamamıştır. Yeni yapılacak çalışmalarda hem blok tabanlı hem de metin tabanlı sorulardan oluşan bir erişi testi geliştirilip gruplar arasında karşılaştırmalar yapılabilir.

Bu çalışma kapsamında mantıksal düşünme testi ve motivasyon ölçeğinin tamamı üzerinden bir değerlendirmede bulunulmuştur. Yeni yapılacak çalışmalarda alt faktörlerin de dâhil edildiği daha kapsamlı bir araştırma gerçekleştirilebilir.

Bu çalışmada öğrencilerin öğrenme stillerine göre bir gruplandırması yapılamamıştır. Yeni yapılacak çalışmalarda öğrenciler öğrenme stillerine göre

sınıflandırılarak blok tabanlı görsel ve metin tabanlı programlama öğretimlerinin erişi, mantıksal düşünme ve motivasyon üzerindeki etkililiği daha net bir şekilde ortaya konulabilir.

Çalışma sonucunda programlama erişi ve mantıksal düşünme puanlarında her iki sınıfta da anlamlı bir artış görülmüştür. Bu durumun bir sebebi olarak çalışmanın Matematik ağırlıklı bir bölümde gerçekleştirilmiş olması gösterilebilir. Bu durumun sebebinin daha ayrıntılı şekilde ortaya konulabilmesi için yapılacak yeni çalışmalarda öğrencilerin matematik puanları çalışma öncesinde tespit edilerek programlamayla olan ilişkisine bakılabilir. Ayrıca benzer bir çalışma farklı bölümlerde öğrenim gören öğrencilerle gerçekleştirilip sonuçları arasında bir karşılaştırma yapılabilir.

Bu çalışmada blok tabanlı görsel programlama öğretiminin motivasyon üzerinde anlamlı bir etkisi olduğu sonucuna ulaşılmıştır. Metin tabanlı programlama öğretiminin yapıldığı sınıfta ise uygulama sonrası motivasyon puanlarında düşüş yaşanmıştır. Her iki sınıfta da aynı eğitimlerin yapılmış olmasına rağmen uygulamalar sonucu oluşan motivasyon farklılıklarının programlama ortamından kaynaklı olduğu düşünülmektedir. Bu doğrultuda çalışmaya farklı blok tabanlı görsel programlama ortamları da dâhil edilerek sonuçlar açısından bir karşılaştırma yapılabilir.

Bu çalışmada blok tabanlı görsel programlama ortamlarından Scratch kullanılmış olup bu ortamda program yazan öğrencilerin mantıksal düşünme düzeylerinin metin tabanlı ortama göre daha fazla geliştiği görülmüştür. Bu kapsamda yapılacak benzer bir çalışmada progralama öğretiminin diğer bilişsel beceriler (problem çözme, eleştirel düşünme, yaratıcılık vb.) üzerindeki etkileri araştırılabilir.

Scratch yazılımıyla ilgili üniversite düzeyinde yapılan çalışmalarda ağırlıklı olarak programlama derslerinde kullanımı üzerine yoğunlaşılmıştır. Scratch ile programlama eğitiminin erişi, mantıksal düşünme ve motivasyon üzerindeki etkisini araştıran yeni çalışmalar yapılarak ulaşılan sonuçların bir karşılaştırması yapılabilir.

KAYNAKÇA

Akbaba, S. (2006). Eğitimde motivasyon. *Kazım Karabekir Eğitim Fakültesi Dergisi, 13*(1), 343-361.

Akçay, T. (2009). *Perceptions of students and teachers about the use of a kid's programming language in computer courses*. Yayımlanmamış Yüksek Lisans Tezi, ODTÜ, Ankara.

Akpınar, Y., & Aslan, Ü. (2015). Supporting childrens learning of probability through video game programming. *Journal of Educational Computing Research, 53*(2), 228-259.

Akpınar, Y. ve Altun, A. (2014). Bilgi toplumu okullarında programlama eğitimi gereksinimi. *İlköğretim Online, 13*(1), 1-4.

Aldağ, H. ve Tekdal, M. (2015).*Bilgisayar kullanımı ve programlama öğretiminde cinsiyet farklılıkları.* 1.Uluslararası Çukurova Kadın Çalışmaları Kongresi. Adana, Türkiye, 9-11 Nisan 2015. [Çevrim-içi: https://goo.gl/2LnIHN, Erişim Tarihi: 20.10.2016.]

Alice (2016). *Alice about*.[Çevrim-içi: http://www.alice.org/index.php, Erişim Tarihi: 02.08.2016.]

Alice Software. (1998). Wikipedia, The Free Encyclopedia. [Çevrim-içi: https://en.wikipedia.org/wiki/Alice_(software), Erişim Tarihi: 02.09.2016.]

Alpar, R. (2014). *Spor, sağlık ve eğitim bilimlerinden örneklerle uygulamalı istatistik ve geçerlik - güvenirlik*. Ankara: Detay Yayıncılık.

Altun, M. (2004). *İlköğretim ikinci kademede (6, 7 ve 8. sınıflarda) matematik öğretimi*. Bursa: Alfa Yayınları.

App Inventor (2016). *What is App Inventor?* [Çevrim-içi: https://goo.gl/6fd8RY, Erişim Tarihi: 03.09.2016.]

Arabacıoğlu, C., Bülbül, H. ve Filiz, A. (2007). *Bilgisayar programlama öğretiminde yeni bir yaklaşım*. Akademik Bilişim 2007 Konferansı. Dumlupınar Üniversitesi, Kütahya, 31 Ocak 2 Şubat 2007. [Çevrim-içi: http://ab.org.tr/ab07/kitap/arabacioglu_bulbul_AB07.pdf, Erişim tarihi: 11 Ağustos 2016.]

Aşkar, P. (1989). *Etkileşimli problem çözme*. Problem Çözme Yöntemleri Sempozyumu. ODTÜ, Ankara, 29-30 Eylül 1989. [Çevrim-içi: https://books.google.com.tr/books/about/Problem_%C3%A7%C3%B6zme_y%C3%B6ntemleri_sempozyumu_2.html?id=1EvGtgAACAAJ&redir_esc=y, Erişim Tarihi: 12.09.2016.]

Aykaç, N. (2014). *Öğretim ilke ve yöntemleri*. Ankara: Pegem Akademi.

Balım, M. A. (2013). *Çocuklara neden programlama öğretmeliyiz?* [Çevrim-içi: http://www.elektrikport.com/teknik-kutuphane/cocuklara-neden-programlama-ogretmeliyiz/8832#ad-image-0, Erişim Tarihi: 10.08.2016.]

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

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

Bayman, P., & Mayer, R. E. (1983). A diagnosis of beginning programmers' misconceptions of Basic programming statements. *Communications of the ACM,26*(9)*,* 677-679.

Bayraktar, H. V. (2015). Sınıf yönetiminde öğrenci motivasyonu ve motivasyonu etkileyen etmenler. *International Periodical For The Languages, Literature and History of Turkish or Turkic, 10*(3), 1079-1100.

Ben-Ari, M. (2013). *Visualization of programming*. New York: Routledge.

Bergin, J., Brodie, K., Patino-Martínez, M., McNally, M., Naps, T., Rodger, S., & Jimenez-Peris, R. (1996). An overview of visualization: Its use and design: report of the working group in visualization. *In ACM SIGCSE Bulletin, 6*(96), 192-200.

Bilen, M. (2002). *Plandan uygulamaya öğretim*. Ankara: Anı Yayıncılık.

Bozdoğan, A. (2007). *Fen bilgisi öğretiminde çalışma yaprakları ile öğretimin öğrencilerin fen bilgisi tutumuna ve mantıksal düşünme becerilerine etkisi*. Yayınlanmamış Yüksek Lisans Tezi. Çukurova Üniversitesi, Sosyal Bilimler Enstitüsü, Adana.

Brown, Q., Mongan, W., Kusic, D., Garbarine, E., Fromm, E., & Fontecchio, A. (2013). *Computer aided instruction as a vehicle for problem solving: Scratch programming environment in the middle years classroom*. [Çevrim-içi: http://www.pages.drexel.edu/~dmk25/ASEE_08.pdf, Erişim Tarihi: 05.09.2016.]

Brusilovsky, P., & Spring, M. (2004*). Adaptive, engaging, and explanatory visualization in a C programming course*. Proceedings of EDMEDIA' 2004 - World Conference on Educational Multimedia, Hypermedia and Telecommunications. Lugano, Switzerland. [Çevrim-içi: http://www.pitt.edu/~peterb/papers/EDMED04ExVis_Final.pdf, Erişim Tarihi: 18.10.2016.]

BTE Derneği (2013). *Türkiye'de ilk ve ortaokullarda (ilköğretim) okutulan bilişim teknolojileri derslerinin tarihi*. [Çevrim-içi:http://bte.org.tr/bte-dernegi/bt-derslerinin-tarihcesi/, Erişim Tarihi: 01.08.2016.]

Burke, Q., & Kafai, Y. B. (2010). *Programming & storytelling: Opportunities for learning about coding & composition*. 9th International Conference on Interaction Design and Children. Barcelona, Spain, June 9-12, 2010. [Çevrim-içi: 10.1145/1810543.1810611, Erişim Tarihi: 16.09.2016.]

Büyüköztürk, Ş., Akgün, Ö. E., Özkahveci, Ö., & Demirel, F. (2004). The validity and reliability study of the Turkish version of the motivated strategies for learning questionnaire. *Educational Sciences: Theory & Practice*, *4*(2), 207-239.

Büyüköztürk, Ş., Çakmak, E. K., Akgün, Ö. E., Karadeniz, Ş. ve Demirel, F. (2015). *Bilimsel araştırma yöntemleri.* Ankara: Pegem Akademi.

Byrne, P., & Lyons, G. (2001). *The effect of student attributes on success in programming*. Proceedings of The 6th Annual Conference On Innovation And Technology In Computer Science Education. Canterbury, UK, June 25-27, 2001. [Çevrim-içi: http://dl.acm.org/citation.cfm?id=377467, Erişim Tarihi: 19.09.2016.]

Calao, L. A., Moreno-Leon, J., Correa, H. E., & Robles, G. (2015). *Developing mathematical thinking with scratch an experiment with 6th grade students*. 10th European Conference on Technology Enhanced Learning. Toledo, Spain, September 15-18, 2015. [Çevrim-içi: http://link.springer.com/chapter/10.1007%2F978-3-319-24258-3_2, Erişim Tarihi: 20.07.2016.]

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

Choi, S., Bell, T., Jun, S. J., & Lee, W. G. (2008). *Designing offline computer science activities for the korean elementary school curriculum.*13th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE 08), Madrid, Spain, June 30 - July 02, 2008. [Çevrim-içi: https://pdfs.semanticscholar.org/d876/c03318d97cce5de6ecc980b8d6449c4c9aed.pdf, Erişim Tarihi: 15.10.2016.]

Chua S. L., Chen D. T., & Wong A. F. L. (1999). Computer anxiety and its correlates: A meta- analysis. *Computers In Human Behavior, 15*(5), 609-623.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programmimg on young children's cognition. *Journal of Educational Psychology. 76*(6). 1051-1058.

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

Cuiumju, V. (2013). *App Inventor ile programlamaya giriş*. [Çevrim-içi: https://www.technopat.net/2013/05/20/app-inventor-programlamaya-giris/, Erişim Tarihi: 02.08.2016.]

Çağıltay, N. E. ve Fal, M. (2014). *Scratch ile programlama öğreniyorum*. Ankara: ODTÜ Yayıncılık.

Çakıroğlu, Ü., Sarı, E. ve Akkan, Y. (2011). *Üstün yetenekli öğrencilere programlama öğretiminin problem çözmeye katkısı konusunda öğretmen görüşleri*. 5th International Computer & Instructional Technologies Symposium, Fırat University, Elazığ, 22-24 Ekim 2011. [Çevrim-içi: http://web.firat.edu.tr/icits2011/papers/27868.pdf, Erişim tarihi: 5 Temmuz 2016.]

Çatlak, Ş., Tekdal, M. ve Baz, F. Ç. (2015). Scratch yazılımı ile programlama öğretiminin durumu: Bir doküman inceleme çalışması. *Journal of Instructional Technologies & Teacher Education*, *4*(3), 13-25.

Çetin, E. (2012). *Bilgisayar programlama eğitiminin çocukların problem çözme beceri üzerine etkisi.* Yayımlanmamış Yüksek Lisans Tezi. Gazi Üniversitesi, Eğitim Bilimleri Enstitüsü, Ankara.

D'Zurilla T. J., & Goldfried M. R. (1971). Problem solving and behavior modification. *Journal of Abnormal Psychology, 78*(1), 107-126.

Dalbey, J., & Linn, M. C. (1985). The demands and requirements of computer programming: A literature review. *Journal of Educational Computing Research, 1*(3), 253-274.

Dalton, D. W., & Goodrum, D. A. (1991). The effects of computer programming on problem-solving skills and attitudes. *Journal of Educational Computing Research*, *7*(4), 483-506.

Kereki, I. F. (2008). *Scratch: Applications in computer science 1*. 38th Annual Frontiers in Education Conference. Saratoga Springs, NY, October 22 – 25, 2008. [Çevrim-içi: http://www.ort.edu.uy/innovaportal/file/5553/1/fie2008.pdf, Erişim Tarihi: 20.09.2016.]

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

Dede, Y. ve Argün, Z. (2004). Öğrencilerin matematiğe yönelik içsel ve dışsal motivasyonlarının belirlenmesi. *Eğilim ve Bilim, 134*, 49-54.

Demir, F. (2015). *Programlama öğretiminde eğitsel programlama dilinin farklı kullanımlarının programlama başarısı ve kaygısına etkisi*. Doktora Tezi, Atatürk Üniversitesi, Eğitim Bilimleri Enstitüsü, Erzurum.

Demirel, Ö. (2003). *Planlamadan değerlendirmeye öğretme sanatı*. Ankara: Pegem Yayıncılık.

Demirer, V. ve Sak, N. (2016). Dünyada ve Türkiye'de programlama eğitimi ve yeni yaklaşımlar. *Eğitimde Kuram ve Uygulama, 12*(3), 521-546.

DuBoulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research, 2*(1), 57-73.

EARGED, (2011). *MEB 21.yy öğrenci profili*. [Çevrim-içi: http://www.meb.gov.tr/earged/earged/21.%20yy_og_pro.pdf, Erişim Tarihi: 14.05.2016.]

Erdoğan, B. (2005). *Programlama başarısı ile akademik başarı, genel yetenek, bilgisayara karşı tutum, cinsiyet ve lise türü arasındaki ilişkilerin incelenmesi*. Yayımlanmamış Yüksek Lisans Tezi, Marmara Üniversitesi, Eğitim Bilimleri Enstitüsü, İstanbul.

Erol, O. (2015). *Scratch ile programlama öğretiminin bilişim teknolojileri öğretmen adaylarının motivasyon ve başarılarına etkisi*. Yayımlanmamış Doktora Tezi. Anadolu Üniversitesi, Eğitim Bilimleri Enstitüsü, Eskişehir.

Ersoy, H. ve Aygın, S. (2015). *Ortaokul öğrencilerine programlama becerileri kazandırmada Scratch'ın etkililiği.* 17. Akademik Bilişim Konferansı. Anadolu Üniversitesi, Eskişehir-Türkiye, 4-6 Şubat 2015. [Çevrim-içi: https://goo.gl/rs4tsi, Erişim Tarihi: 20.07.2016.]

Ersoy, H., Madran, R. O. ve Gülbahar, Y. (2011). *Programlama dilleri öğretimine bir model önerisi: Robot programlama*. Akademik Bilişim'11-XIII. Akademik Bilişim Konferansı Bildirileri. Malatya İnönü Üniversitesi, 2-4 Şubat 2011. [Çevrim-içi: http://ab.org.tr/ab11/bildiri/145.pdf, Erişim tarihi: 5 Ağustos 2016.]

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

Esteves, M., & Mendes, A., (2004). *A Simulation Tool to Help Learning of Object Oriented Programming Basics*. In Proceedings of the 34th ASEE/IEEE Frontiers in Education Conference. Savannah, Georgia, 20-23 Ekim 2004. [Çevrim-içi: http://ieeexplore.ieee.org/document/1408649/, Erişim tarihi:3 Ağustos 2016.]

Euronews (2015). *Avrupa'da bilgisayar programlama dersleri ilköğretim müfredatına giriyor*. [Çevrim-içi: http://tr.euronews.com/2015/09/03/avrupa-da-bilgisayar-programlama-dersleri-ilkogretim-mufredatina-giriyor, Erişim Tarihi: 01.08.2016.]

Ferrer-Mico, T., Prats-Fernandez, M. A., & Redo-Sanchez, A. (2012). Impact of Scratch programming on students' understanding of their own learning process. *Procedia-Social and Behavioral Sciences*, *46*(1), 1219-1223.

Fesakis, G., & Serafeim, K. (2009). Influence of the familiarization with scratch on future teachers' opinions and attitudes about programming and ICT in education. In *ACM SIGCSE Bulletin*, *41*(3), 258-262.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education*, *63*, 87-97.

Fields, D., Vasudevan, V., & Kafai, Y. B. (2015). The programmers' collective: fostering participatory culture by making music videos in a high school Scratch coding workshop. *Interactive Learning Environments*, *23*(5), 613-633.

Fraenkel, J., & Wallen, N. (2003). *The nature of qualitative research. How to Design and Evaluate Research in Education*. 5th Ed., NY: McGraw Hill.

Friedman,, M., & Abelson, H. (2009). *What is MIT Application Inventor?*. [Çevrim-içi: http://www.unesco.org/new/fileadmin/MULTIMEDIA/HQ/CI/CI/pdf/In_Focus/mit_app-inventor.pdf, Erişim Tarihi: 02.07.2016.]

Geban, Ö., Aşkar, P., & Özkan, İ. (1992). Effects of computer simulations and problem solving approaches on high school students. *Journal of Educational Research, 86*(1), 5–10.

Genç, Z. ve Karakuş, S. (2011). *Tasarımla öğrenme: Eğitsel bilgisayar oyunları tasarımında Scratch kullanımı*. 5th International Computer & Instructional

Technologies Symposium (ICITS), Elazığ, 22-24 Ekim 2011. [Çevrim-içi: http://web.firat.edu.tr/icits2011/icits2011ProceedingBook.pdf, Erişim Tarihi: 15.07.2016.]

Giordano, D., & Maiorana, F. (2014). *Use of cutting edge educational tools for an initial programming course*. IEEE Global Engineering Education Conference (EDUCON). Military Museum and Cultural Center, Harbiye, Istanbul, Turkey, 3-5 April 2014. [Çevrim-içi:http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6826147, Erişim Tarihi: 19.11.2016.]

Giordano, D., & Maiorana, F. (2015). *Teaching algorithms: Visual language vs flowchart vs textual language*. IEEE Global Engineering Education Conference. Tallinn University of Technology, Tallinn, Estonia, 18-20 March 2015. [Çevrim-içi: http://ieeexplore.ieee.org/document/7096016/?arnumber=7096016&tag=1, Erişim Tarihi: 19.11.2016.]

Gomes, A., & Mendes, A. J. (2007). *Learning to program difficulties and solutions*. International conference on Engineering Education, Portugal, Coimbra, 3-7 Ekim 2007. [Çevrim-içi: http://icee2007.dei.uc.pt/proceedings/papers/411.pdf, Erişim tarihi: 5 Temmuz 2016.]

Green, S. B., & Salkind, N. J. (2005). *Using SPSS for Windows and Machintoch: Analyzing and understanding data*. New Jersey: Pearson.

Gupta, N., Tejovanth, N., & Murthy, P. (2012). *Learning by creating: Interactive programming for Indian high schools*. In Technology Enhanced Education (ICTEE), 2012 IEEE International Conference. 3-5 Jan. 2012. [Çevrim-içi: DOI: 10.1109/ICTEE.2012.6208643, Erişim Tarihi: 21.10.2016.]

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

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

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

Hatısaru, S. (2016). *Vodafone'dan çocuklara kod yazma dersi*. [Çevrim-içi: http://www.milliyet.com.tr/vodafone-dan-cocuklara-kod yazma/gundem/ydetay/2241732/default.htm, Erişim Tarihi: 01.08.2016.]

Helminen, J., & Malmi, L. (2010). *Jype-a program visualization and programming exercise tool for Python.* In Proceedings of the 5th international symposium on Software visualization. Salt Lake City, UT, USA, 25-26 Ekim 2010. [Çevrim-içi: http://dl.acm.org/citation.cfm?id=1879234, Erişim tarihi: 10 Ağustos 2016.]

Heppner, P. P., & Petersen, C. H. (1982). The development and implications of a personal problem-solving inventory. *Journal of Counseling Psychology, 29*(1), 66-75.

Hoegh, A., & Moskal, B. M. (2009). *Examining science and engineering students' attitudes toward computer science*. 39th IEEE Frontiers in Education Conference. San Antonio, TX, October 18 - 21, 2009. [Çevrim-içi: https://goo.gl/l0QDVF, Erişim Tarihi: 25.10.2016.]

Hongwarittorrn, N., & Krairit, D. (2010). *Effects of program visualization (jeliot3) on students' performance and attitudes towards java programming*. In The spring 8th International conference on Computing, Communication and Control Technologies, Orlando, Florida USA, 6-9 April 2010. [Çevrim-içi: http://www.iiis.org/CDs2010/CD2010IMC/CCCT_2010/PapersPdf/TA750PM.pdf, Erişim Tarihi: 11.09.2016.]

Hooshyar, D., Ahmad, R. B., Shamshirband, S., Yousefi, M., & Horng, S. J. (2015). A flowchart-based programming environment for improving problem solving skills of Cs minors in computer programming. *The Asian International Journal of Life Sciences*, *24*(2), 629-646.

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

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). Instructional strategy in the teaching of computer programming: A need assessment analyses. *TOJET*, *9*(2), 125-131.

İmal, N. ve Eser, M. (2009). *Programlama dili öğrenmedeki zorluklar ve çözüm yaklaşımları*. Elektrik Elektronik Bilgisayar Biyomedikal Mühendislikleri Eğitimi IV. Ulusal Sempozyumu. Eskişehir Osmangazi Üniversitesi Kongre ve Kültür Merkezi, Eskişehir, 22-24 Ekim 2009. [Çevrim-içi: http://www.emo.org.tr/ekler/8bd988bd20804a2_ek.pdf, Erişim tarihi: 11 Ağustos 2016.]

Jenkins, T. (2002). On the difficulty of learning to program. *In Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences*, *4*, 53-58.

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.

Kahraman, B. (2015). *Avustralya'da ilkokullarda programcılık eğitimi veriliyor*. [Çevrim-içi: http://www.webtekno.com/sektorel/avustralya-da-ilkokullarda-programcilik-egitimi-veriliyor-h10859.html, Erişim Tarihi: 01.08.2016.]

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

Karabak, D. ve Güneş, A. (2013). Ortaokul birinci sınıf öğrencileri için yazılım geliştirme alanında müfredat önerisi. *Journal of Research in Education and Teaching, 2*(3), 175-181.

Karasar, N. (2005). *Bilimsel araştırma yöntemleri*. Ankara: Nobel Yayınları.

Karasar, N. (2010). *Bilimsel Araştırma Yöntemi: Kavramlar, ilkeler, Teknikler.* Ankara: Nobel Yayınevi.

Karplus, R. (1977). Science teaching and the development of reasoning. *Journal of Research in Science Teaching, 14*(2), 169-175.

Kaucic, B., & Asic, T. (2011). *Improving introductory programming with Scratch?* Proceedings of the 34th International Convention. Opatija, Croatia, May, 23-27, 2011. [Çevrim-içi: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5967218, Erişim Tarihi: 26.10.2016.]

Kaya, M. (2011). *Problem ve problem çözme.* [Çevrim-içi: http://ismailenesaltn.blogspot.com.tr/2011/06/problem-ve-problem-cozme.html, Erişim Tarihi: 11.08.2016.]

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education*, *73(1)*, 26-39.

Kelleher, C. (1999). *Alice*. [Çevrim-içi: http://www.alice.org, Erişim Tarihi: 10.06.2016.]

Kelleher, C. (2006). *Motivating programming: Using story telling to make computer programming attractive to middle school girls*. Unpublished Doctoral Dissertation. School of Computer Science Carnegie Mellon University, Pittsburgh.

Kelleher, C., Pausch, R., & Kiesler, S. (2007). *Storytelling Alice motivates middle school girls to learn computer programming*. Proceedings of the SIGCHI Conference On Human Factors İn Computing Systems Table Of Contents. San Jose, California, USA, April 28-May 3, 2007. [Çevrim-içi: http://www.cse.wustl.edu/~ckelleher/StorytellingCHI.pdf, Erişim Tarihi: 15.10.2016.]

Kert, S. B. ve Uğraş, T. (2009). *Programlama eğitiminde sadelik ve eğlence: Scratch örneği*. The First International Congress of Educational Research. Çanakkale 18 Mart Üniversitesi, 1 Mayıs 2009. [Çevrim-içi: https://goo.gl/Qz3fbT, Erişim tarihi: 5 Ağustos 2016.]

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

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

Kinnunen, P., & Malmi, L. (2008). *CS minors in a CS1 course*. In Proceeding of the Fourth international Workshop on Computing Education Research Sydney,

Australia, 06 – 07 Ekim 2008. [Çevrim-içi: http://dl.acm.org/citation.cfm?id=1404529, Erişim tarihi: 11 Ağustos 2016.]

Klassen, M. (2006). *Visual approach for teaching programming concepts*. In Proceedings of the 9th International Conference on Engineering Education. San Juan, PR, July, 23-28, 2006. [Çevrim-içi: http://www.icee.usm.edu/ICEE/conferences/ICEE2006/papers/3119.pdf, Erişim Tarihi: 20.11.2016.]

Kneeland, S. (2001). *Problem çözme*. Ankara: Gazi Kitabevi.

Kobsiripat, W. (2015). Effects of the media to promote the scratch programming capabilities creativity of elementary school students. *Procedia-Social and Behavioral Sciences*, *174*, 227-232.

Kocaman, H. (2010). *Microsoft Small Basic*. [Çevrim-içi: http://www.hakankocaman.com/microsoft-small-basic, Erişim Tarihi: 06.09.2016.]

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

Korkmaz, Ö. (2016). The effect of scratch- and lego mindstorms Ev3-based programming activities on academic achievement, problem-solving skills and logical-mathematical thinking skills of students. *Malaysian Online Journal of Educational Sciences, 4*(3), 73-88.

Köklü, N. (2002). *Açıklamalı istatistik terimleri sözlüğü*. Ankara: Nobel Yayın Dağıtım.

Kukul, V. ve Gökçearslan, Ş. (2014). *Scratch ile programlama eğitimi alan öğrencilerin problem çözme becerilerinin incelenmesi*. 8th International Computer & Instructional Technologies Symposium. Trakya Üniversitesi, Edirne, 18-20 Eylül 2014. [Çevrim-içi: https://goo.gl/n2Iur8, Erişim Tarihi: 14.07.2016.]

Küçükoğlu, Ö. (2014). *21.yüzyıl becerileri için en etkin oluşumlar: "Kod okur-yazarlığı" ve "FLL"*. [Çevrim-içi: http://www.egitimdeteknoloji.com/21yuzyil-becerileri-kod-okur-yazarligi-ve-fll/, Erişim Tarihi: 10.08.2016.]

Lahtinen, E., Ala-Mutka, K., & Jarvinen, H. M. (2005). *A study of the difficulties of novice programmers*. Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education. New York, USA, 26-29 Haziran. [Çevrim-içi: http://dl.acm.org/citation.cfm?id=1067453, Erişim tarihi: 11 Ağustos 2016.]

Lai, A. F., & Yang, S. M. (2011). *The learning effect of visualized programming learning on 6 th graders' problem solving and logical reasoning abilitie*s. Electrical and Control Engineering (ICECE), 2011 International Conference. Ramada Yichang Hotel Yichang, China, September 16-18 2011. [Çevrim-içi: https://goo.gl/K4fc3a, Erişim Tarihi: 14.07.2016.]

Lai, C. S., & Lai, M. H. (2012). *Using computer programming to enhance science learning for 5th graders in Taipei*. In Computer, Consumer and Control (IS3C), 2012

International Symposium. 4-6 June 2012. [Çevrim-içi: 10.1109/IS3C.2012.45, Erişim Tarihi: 19.07.2016.]

Lau, W. W., & Yuen, A. H. (2009). Exploring the effects of gender and learning styles on computer programming performance: implications for programming pedagogy. *British Journal of Educational Technology, 40*(4), 696-712.

Liao, Y. C., & Bright, G. W. (1991). Effects of computer programming on cognitive outcomes: A meta analysis. *Journal of Educational Computing Research, 7*, 251-268.

Linn, M. C., & Clancy, M. J. (1992). The case for case studies of programming problems. *Communications of the ACM*, *35*(3), 121-132.

Mains, M. G. (1997). *The effects of learning a programming language on logical thinking skills*. Unpublished Doctoral Dissertation. University of Nevada.

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004). *Scratch: a sneak preview [education]*. *Creating, connecting and collaborating through computing,* 30-30 Jan. 2004. [Çevrim-içi: http://ieeexplore.ieee.org/document/1314376/, Erişim Tarihi: 02.08.2016.]

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. *ACM 40*(1), 367-371.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *Computer Education, 10*(4), 1-16.

Many, W. A., Lockard, J., Abrams, P. D., & Friker, W. (1988). The effect of learning to program in logo on reasoning skills of junior high school students. *Journal of Educational Computing Research, 4*(2), 203-212.

Martens, B., & Lemmens, S. (2014). *Starting from scratch: experimenting with computer science in Flemish secondary education*. InProceedings of the 9th Workshop in Primary and Secondary Computing Education. Berlin, Germany, November 05 - 07 2014. [Çevrim-içi: https://biblio.ugent.be/publication/5750465/file/5750486, Erişim Tarihi: 19.07.2016.]

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B. D., & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ACM SIGCSE Bulletin*, *33*(4), 125-180.

McGill, T. J., & Volet, S. E. (1997). A conceptiual framework for analyzing students' knowledge of programming. *Journal of Research on Computing in Education*, *29*(3), 276-197.

MEB, (2016). *Bilgisayar bilimi dersi öğretim programı kur 1 - kur 2*. [Çevrim-içi: http://ttkb.meb.gov.tr/www/ogretim-programlari/icerik/72, Erişim Tarihi: 06.10.2016.]

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M., (2010). *Learning computer science concepts with Scratch.* In Proceedings of the Sixth International Workshop on Computing Education Research (ICER '10). ACM, New York, 69-76.

Miller, R. B., Kelly, G. N., & Kelly, J. T. (1988). Effects of Logo computer programming experience on problem solving and spatial relations ability. *Contemporary Educational Psychology, 13*(4), 348-357.

Monroy-Hernández, A., & Resnick, M. (2008). Empowering kids to create and share programmable media. *ACM Digital Library, 15*(2), 50-53.

Morelli, R., De Lanerolle, T., Lake, P., Limardo, N., Tamotsu, E., & Uche, C. (2011). *Can android app inventor bring computational thinking to K-12*. In Proc. 42nd ACM technical symposium on Computer science education (SIGCSE'11). Dallas, Texas, USA, March 9-12, 2011. [Çevrim-içi: http://hermes.di.uoa.gr/gregor/file/appinventor_manuscript.pdf, Erişim Tarihi: 8.08.2016.]

Moreno-Leon, J., & Robles, G. (2015). *Computer programming as an educational tool in the English classroom a preliminary study*. In 2015 IEEE Global Engineering Education Conference (EDUCON). Tallinn University of Technology, Tallinn, Estonia, 18-20 March 2015. [Çevrim-içi: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7096089, Erişim Tarihi: 29.07.2016.]

Nam, D., Kim, Y., & Lee, T. (2010). *The effects of scaffolding-based courseware for the scratch programming learning on student problem solving skill*. 18th International Conference on Computers in Education. Putrajaya, Malaysia, November 29-December 3, 2010. [Çevrim-içi:https://goo.gl/ZylMej, Erişim Tarihi: 30.08.2016.]

Namlu, A. G. ve Ceyhan, E. (2002). *Bilgisayar kaygısı: Üniversite öğrencileri üzerinde bir çalışma*. Eskişehir: T.C. Anadolu Üniversitesi Yayınları, No:1353.

Nikiforos, S., Kontomaris, C., & Chorianopoulos, K. (2013). MIT Scratch: A Powerful tool for improving teaching of programming. *Conferrence on Informatics in Education*, 1-5.

Nikou, S. A., & Economides, A. A. (2014). *Transition in student motivation during a scratch and an app inventor course*. In 2014 IEEE Global Engineering Education Conference (EDUCON). Military Museum and Cultural Center, Harbiye, Istanbul, Turkey, 3-5 April 2014. [Çevrim-içi: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6826234, Erişim Tarihi: 03.09.2016.]

Oliva, J. M. (2003). The Structural Coherence of Students' Conceptions in Mechanics and Conceptual Change. *International Journal of Science Education, 25*(5), 539-561.

Olson, P. (2012). *Why Estonia has started teaching its first-graders to code*. [Çevrim-içi: http://www.forbes.com/sites/parmyolson/2012/09/06/why-estonia-has-started-teaching-its-first-graders-to-code/#125f6b675790, Erişim Tarihi: 02.08.2016.]

Ortiz-Colon, A. M., & Maroto Romo, J. L. (2016). Teaching with Scratch in Compulsory Secondary Education. *International Journal of Emerging Technologies in Learning*, *11*(2).

Osman, M. A., Loke, S. P., Zakaria, M. N., & Downe, A. G. (2012). *Secondary students' perfectionism and their response to different programming learning tools*. In Humanities, Science and Engineering (CHUSER). Kota Kinabalu, Sabah, Malaysia, December 3-4, 2012. [Çevrim-içi: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6504380, Erişim Tarihi: 17.11.2016.]

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A., & Lahmine, S. (2015). Learning basic programming concepts by creating games with scratch programming environment. *Procedia-Social and Behavioral Sciences*, *191*, 1479-1482.

Ozoran, D., Çağıltay, N. E., & Topallı, D. (2012). Using scratch in introduction to programming course for engineering students. *2nd International Engineering Education Conference (IEEC2012), 2,* 125-132.

Öçalan, H. (2015). *Avustralya'da kodlama dersi ilköğretim müfredatına girdi*. [Çevrim-içi: http://onedio.com/haber/avustralya-da-kodlama-dersi-ilkogretim-mufredatina-girdi-593639, Erişim Tarihi: 01.06.2016.]

Öndeş, Ö. (2016). *İngiltere ve ABD'de kodlama eğitimi*. [Çevrim-içi: http://www.hurriyet.com.tr/ingiltere-ve-abdde-kodlama-egitimi-40061515, Erişim Tarihi: 02.08.2016.]

Özçakmak, Ş. (2014). *Bilgisayar kullanımı çocuklukta bağımlılık yapar mı?* [Çevrim-içi: haberturk.com/polemik/haber/973204-bilgisayar-kullanimi-cocukta-bagimlilik-yapar-mi, Erişim Tarihi: 02.08.2016.]

Özdemir, A. (2015). *Kodlama, okuma yazma bilmek kadar önemli!* [Çevrim-içi: http://www.sozcu.com.tr/egitim/kodlama-okuma-yazma-bilmek-kadar-onemli.html, Erişim Tarihi: 07.10.2016.]

Özmen, B., & Altun, A. (2014). Undergraduate students' experiences in programming: difficulties and obstacles. *Turkish Online Journal of Qualitative Inquiry, 5*(3), 9-27.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Pae, R. D., & Kurland, D. M. (1983). *On the cognitive prerequisites of learning computer programming (Tech. Rep. No. 18)*. New York: Bank Street College of Education, Center for Children and Technology.

Pea, R. D., & Kurland, D. M. (1987). On the cognitive effects of learning computer programming: A critical look. *New Ideas Psychology, 2(2),* 137-168.

Peppler, K. A., & Kafai, Y. B. (2007). From supergoo to scratch: exploring creative digital media production in informal learning. *Learning, Media andTechnology*, *32*(2), 149-166.

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

Pinto, A., & Escudeiro, P. (2014). *The use of Scratch for the development of 21st century learning skills in ICT.* In Information Systems and Technologies (CISTI), 9th Iberian Conference. [Çevrim-içi:10.1109/CISTI.2014.6877061, Erişim Tarihi: 11.06.2016.]

Pintrich, P. R., & Schunk, D. H. (2002). *Motivastion in education: Theory, research and application* (2nd ed.). Upper Saddle River, Merrill–Prentice Hall.

Pintrich , P. R., Smith, D. A., Garcia, T., & McKeachie, W. J.(1991). *A manual for The use of the motivated strategies for learning questionnaire (MSLQ)*. Ann Arbor, MI: University of Michigan.

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

Rizvi, M., Humphries, T., Major, D., Jones, M., & Lauzun, H. (2011). A CS0 course using scratch. *Journal of Computing Sciences in Colleges*, *26*(3), 19-27.

Robbins, J. K. (2011). Problem solving, reasoning, and analytical thinking in a classroom environment*. The Behavior Analyst Today*, *12*(1), 41-48.

Romeike, R. (2007). Applying creativity ın cs high school education: Criteria, teaching example and evaluation. *InProceedings of the Seventh Baltic Sea Conference on Computing Education Research, 88,* 87-96.

Ruf, A., Muhling, A., & Hubwieser, P. (2014). *Scratch vs. Karel: Impact on learning outcomes and motivation*. Paper presented at the Proceedings of the 9th Workshop in Primary and Secondary Computing Education, Berlin, Germany, 5-7 November 2014. [Çevrim-içi: http://dl.acm.org/citation.cfm?id=2670772, Erişim Tarihi: 15.07.2016.]

Sainz, M., & Lopez-Saez, M. (2010). Gender differences in computer attitudes and the choice of technology-related occupations in a sample of secondary students in Spain. *Computers & Education, 54*(2), 578-587.

Sanjanaashree, P., Anand, K. M., & Soman, K. P. (2014).*Language learning for visual and auditory learners using scratch toolkit*. International Conference on Computer Communication and Informatics, Coimbatore, India, 03 – 05, 2014. [Çevrim-içi: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6921765, Erişim Tarihi: 05.09.2016.]

Sayın, Z. ve Seferoğlu, S. S. (2016). *Kodlama eğitimi ve kodlamanın eğitim politikalarına etkisi*. XVIII. Akademik Bilişim Konferansı (AB16). Adnan Menderes Üniversitesi, Aydın, 3-5 Şubat 2016. [Çevrim-içi: http://ab.org.tr/ab16/sunum/258.pdf, Erişim Tarihi:07.07.2016.]

Schunk, D. (2003). *Learning theories. An educational perspective*. Upper Saddle River: Merrill.

Scratch (2016). *Scratch about.* [Çevrim-içi: https://scratch.mit.edu/about, Erişim Tarihi: 20.08.2016.]

Sebetci, Ö. ve Aksu, G. (2014). Öğrencilerin mantıksal ve analitik düşünme becerilerinin programlama dilleri başarısına etkisi. *Eğitim Bilimleri ve Uygulama, 13*(25), 65-83.

Shin, S., & Park, P. (2014). A study on the effect affecting problem solving ability of primary students through the scratch programming. *Advanced Science and Technology Letters, 59*(1), 117-120.

Shin, S., Park, P., & Bae, Y. (2013). The effects of an information-technology gifted program on friendship using scratch programming language and clutter. *International Journal of Computer and Communication Engineering, 2*(3), 246.

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

Soylu, Y. ve Soylu, C. (2006). Matematik derslerinde başarıya giden yolda problem çözmenin rolü. *Eğitim Fakültesi Dergisi, 7*(11), 97-111.

SSRVM (2007). *Model curriculum and teaching material for K-12 Indian schools*. [Çevrim-içi: http://www.it.iitb.ac.in/~sri/papers/SSRVM-CS-March07.pdf , Erişim Tarihi: 01.08.2016.]

Sternberg, R. J., & Williams, W.M. (2002). *Educational Psychology*. USA: Allyn and Bacon.

Su, A. Y., Huang, C. S., Yang, S. J., Ding, T. J., & Hsieh, Y. Z. (2015). Effects of Annotations and homework on learning achievement: An empirical study of scratch programming Pedagogy. *Educational Technology & Society*, *18*(4), 331-343.

Su, A., Yang, S. J., Hwang, W. Y., Huang, C. S., & Tern, M. Y. (2014). Investigating the role of computer-supported annotation in problem-solving-based teaching: An empirical study of a Scratch programming pedagogy. *British Journal of Educational Technology*, *45*(4), 647-665.

Swain, P. E. (2013). RAPTOR-A vehicle to enhance logical thinking. *Journal of Environmental Hazards*, *7*(4), 353-359.

Şahin, N., Şahin, N. H., & Heppner, P. P. (1993). The psychometric properties of the Problem Solving Inventory. *Cognitive Therapy and Research*, *17*(4), 379-396.

Şendağ, S. ve Erol, O. (2012). *İlköğretim ikinci kademede alice 3 boyutlu animasyon yazılımının bilişsel araç olarak kullanımına yönelik bir durum çalışması*. 6 th International Computer & Instructional Technologies Symposium, Gaziantep University, Gazıantep, 4-6 Ekim 2012. [Çevrim-içi: http://www1.gantep.edu.tr/~icits2012/icits2012_proceeding.pdf, Erişim Tarihi: 02.05.2016.]

Şentürk, C. (2009). *Eğitimde yeniden yapılanma ve yapılandırmacılık*. [Çevrim-içi: http://www.efkandurmus.com/site/arsiv/57-23/83-egitimde-yeniden-yapilanma-ve-yapilandirmacilik.html, Erişim Tarihi: 20.06.2016.]

Taylan, E. (2013). *Kod okuryazarlığında genç yaşta eğitimin önemi ve online platformların etkisi.* [Çevrim-içi: http://webrazzi.com/2013/03/05/kod-okuryazarligi/, Erişim Tarihi: 10.08.2016.]

Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia-Social and Behavioral Sciences, 8*, 561-570.

TBD (2014). *Programlama Çocuk Oyuncağı*. [Çevrim-içi: http://egitimplatformu.aydin.edu.tr/gundem/haber_detay.asp?haberID=32, Erişim Tarihi: 01.08.2016.]

Tekerek, M., & Altan, T. (2014). The effect of scratch environment on student's achievement in teaching algorithm. *World Journal on Educational Technology*, *6*(2), 132-138.

Tobin, K. G., & Capie, W. (1981). The Development and Validation of a Group Test of Logical Thinking. *Educational and Psychological Measurement, 4*, 413-423.

Türnüklü, E. B. ve Yeşildere S. (2005). Problem, problem çözme ve eleştirel düşünme. *Gazi Eğitim Fakültesi Dergisi, 25*(3), 107-123.

Ünal, E. (2015). *Scratch nedir?* [Çevrim-içi: http://seset.ceit.metu.edu.tr/2015/01/scratch-nedir/, Erişim Tarihi:03.08.2016.]

Wang, H. Y., Huang, I., & Hwang, G. J. (2014). *Effects of an integrated Scratch and project-based learning approach on the learning achievements of gifted students in computer courses*. In Advanced Applied Informatics (IIAIAAI), 2014 IIAI 3rd International Conference. 31 Aug.-4 Sept. 2014. [Çevrim-içi: DOI: 10.1109/IIAI-AAI.2014.85, Erişim Tarihi: 20.10.2016.]

Wang,T., Mei. W, Lin, S., Chiu, S., & Lin, J. M. (2009). *Teaching programming concepts to high school students with Alice*. 39th ASEE/IEEE Frontiers in Education Conference, San Antonio, TX, 18-21 Oct. 2009. [Çevrim-içi: http://ieeexplore.ieee.org/document/5350486/, Erişim Tarihi: 02.08.2016.]

Wilson, A., Hainey, T., & Connolly, T. (2012). *Evaluation of computer games developed by primary school children to gauge understanding of programming concepts*. 6th European Conference on Games-based Learning (ECGBL), Cork, Ireland, October, 2012. [Çevrim-içi: http://search.proquest.com/openview/0f1489b8d69c80b6e6da0d0571832991/1?pq-origsite=gscholar, Erişim Tarihi: 27.10.2016.]

Wyffles, F., Martens, B., & Lemmens, S. (2014*). Starting from scratch: Experimenting with computer science ın flemish secondary education*. Proceedings of the 9th Workshop in Primary and Secondary Computing EducationACM. Berlin, Germany, November 05 - 07 2014. [Çevrim-içi: 10.1145/2670757.2670763, Erişim Tarihi: 15.07.2016.]

Yaman, S. ve Karamustafaoğlu, S. (2006). Öğretmen adaylarının mantıksal düşünme becerileri ve kimya dersine yönelik tutumlarının incelenmesi. *Erzincan Eğitim Fakültesi Dergisi, 8*(1), 91-106.

Yazıcı, H. ve Altun, F. (2013). The assocation between university students' ınternal and external motivation sources and their academic achievement. *International Journal of Social Science, 6*(6), 1241-1252.

Yenilmez, A., Sungur, S., & Tekkaya, C. (2006). Students' achievement in relation to reasoning ability, prior knowledge and gender. *Research in Science & Technological Education*, *24*(1), 129-38.

Yılmaz, H., & Çavaş, P. H. (2007). Reliability and validity study of the students' motivation toward science learning (smtsl) questionnaire. *Elementary Education Online, 6*(3), 430-440.

Yorulmaz, M. (2008). *İnternet kafelerin daha faydalı kullanılabilmeleri için bir öneri: Scratch*. Inet-tr'08 - XIII. Türkiye'de İnternet Konferansı, Orta Doğu Teknik Üniversitesi, Ankara, 22-23 Aralık 2008. [Çevrim-içi: http://inet-tr.org.tr/inetconf13/kitap/yorulmaz_inet08.pdf, Erişim Tarihi: 04.07.2016.]

Yükseltürk, E., & Altıok, S. (2016). Investigation of pre-service information technology teachers' game projects prepared with Scratch. *SDU International Journal of Educational Studies, 3*(1), 59-66.

Yüzak, Ö. (2016). *Kodlama yapacak çocuk yetiştirecek*. [Çevrim-içi: http://www.cumhuriyet.com.tr/haber/ekonomi/529954/Kodlama_yapacak_cocuk_yetistirecek.html, Erişim Tarihi: 01.08.2016.]

EKLER DİZİNİ

EK 1. PROGRAMLAMA ERİŞİ TESTİ

Adı Soyadı:…….. **Cinsiyet:** Kadın ☐ Erkek ☐

Numarası :……..

Şube : (A) (B)

…SORULAR…

Programlamada kullanılan;

1 "**Değişken**" kavramını tanımlayınız.

……
……
……

2 "**Operatör**" kavramını tanımlayarak programlamada kullanılan 3 operatör sembolünü yazınız.

……
……
……
……

3 "**Veri**" kavramını tanımlayarak programlamada kullanılan veri türlerinden 3 tanesini açıklayınız.

……
……
……
……

4 Programlamada kullanılan döngülerden 2 tanesini yazarak açıklayınız.

……
……
……
……

5 Aşağıda bir program geliştirirken yapılan adımlar verilmiştir. Bu adımları uygun şekilde sıralayınız.

a) Problemi / görevi anlama
b) Çözüm yöntemi geliştirme
c) Yöntemi bir programlama dili ile kodlama
d) Programı test edip varsa hataları düzeltme

1. Adım	2. Adım	3. Adım	4. Adım

6 Basit bir "bankamatik" algoritması yazılmak istendiğinde aşağıdaki boşluğa gelmesi gereken ifade seçeneklerden hangisinde yer almaktadır?

1- BAŞLA
2- YAZ "lütfen şifrenizi giriniz"
3- OKU *sifre*
4- EĞER *bilinen_sifre <> sifre* İSE **GİT 2**
5- YAZ "hoş geldiniz"
6- YAZ "lütfen çekmek istediğiniz para miktarını giriniz"
7- OKU *miktar*
8- **.....**
9- YAZ "Paranızı para haznesinden alınız - İyi günler"
10- BİTİR

a) EĞER *bankada_kalan_miktar > miktar* İSE GİT 10 DEĞİLSE GİT 9
b) EĞER *bankada_kalan_miktar >= miktar* İSE GİT 9 DEĞİLSE GİT 10
c) EĞER *bankada_kalan_miktar < miktar* İSE GİT 9 DEĞİLSE GİT 10
d) EĞER miktar = 5000 İSE GİT 10 DEĞİLSE GİT 9
e) EĞER miktar < 500 İSE GİT 10 DEĞİLSE GİT 9

7

1) BAŞLA
2) OKU sayi
3) .....
4) BİTİR

Yukarıdaki algoritmada **sayi = 5** ise ekrana "tek sayıdır" çıktısını vermek için 3 numaralı koda ne yazılmalıdır?

a) EGER sayi / 2 = 1 YAZ "tek sayıdır"
b) EGER sayi mod 2 = 1 YAZ "tek sayıdır"
c) EGER sayi < 2 YAZ "tek sayıdır"
d) EGER sayi * 1 = 1 YAZ "tek sayıdır"
e) EGER sayi mod 2 = 0 YAZ "tek sayıdır"

8 For (i=0; i<10; i++) ifadesi için aşağıdakilerden hangisi yanlıştır?

a) for: Döngünün türünü gösterir.
b) i=0: Döngünün başlangıç değerini gösterir.
c) i++: i'nin ikişer ikişer artacağını gösterir.
d) i<10: Döngünün bitiş değerini verir.
e) i++: i'nin birer birer artacağını gösterir.

---

9 Aşağıdaki programın çıktısı nedir?

```
Var a,b,c: Integer;
Begin
a=1; b=2; c=3;
a = b + c;   b = a - b;  c = a – b + c;
writeln(a,'  ',b,' ',c);
End.
```

a) 1 2 3
b) 5 3 6
c) 3 2 1
d) 5 3 5
e) 3 5 6

---

10 Aşağıdaki program verilen üç uzunluğun bir üçgenin kenar uzunlukları olup olamayacağını kontrol eder. Bir üçgende herhangi iki kenarın uzunluğu her zaman üçüncü kenardan daha uzundur. Boş bırakılan yerlere sırasıyla ne gelmelidir?

```
Var a,b,c: Integer;
Begin
Readln(a,b,c);
If (a+b>c)…?....(a+c>b)…?.... (b+c>a)
Then Writeln('Evet')
Else Writeln('Hayir');
end.
```

a) AND , OR
b) OR , OR
c) OR , AND
d) NOT , AND
e) AND , AND

---

11 Aşağıdaki programın çıktısı nedir?

```
Var a,b,d: Integer;
Begin
a:=3; b:=4; d:=2;
If (a<b) OR (b<d) Then
If (a>d) AND (b>d) Then Write('bir')
Else Write ('iki ')
Else If (b>a) AND (d>b) Then
Write ('iki')
Else If (a+b<d) Then Write ('uc ');
If (a<b) Then Write ('dört ')
Else Write ('beş');
End.
```

a) iki iki **b)** iki dört **c)** bir dört
d) bir iki **e)** bir beş

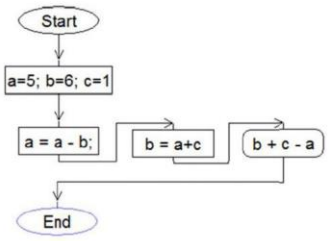
---

12 Aşağıdaki program çalıştığında ekrana ne yazar?

```
Var i:Integer;
Begin
For i:=0 to 9 do
Write (i,' ');
End.
```

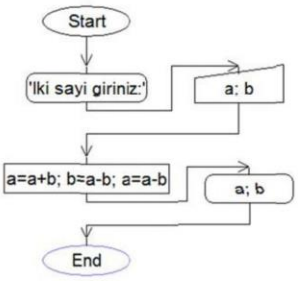
a) 0 1 2 3 4 5 6 7 8 9 10
b) 1 2 3 4 5 6 7 8 9 10
c) 0 1 2 3 4 5 6 7 8 9
d) 1 2 3 4 5 6 7 8 9
e) Yanıt belirsizdir.

13 Aşağıda verilen örneklerin çıktısı nedir? (13, 14 ve 15. Sorular)



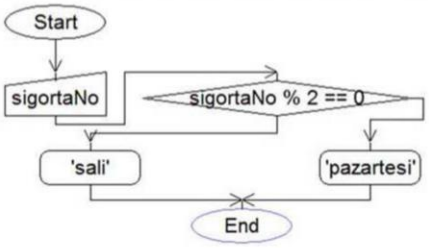
a)1 **b)** 2 **c)** 3 **d)** 4 **e)** 5

14



a) Girilen iki sayının değerlerini değiştirip ekrana yazar.
b) Girilen iki sayının önce büyük olanını sonra diğerini yazar.
c) Girilen iki sayının önce küçük olanını sonra diğerini yazar.
d) Girilen iki sayının eşit olup olmadıklarını kontrol eder.
e) Girilen iki sayıdan birincisi küçük ise, ikincinin değerini alır.

15



a) Sigorta numarasının son hanesi tek olanlar Salı günü maaş alırlar.
b) Sigorta numarasının son hanesi 0 olanlar Pazartesi günü maaş alırlar.
c) Sigorta numarasının son hanesi 2 olanlar Pazartesi günü maaş alırlar.
d) Sigorta numarası tek olanlar Salı günü diğerleri Pazartesi günü maaş alırlar
e) Sigorta numarası çift olanlar Salı günü diğerleri Pazartesi günü maaş alırlar

16 16, 17 ve 18. sorularda verilen programlama örneklerinin ekran çıktıları nedir?

```
Begin
a:=50; b:=15;
İf (a>80) or (b<80)  then b:=a-b
İf (a<70) and (b=15) then b:=10
İf a>60 then b:=b+5
İf a>50 then b:=a-b
İf b:=25;
Write(a-b);
End.
```

a) 10 **b)** 15 **c)** 20 **d)** 25 **e)** 30

17

```
Begin
a:=3; b:=22; c:=4;
if a<b then begin d:=a; a:=b; b:=d; end;
if a<c then begin d:=a; a:=c; c:=d; end;
if b<c then begin d:=b; b:=c; c:=d; end;
write(a); write(c); write(b);
End.
```

a) 2234
b) 3422
c) 2324
d) 4223
e) 2432

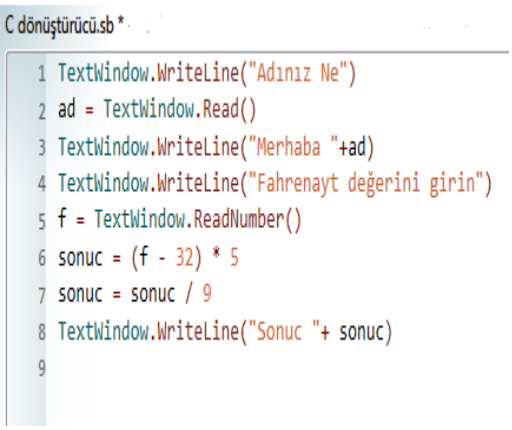
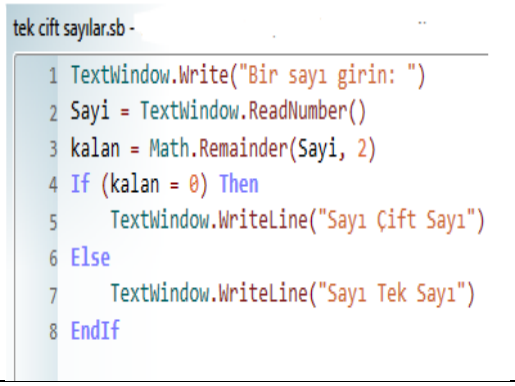
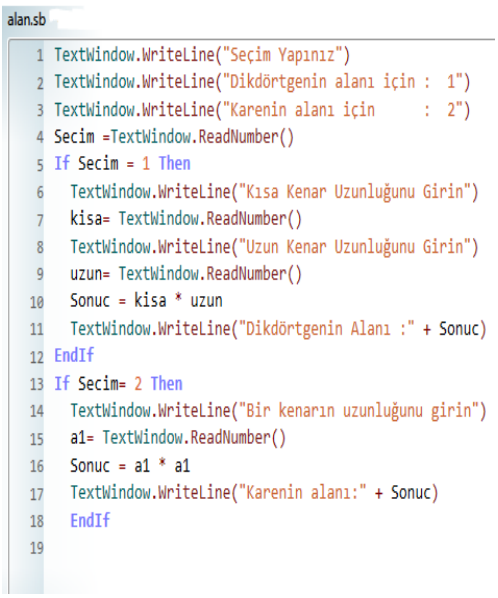
18

```
Begin
a:=1; b:=3; c:=57;
if a>b then begin write('A'); write('B'); end;
if b>c then write('C') else write('D');
if a+b>c then write('E')
else begin write('F'); write('G'); end;
if c-a>b then write('A')
else begin write('S'); write('T'); end;
End.
```

a) ABCD
b) CEFG
c) DEST
d) CDST
e) DFGA

Testin Cevap Anahtarı													
5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
abcd	b	b	e	d	e	c	c	b	a	e	b	a	e

EK 2. DERSLERDEN ÖRNEKLER

<table>
<tr><th>Blok Tabanlı Görsel Programlama (Scratch)</th><th>Metin Tabanlı Programlama (Small Basic)</th></tr>
<tr><td colspan="2">Derece dönüştürme</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Çift sayılar</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Alan hesaplama</td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

Negatif – pozitif sayılar

negatif pozitif.sb

```
1  TextWindow.Write("Bir sayı girin: ")
2  Sayi = TextWindow.ReadNumber()
3  If (Sayi = 0) Then
4  TextWindow.WriteLine("Sayı sıfıra eşit")
5  EndIf
6  If (Sayi<0) Then
7  TextWindow.WriteLine("sayı negatif")
8  EndIf
9  If (Sayi>0) Then
10 TextWindow.WriteLine("sayı pozitif")
11 EndIf
```

Büyük – küçük sayıyı bulma

büyük sayı.sb

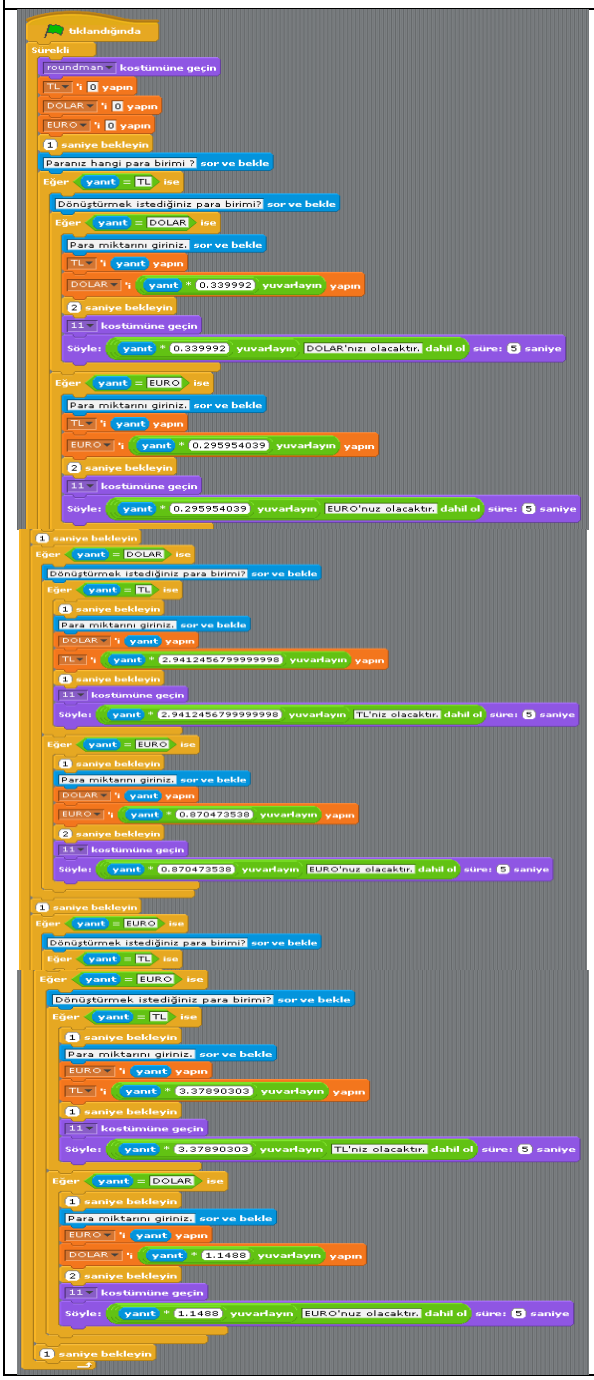
```
4  Sayi2 = TextWindow.ReadNumber()
5  TextWindow.Write("Üçüncü sayıyı girin: ")
6  Sayi3 = TextWindow.ReadNumber()
7  If (Sayi1 > Sayi2) And (Sayi1 > sayi3) Then
8  TextWindow.WriteLine("En büyük sayı "+Sayi1)
9  EndIf
10 If (Sayi2 > Sayi1) And (Sayi2 > sayi3) Then
11 TextWindow.WriteLine("En büyük sayı "+Sayi2)
12 EndIf
13 If (Sayi3 > Sayi2) And (Sayi3 > sayi1) Then
14 TextWindow.WriteLine("En büyük sayı "+Sayi3)
15 EndIf
```

Hesap makinesi

hesap makinesi.sb

```
1  TextWindow.WriteLine("Seçim Yapınız")
2  TextWindow.WriteLine("Toplama işlemi için   :  1")
3  TextWindow.WriteLine("Çıkarma işlemi için   :  2")
4  TextWindow.WriteLine("Çarpma işlemi için    :  3")
5  TextWindow.WriteLine("Bölme işlemi için     :  4")
6  Secim =TextWindow.ReadNumber()
7  If Secim = 1 Then
8    TextWindow.WriteLine("1.Sayıyı Giriniz")
9    S1= TextWindow.ReadNumber()
10   TextWindow.WriteLine("2.Sayıyı Giriniz")
11   S2= TextWindow.ReadNumber()
12   Sonuc = S1 +S2
13   TextWindow.WriteLine("Sonuc:" + Sonuc)
14 EndIf
15 If Secim= 2 Then
16   TextWindow.WriteLine("1.Sayıyı Giriniz")
17   S1= TextWindow.ReadNumber()
18   TextWindow.WriteLine("2.Sayıyı Giriniz")
19   S2= TextWindow.ReadNumber()
20   Sonuc = S1 - S2
21   TextWindow.WriteLine("Sonuc:" + Sonuc)
22   EndIf
23 If Secim= 3 Then
24   TextWindow.WriteLine("1.Sayıyı Giriniz")
25   S1= TextWindow.ReadNumber()
26   TextWindow.WriteLine("2.Sayıyı Giriniz")
27   S2= TextWindow.ReadNumber()
28   Sonuc = S1 * S2
29   TextWindow.WriteLine("Sonuc:" + Sonuc)
30   EndIf
31 If Secim= 4 Then
32   TextWindow.WriteLine("1.Sayıyı Giriniz")
33   S1= TextWindow.ReadNumber()
34   TextWindow.WriteLine("2.Sayıyı Giriniz")
35   S2= TextWindow.ReadNumber()
36   Sonuc = S1 / S2
37   TextWindow.WriteLine("Sonuc:" + Sonuc)
38   EndIf
```

Oyun (Sayı bulmaca)



sayı bulmaca oyunu.sb

```
1  TextWindow.Title="Sayı Bulmaca Oyunu (0-100)"
2  basla:
3  TextWindow.Clear()
4  TextWindow.WriteLine("0 ile 100 arasında bir sayı tuttum. 10 hakkın var. Kolaysa bul bakalım :)")
5  TextWindow.WriteLine("Sayı Gir:")
6  Sayi=Math.GetRandomNumber(100)
7  For x = 1 To 10
8  girilen = TextWindow.ReadNumber()
9  If (Sayi=girilen) Then
10     goto tebrik
11 EndIf
12 If (Sayi>girilen) Then
13   TextWindow.WriteLine("Daha büyük bir sayı girmelisin. " + (10-x) + " hakkın kaldı")
14 Else
15   TextWindow.WriteLine("Daha küçük bir sayı girmelisin. " + (10-x) + " hakkın kaldı")
16 EndIf
17 TextWindow.WriteLine("Sayı Gir:")
18 EndFor
19 TextWindow.WriteLine("Üzgünüm bilemedin. Enter tuşuna basıp yeniden oynayabilirsin")
20 TextWindow.Read()
21 Goto basla
22 tebrik:
23 TextWindow.WriteLine("Tebrikler! Bildiniz...")
24 TextWindow.WriteLine("Sayımız:" + Sayi + " idi")
25 TextWindow.WriteLine("Tekrar oynamak için Enter tuşuna basınız")
26 For z = 0 To 5
27 EndFor
28 TextWindow.Read()
29 Goto basla
30
```

EK 3. ETİK KURUL ONAY BİLDİRİMİ

T.C.
MUSTAFA KEMAL ÜNİVERSİTESİ
ÜNİVERSİTE ETİK KURULU KARARLARI

TARİH	TOPLANTI SAYISI	KARAR NO	SAYFA NO
18.05.2016	09	02	01/02

Üniversitemiz Etik Kurulu 18.05.2016 tarihinde Prof. Dr. Seval YAVUZ başkanlığında toplanarak aşağıdaki kararı almıştır.

KARAR-02 Üniversitemiz Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü öğretim elemanı Arş. Gör. Şenol ÇATLAK'ın yüksek lisans programında yürütmekte olduğu "Algoritma ve Programlama Öğrenmede Scratch Kullanımının Öğrenenlerin Ders Başarısı, Motivasyon, Mantıksal Düşünme ve Problem Çözme Becerilerine Etkisi" konulu tezi ile ilgili anket ve testleri Eğitim Fakültesi Fen Bilgisi Öğretmenliği 2. sınıf öğrencilerine uygulama talebi MKÜ Etik Kurul Yönergesine göre uygun görülmüş olup; Eğitim Fakültesi Dekanlığının izni ile uygulanmasının kabulüne; durumun başvuru sahibine ve Eğitim Fakültesi Dekanlığı'na bildirilmesine oy birliği ile, karar verilmiştir.

(İMZA)
Prof. Dr. Seval YAVUZ
Başkan

(İMZA)
Prof. Dr. Nafiz ÇELİKTAŞ
ÜYE

(İMZA)
Prof. Dr. Necmi İŞLER
ÜYE

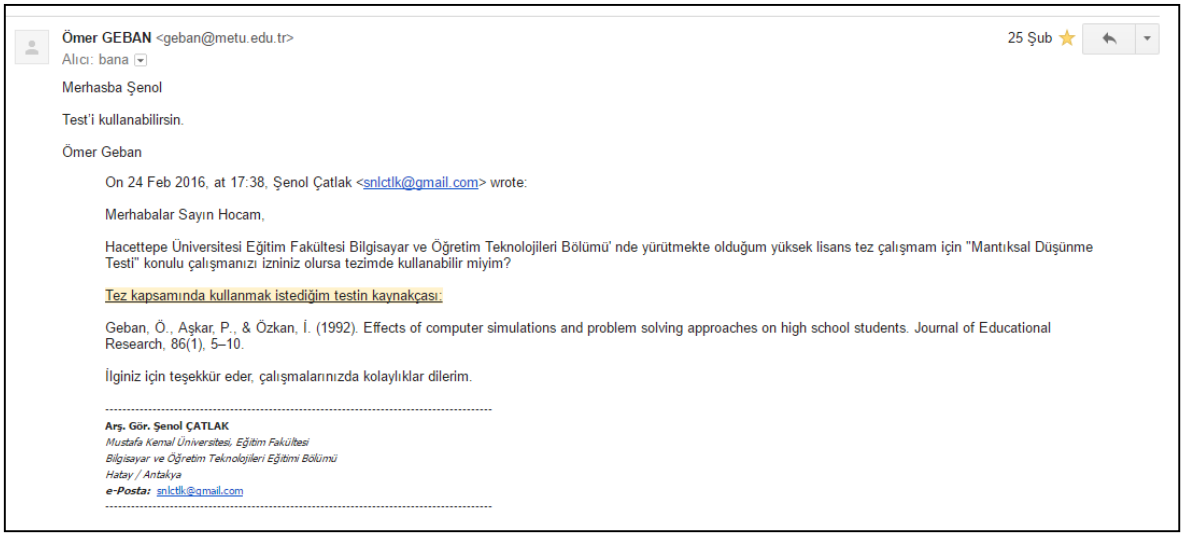
(İMZA)
Prof. Dr. Songül KAKİLLİ ACARAVCI
ÜYE

(İMZA)
Prof. Dr. Ayda TELLİOĞLU
ÜYE

(İMZA)
Doç. Dr. Akın YAKAN
ÜYE

(İMZA)
Doç. Dr. Alper ASLAN
ÜYE

EK 4. VERİ TOPLAMA ARAÇLARI KULLANIM İZİNLERİ



Ömer GEBAN <geban@metu.edu.tr> 25 Şub

Alıcı: bana

Merhasba Şenol

Test'i kullanabilirsin.

Ömer Geban

On 24 Feb 2016, at 17:38, Şenol Çatlak <snlctlk@gmail.com> wrote:

Merhabalar Sayın Hocam,

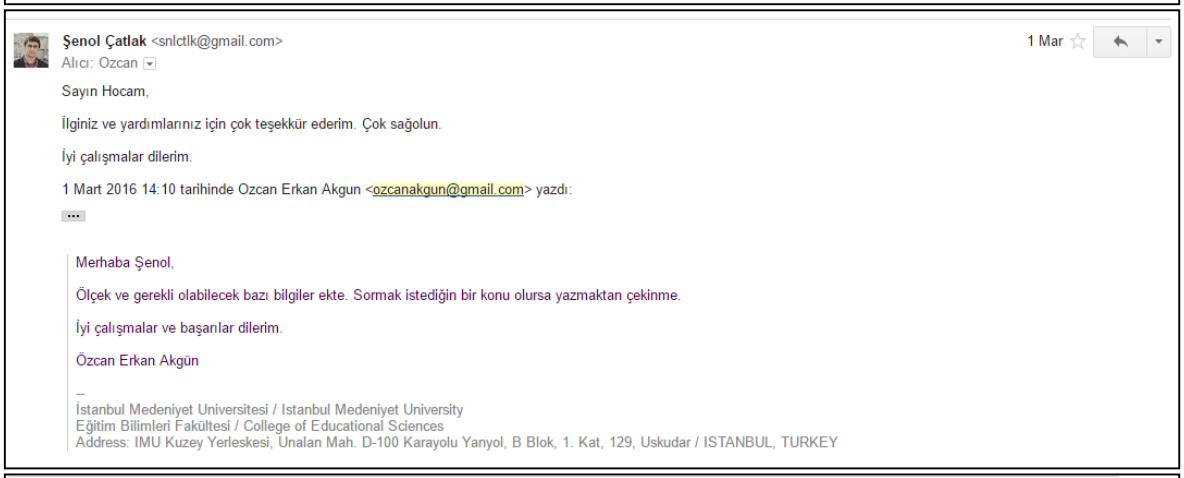
Hacettepe Üniversitesi Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Bölümü' nde yürütmekte olduğum yüksek lisans tez çalışmam için "Mantıksal Düşünme Testi" konulu çalışmanızı izniniz olursa tezimde kullanabilir miyim?

Tez kapsamında kullanmak istediğim testin kaynakçası:

Geban, Ö., Aşkar, P., & Özkan, İ. (1992). Effects of computer simulations and problem solving approaches on high school students. Journal of Educational Research, 86(1), 5–10.

İlginiz için teşekkür eder, çalışmalarınızda kolaylıklar dilerim.

Arş. Gör. Şenol ÇATLAK
Mustafa Kemal Üniversitesi, Eğitim Fakültesi
Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü
Hatay / Antakya
e-Posta: snlctlk@gmail.com



Şenol Çatlak <snlctlk@gmail.com> 1 Mar

Alıcı: Ozcan

Sayın Hocam,

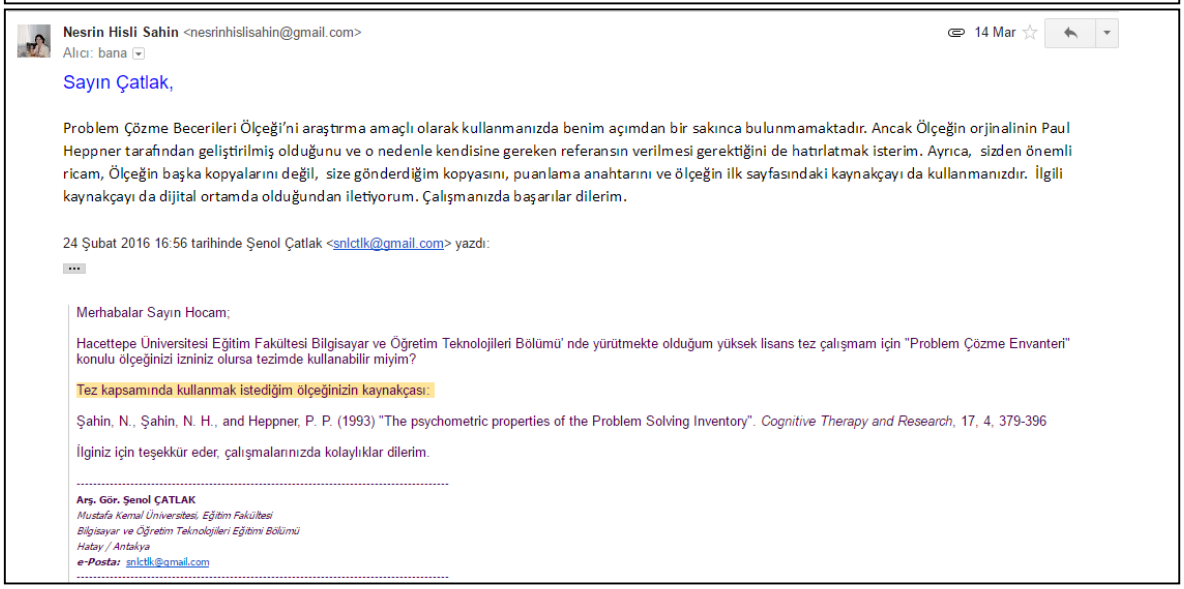
İlginiz ve yardımlarınız için çok teşekkür ederim. Çok sağolun.

İyi çalışmalar dilerim.

1 Mart 2016 14:10 tarihinde Ozcan Erkan Akgun <ozcanakgun@gmail.com> yazdı:

...

> Merhaba Şenol,
>
> Ölçek ve gerekli olabilecek bazı bilgiler ekte. Sormak istediğin bir konu olursa yazmaktan çekinme.
>
> İyi çalışmalar ve başarılar dilerim.
>
> Özcan Erkan Akgün
>
> --
> İstanbul Medeniyet Universitesi / Istanbul Medeniyet University
> Eğitim Bilimleri Fakültesi / College of Educational Sciences
> Address: IMU Kuzey Yerleskesi, Unalan Mah. D-100 Karayolu Yanyol, B Blok, 1. Kat, 129, Uskudar / ISTANBUL, TURKEY



Nesrin Hisli Sahin <nesrinhislisahin@gmail.com> 14 Mar

Alıcı: bana

Sayın Çatlak,

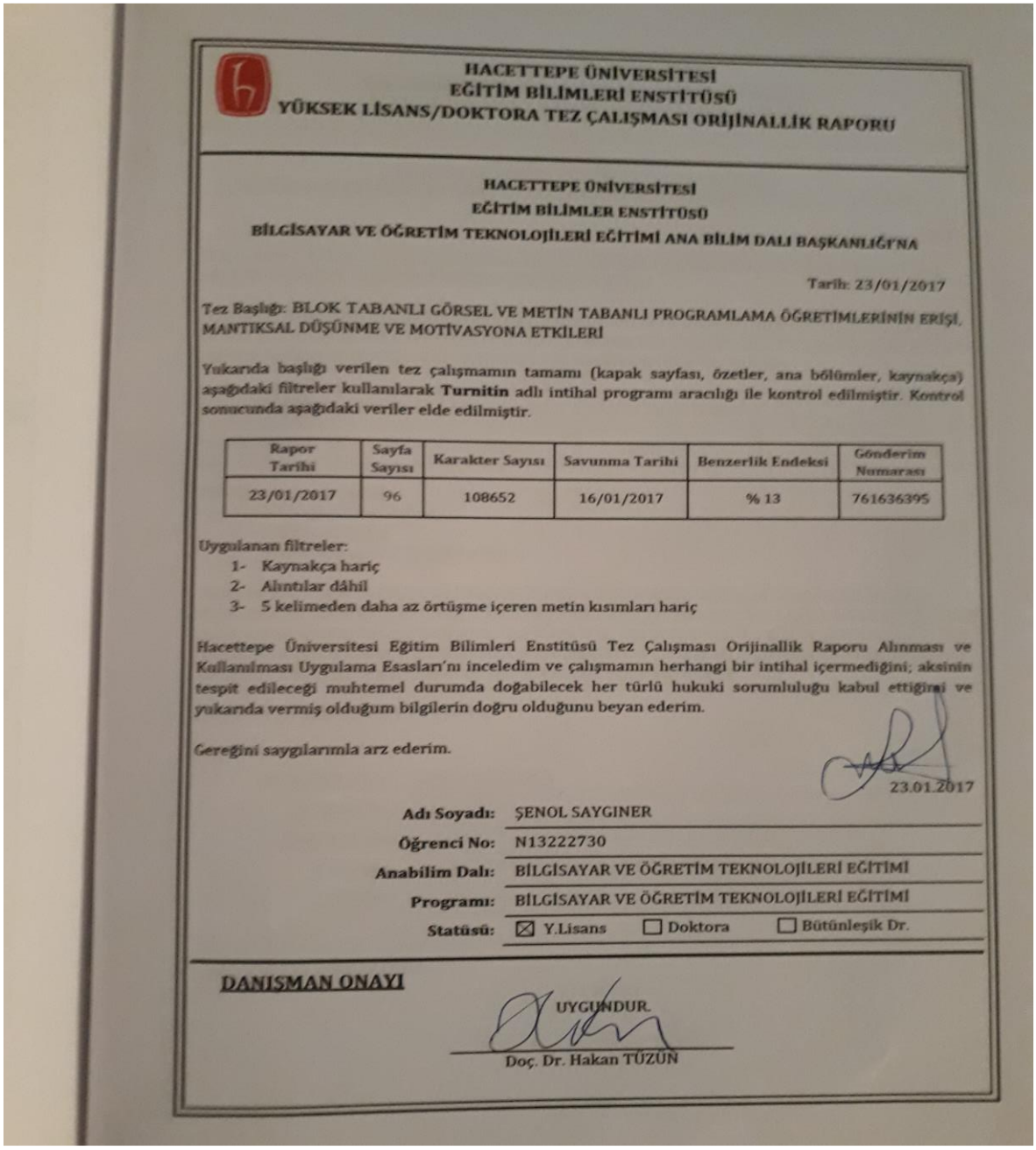
Problem Çözme Becerileri Ölçeği'ni araştırma amaçlı olarak kullanmanızda benim açımdan bir sakınca bulunmamaktadır. Ancak Ölçeğin orjinalinin Paul Heppner tarafından geliştirilmiş olduğunu ve o nedenle kendisine gereken referansın verilmesi gerektiğini de hatırlatmak isterim. Ayrıca, sizden önemli ricam, Ölçeğin başka kopyalarını değil, size gönderdiğim kopyasını, puanlama anahtarını ve ölçeğin ilk sayfasındaki kaynakçayı da kullanmanızdır. İlgili kaynakçayı da dijital ortamda olduğundan iletiyorum. Çalışmanızda başarılar dilerim.

24 Şubat 2016 16:56 tarihinde Şenol Çatlak <snlctlk@gmail.com> yazdı:

...

> Merhabalar Sayın Hocam;
>
> Hacettepe Üniversitesi Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Bölümü' nde yürütmekte olduğum yüksek lisans tez çalışmam için "Problem Çözme Envanteri" konulu ölçeğinizi izniniz olursa tezimde kullanabilir miyim?
>
> Tez kapsamında kullanmak istediğim ölçeğinizin kaynakçası:
>
> Şahin, N., Şahin, N. H., and Heppner, P. P. (1993) "The psychometric properties of the Problem Solving Inventory". *Cognitive Therapy and Research*, 17, 4, 379-396
>
> İlginiz için teşekkür eder, çalışmalarınızda kolaylıklar dilerim.
>
> ---
>
> **Arş. Gör. Şenol ÇATLAK**
> *Mustafa Kemal Üniversitesi, Eğitim Fakültesi*
> *Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü*
> *Hatay / Antakya*
> ***e-Posta:*** snlctlk@gmail.com
>
> ---

EK 5. ORJİNALLİK RAPORU



HACETTEPE ÜNİVERSİTESİ
EĞİTİM BİLİMLERİ ENSTİTÜSÜ
YÜKSEK LİSANS/DOKTORA TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

HACETTEPE ÜNİVERSİTESİ
EĞİTİM BİLİMLER ENSTİTÜSÜ
BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ ANA BİLİM DALI BAŞKANLIĞI'NA

Tarih: 23/01/2017

Tez Başlığı: BLOK TABANLI GÖRSEL VE METİN TABANLI PROGRAMLAMA ÖĞRETİMLERİNİN ERİŞİ, MANTIKSAL DÜŞÜNME VE MOTİVASYONA ETKİLERİ

Yukarıda başlığı verilen tez çalışmamın tamamı (kapak sayfası, özetler, ana bölümler, kaynakça) aşağıdaki filtreler kullanılarak **Turnitin** adlı intihal programı aracılığı ile kontrol edilmiştir. Kontrol sonucunda aşağıdaki veriler elde edilmiştir.

Rapor Tarihi	Sayfa Sayısı	Karakter Sayısı	Savunma Tarihi	Benzerlik Endeksi	Gönderim Numarası
23/01/2017	96	108652	16/01/2017	% 13	761636395

Uygulanan filtreler:
1- Kaynakça hariç
2- Alıntılar dâhil
3- 5 kelimeden daha az örtüşme içeren metin kısımları hariç

Hacettepe Üniversitesi Eğitim Bilimleri Enstitüsü Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Uygulama Esasları'nı inceledim ve çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Gereğini saygılarımla arz ederim.

23.01.2017

Adı Soyadı: ŞENOL SAYGINER
Öğrenci No: N13222730
Anabilim Dalı: BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ
Programı: BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ
Statüsü: ☒ Y.Lisans ☐ Doktora ☐ Bütünleşik Dr.

DANIŞMAN ONAYI

UYGUNDUR.

Doç. Dr. Hakan TÜZÜN

HACETTEPE UNIVERSITY
GRADUATE SCHOOL OF EDUCATIONAL SCIENCES
THESIS/DISSERTATION ORIGINALITY REPORT

HACETTEPE UNIVERSITY
GRADUATE SCHOOL OF EDUCATIONAL SCIENCES
TO THE DEPARTMENT OF COMPUTER EDUCATION AND INSTRUCTIONAL TECHNOLOGY

Date: 23/01/2017

Thesis Title : EFFECTS OF BLOCK-BASED VISUAL AND TEXT-BASED PROGRAMMING INSTRUCTION ON ACHIEVEMENT, LOGICAL THINKING AND MOTIVATION

The whole thesis that includes the *title page, introduction, main chapters, conclusions and bibliography section* is checked by using **Turnitin** plagiarism detection software take into the consideration requested filtering options. According to the originality report obtained data are as below.

Time Submitted	Page Count	Character Count	Date of Thesis Defence	Similarity Index	Submission ID
23/01/2017	96	108652	16/01/2017	% 13	761636395

Filtering options applied:

1. Bibliography excluded
2. Quotes excluded
3. Match size up to 5 words excluded

I declare that I have carefully read Hacettepe University Graduate School of Educational Sciences Guidelines for Obtaining and Using Thesis Originality Reports; that according to the maximum similarity index values specified in the Guidelines, my thesis does not include any form of plagiarism; that in any future detection of possible infringement of the regulations I accept all legal responsibility; and that all the information I have provided is correct to the best of my knowledge.

I respectfully submit this for approval.

23.01.2017

Name Surname: ŞENOL SAYGINER
Student No: N13222730
Department: COMPUTER EDUCATION AND INSTRUCTIONAL TECHNOLOGY
Program: COMPUTER EDUCATION AND INSTRUCTIONAL TECHNOLOGY
Status: ☒ Masters ☐ Ph.D. ☐ Integrated Ph.D.

ADVISOR APPROVAL

APPROVED

Assoc. Prof. Dr. Hakan TUZUN

EK 6. ÖZGEÇMİŞ

Kişisel Bilgiler

Adı Soyadı	Şenol SAYGINER
Doğum Yeri	Hatay
Doğum Yılı	1990
Medeni Hali	Bekâr

Eğitim Durumu

Lise	Antakya Anadolu Teknik Lisesi
Lisans	Çukurova Üniversitesi
Yüksek Lisans	Hacettepe Üniversitesi
Yabancı Dil	İngilizce

İş Deneyimi

Çalıştığı Kurumlar	Mustafa Kemal Üniversitesi – Araştırma Görevlisi	2014 -

Akademik Çalışmalar

Yayınlar (Ulusal, uluslararası makale, bildiri, poster vb gibi.)

Saygıner Ş. ve Tekdal M. (2016). *Programlama öğretimi için bulut tabanlı entegre geliştirme ortamları.* Fourth International Instructional Technologies & Teacher Education Symposium, Elazığ, Türkiye, 6-8 Ekim 2016.

Tekdal, M., Çatlak, Ş., Kadirhan, Z. ve Kılıç, M. (2016). *Açık kaynak kodlu yazılımlarda çeviri ve uyarlama: XERTE Online ToolKits örneği*. International Conference on New Trends in Educational Technology-INTET2016, 03-04 May, Famagusta, CYPRUS.

Tekdal, M., & Saygıner, Ş. (2016). *Augmented reality use in educational meaning: A content analysis study.* 10th International Computer and Instructional Technologies Symposium (ICITS). Rize, Turkey, 16-18 May 2016.

Tekdal M. ve Saygıner Ş. (2016). *Öğrenme ve öğretme sürecinde mobil teknolojilerin kullanımı*. Fourth International Instructional Technologies & Teacher Education Symposium, Elazığ, Türkiye, 6-8 Ekim 2016.

Çatlak, Ş., Tekdal, M. ve Baz, F. Ç. (2015). Scratch yazılımı ile programlama öğretiminin durumu: Bir doküman inceleme çalışması. *Journal of Instructional Technologies & Teacher Education*, *4*(3), 13-25.

Tekdal, M., Baz, F. Ç., & Çatlak, Ş. (2015). Current MOOC platforms at online education. *International Journal of Scientific and Technological Research,1*(2), 144-149.

Tekdal M. ve Çatlak Ş. (2014). *MOOC platformlarının karşılaştırılması: Bir durum değerlendirmesi*. 8. Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumu. Edirne, Türkiye, 18-20 Eylül 2014.

İletişim

e-posta Adresi	senolsayginer@gmail.com