Scratch'ın 5. Sınıf Öğrencilerinin Algoritma Geliştirme ve Bilgi-İşlemsel Düşünme Becerilerine Etkisi

Makale Geçmişi

Makale geliş tarihi: 18 Haziran 2016
Yayına kabul tarihi: 8 Şubat 2017
Çevrimiçi yayın tarihi: 28 Şubat 2018

Ali Oluk[1], Özgen Korkmaz[2] ve Hayriye Ayşe Oluk[3]

Öz: Bu çalışmanın amacı Scratch kullanımının algoritma geliştirme ve bilgi-işlemsel düşünme becerilerini geliştirmede etkisini incelemektir. Araştırma öntest sontest kontrol gruplu yarı deneysel bir çalışma olup 31 deney 31 kontrol grubu olmak üzere 62 tane 5. Sınıf öğrencisi ile yürütülmüştür. Araştırma 5. sınıf bilişim teknolojileri ve yazılım dersinde 6 haftalık bir süreci kapsayacak şekilde tasarlanmıştır. Bu süreçte deney grubu öğrencilerine Scratch programı kullanılarak algoritma anlatılırken, kontrol grubu öğrencilerine mevcut müfredata göre algoritma anlatılmıştır. Öğrencilere öntest ve sontest olarak bilgi-işlemsel düşünme ölçeği ve algoritma geliştirme başarı testi uygulanmıştır. Bilgi-işlemsel düşünme ölçeğinin gerekli faktör ve güvenirlik analizleri yapılmış ve Cronbach Alpha güvenirlik katsayısı 0.809 olarak belirlenmiştir. Algoritma geliştirme başarı testinin Kr-20 iç tutarlılık kat sayısı ise 0.85 olarak bulunmuştur. Grupların başarı testleri arasında istatistiksel olarak bir fark olup olmadığını anlamak için bağımsız t testi uygulanmıştır. Çalışma sonucu olarak deney grubu öğrencilerinin algoritma geliştirme ve bilgi-işlemsel düşünme becerilerinin kontrol grubuna göre anlamlı derecede daha fazla yükseldiği bulunmuştur. Bu durumda Scratch programının algoritma geliştirme ve bilgi-işlemsel düşünme becerilerini geliştirmek için kullanılabilir bir öğrenme aracı olduğu söylenebilir.

Anahtar Kelimeler: Bilgi-işlemsel düşünme, Scratch, algoritma geliştirme, programlama

DOI: 10.16949/turkbilmat.399588

Abstract: This study aimed to investigate the effect of using Scratch on developing skills related to algorithm development and computational thinking. The quasi experimental study with a pretest-posttest control group design was carried out with 62 (31assigned to Experimental and assigned to Control group respectively) 5th grader students. The study was conducted in a 6-week period during information technologies and software classes. The experimental group students were taught algorithms by using the Scratch program while control group students learned algorithms with the help of the current curriculum. Students were administered Computational Thinking Scale and Algorithm Development Achievement Test as pre and post-tests. The required factor and reliability analyses were carried out for computational thinking scale and Cronbach Alpha reliability coefficient was calculated as 0.809. KR-20 internal consistency coefficient for Algorithm Development Achievement Test was found to be 0.85. Independent Samples t test was administered to see whether there was a statistically significant difference between the groups' achievement test results. Results show that experimental group students' skills regarding algorithm development and computational thinking significantly developed compared to those of the control group. Based on the findings, it can be argued that Scratch Program is a learning tool that can be used to develop skills related to algorithm development and computational thinking.

Keywords: Computational thinking, Scratch, algorithm development, programming

See Extended Abstract

[1] Öğr. Gör. Kastamonu Üniversitesi, Taşköprü Meslek Yüksekokulu, alioluk85@gmail.com
[2] Doç. Dr. Amasya Üniversitesi, Teknoloji Fakültesi, Bilgisayar Mühendisliği, ozgenkorkmaz@gmail.com
[3] Öğretmen, Kastamonu Atatürk Ortaokulu, Bilişim Teknolojileri, hayriyeayse@hotmail.com

1. Giriş

Gelişen teknoloji ile birlikte öğrencilerden beklenen temel becerilerde değişmekte ve eğitim sistemlerinde bu becerileri kazandırmak için yeni çalışmalar yapıldığı bilinmektedir. Türkiye'de eğitim sisteminde seçmeli ders kapsamından çıkarılıp zorunlu ders haline getirilen bilişim teknolojileri ve yazılım dersi de bu çalışmalardan biri olarak düşünülebilir. Dersin genel amaç ve yeterliliklerine bakıldığında bilişim okuryazarlığı, bilişim teknolojilerini kullanarak iletişim kurma, bilgi paylaşma, kendini ifade edebilme, araştırma, bilgiyi yapılandırma, işbirlikli çalışma, problem çözme, programlama ve özgün ürün geliştirme becerilerinin kazandırılmasının hedeflendiği görülmektedir (Milli Eğitim Bakanlığı [MEB], 2012). Bu hedefler son yıllarda birçok ülkenin müfredatlarına dahil etmeye çalıştıkları bilgi-işlemsel düşünme becerilerinin alt becerileri ile benzerlik göstermektedir (León & Robles, 2015). Bilgi-işlemsel düşünme, temel bilgisayar bilimlerinin ışığında problem çözme, sistem tasarlama ve insan davranışlarını anlamaya çalışmaktadır (Wing, 2006). ISTE (International Society for Technology in Education) (2015) bilgi-işlemsel düşünmenin yaratıcı düşünme, algoritmik düşünme, eleştirel düşünme, işbirlikli öğrenme ve iletişim becerileri gibi alt beceriler olmadan tanımlanamayacağına değinmektedir. Bu beceriler çağımız öğrencilerinden beklenen temel beceriler ile de örtüşmektedir (Günüç, Odabaşı ve Kuzu, 2013). Bilgi-işlemsel düşünme bilgisayarı kullanarak problem çözme kapasitesini artırmak için yaratıcılık, mantıklı düşünme, eleştirel düşünme gibi becerileri öne çıkarmayı hedeflemektedir (Korkmaz, Çakır, Özden, Oluk ve Sarıoğlu, 2015). Bilgi-işlemsel düşünmenin tanımlarına bakıldığında alt becerileri ile çağımız öğrencilerinden beklenen becerilerinin birbirine benzer oldukları görülmektedir. Öğrencilere bu becerileri kazandırmak için programlama eğitiminin önemli bir yeri olduğu bilinmektedir (Akpınar ve Altun, 2014; Karabak ve Güneş, 2013; Shin, Park & Bae, 2013). Bu bağlamda bilgi-işlemsel düşünme becerileri kazandırmak için programlama eğitimi önemli bir araç olduğu söylenebilir (Lye & Koh, 2014).

Programlama öğrenmeye yeni başlayacak olan öğrencilerin birçoğu programlamayı ileri düzeyde eğitim alan kişilerin başarabileceği zor bir iş olarak görmektedirler (Genç ve Karakuş, 2011). Buna sebep olarak genelleme, soyutlama, eleştirel düşünme gibi birçok becerinin programlamada bir arada kullanma gerekliliği gösterilebilir (Gomes & Mendes, 2007). Programlama öğrenmeyi daha kolay bir hale getirmek için görselliği öne çıkaran Scratch, Alice, Microsoft Small Basic ve Toontalk gibi programlama araçları bulunmaktadır (Çatlak, Tekdal ve Baz, 2015). Bireylerin programlama becerisini kazanabilmeleri için üst düzey düşünme becerilerine sahip olmaları, problemler için farklı çözüm yolları bulabilmeleri ve çözümün en kestirme yolunu keşfedebilmeleri gerekmektedir (Yükseltürk ve Altıok, 2015). Çözümün en kestirme yolunu bulmak için kişinin algoritma mantığını iyi bir şekilde anlaması gerekmektedir. Bu nedenle programlama öğretiminde kazandırılması gereken en önemli becerilerin algoritma ve akış şeması kavramlarının öğretimi olduğu söylenebilir (Köse ve Tüfekçi, 2015). Öğrencilere algoritma mantığı ve algoritmik düşünmeyi kazandırmak için Scratch kullanılabilir bir araç olduğu bilinmektedir (Armoni, Meerbaum-Salant & Ben-Ari, 2015; Maloney, Resnick, Rusk, Silverman & Eastmond, 2010). Ücretsiz ve basit bir kullanıma sahip olan

bu görsel programlama araçları ile ilköğretim çağı öğrencilerine programlama öğretimi için önemli bir avantaj sağladığı düşünülebilir. Bu görsel programlama araçlarından Scratch, program arayüzü ve sitesinde sunduğu Türkçe dil desteği sebebiyle Türk öğrenciler için bir adım öne çıkmaktadır (Karabak ve Güneş, 2013). Scratch, programlama mantığı ve algoritmik düşünce yeteneği kazandıran, dünyada programlama eğitimine başlayanlar için önerilen bir programdır (Çağıltay Ercil ve Fal, 2013; Garner, 2009).

Scratch 2003 yılında Massachusetts Institute of Technology (MIT) laboratuvarlarında geliştirilmeye başlanmış olan ücretsiz bir görsel programlama aracıdır. Web sitesi sayesinde hazırlanan projeler paylaşılıp diğer kullanıcılar ile iletişim sağlanabilmektedir. Scratch programlama aracı bir çok dil desteğini bünyesinde barındırmaktadır. Sürekle bırak mantığı ile çalışan Scratch programlama aracı programlama bilgisi olmayan kişiler için dahi cazip bir araçtır (Resnick ve ark., 2009). Literatürde Scratch programı kullanılarak algoritma geliştirme ve programlama öğretimine yönelik yapılmış çalışmalar bulunmaktadır (Begosso & Silva, 2013; Calder, 2010; Federici, 2011; Kaučič & Asič, 2011; Korkmaz, 2016; Maloney ve ark., 2010; Nikou & Economides, 2014; Ozoran, Çağıltay & Topalli, 2012; Su, Huang, Yang, Ding & Hsieh, 2015). Bu çalışmalardan Begosso ve Silva (2013), Maloney ve arkadaşları (2010) Scratch programlama aracı yardımı ile 8 ile 16 yaş arasında değişen öğrencilere algoritma geliştirme ve programlama öğretimine yönelik bir çalışma yapmışlardır. Ozoran ve arkadaşları (2012) yaptıkları çalışmada Scratch programlama aracı ile üniversite öğrencilerine algoritma geliştirme ve programlama eğitimi vermişlerdir. Scratch programlama aracının sadece programlama öğretimi üzerindeki etkisini inceleyen çalışmalarda olduğu bilinmektedir (Federici, 2011; Kaučič & Asič, 2011; Su ve ark., 2015). Fakat programlama ve algoritma geliştirme gibi önemli bir konu olan bilgi-işlemsel düşünme becerilerinin Scratch programı kullanarak gelişim gösterip göstermediğini araştıran alan yazınında çok sınırlıdır. Brennan ve Resnick (2012) oluşturdukları online topluluk üyelerinin Scratch programlama aracı ile geliştirdikleri çalışmaların bilgi-işlemsel düşünme açısından nasıl bir gelişim gösterdiğini ve bilgi-işlemsel düşünmeyi geliştirmek için Scratch programlama aracının nasıl kullanılabileceğini araştırmışlardır. Leon ve Robles yaptıkları çalışmada Scratch sitesinde yer alan rastgele seçilmiş 100 projeyi Dr. Scratch web aracı ile değerlendirmiş ve Scratch projelerinin bilgi-işlemsel düşünme becerilerini ölçmede ne derece etkili olduğunu anlamaya çalışmışlardır. Bu çalışmada Scratch programlama aracının algoritma geliştirme becerisi üzerindeki etkisinin yanında bilgi-işlemsel düşünme becerisi üzerindeki etkisinin de incelenmesi ile benzerlerinden ayrılmaktadır. Bu çerçevede bu çalışmanın amacı Scratch kullanımının algoritma geliştirme ve bilgi-işlemsel düşünme becerilerini geliştirmede etkisini incelemektir. Çalışmamızın alt problemleri şunlardır:

- 5. sınıf bilişim teknolojileri ve yazılım dersinde Scratch kullanımının algoritma geliştirme becerisi kazandırmaya etkisi var mıdır?
- 5. sınıf bilişim teknolojileri ve yazılım dersinde Scratch kullanımının bilgi-işlemsel düşünme becerisi kazandırmaya etkisi var mıdır?

2. Yöntem

Yapılan çalışmada öntest – sontest kontrol gruplu yarı deneysel bir desen kullanılmıştır. Deneysel desenler araştırmacı tarafından oluşturulan farkların bağımlı değişken üzerindeki etkisini ve değişkenler arasında oluşturulan neden sonuç ilişkisini test etme amacıyla kullanılır (Büyüköztürk, Kılıç Çakmak, Akgün, Karadeniz ve Demirel, 2014). Deney grubu öğrencilerine bilişim teknolojileri ve yazılım dersinde algoritma konusu Scratch programı kullanılarak ders işlemeye uygun bir plan hazırlanmıştır. Kontrol grubu öğrencilerine ise mevcut müfredata göre bir ders planı tasarlanarak işlenmiştir. Müfredat içerisinde öğrencilerin algoritma geliştirme mantığının kavratılması, farklı algoritmalar içerisinden en uygun olanın seçilebilmesi ve hatalı algoritmaların düzeltilerek problemin çözümüne uygun hale getirilme becerilerinin kazanması hedeflenmektedir. Bu çalışmada her iki gruba bu becerileri kazandırmak için uygun çalışma planları hazırlanmıştır. Uygulama yapılacak olan 5. sınıflarda 2 saat bilişim teknolojileri ve yazılım dersi vardır. Uygulama süreci 6 haftalık bir zamanı kapsamaktadır.

2.1. Araştırma Grubu

Çalışma grubu öğrencileri Kastamonu ilinde yer alan bir ortaokulun 5. Sınıf öğrencilerinden oluşmaktadır. Bu öğrencilerden 31'i deney 31'i kontrol grubunda yer almaktadır.

Tablo 1. Deney ve kontrol grubu öğrencilerinin cinsiyet dağılım tablosu

Grup	*Cinsiyet*				
	Kız		Erkek		Toplam
	N	%	N	%	N
Kontrol	16	51.6	15	48.4	31
Deney	16	51.6	15	48.4	31
Toplam	32	51.6	30	48.4	62

Kontrol grubu; 16 kız 15 erkek öğrenciden oluşmaktadır. Kız öğrenciler kontrol grubunun %51,6'sını, erkek öğrenciler ise % 48,4'ünü oluşturmaktadır. Deney grubu; 16 kız 15 erkek öğrenciden oluşmaktadır. Kız öğrenciler deney grubunun % 51,6'ini, erkek öğrenciler ise % 48,4'ünü oluşturmaktadır.

2.2. Veri Toplama Araçları

Veri toplama aracı olarak öğrencilere öntest sontest olarak Bilgisayarca Düşünme Ölçeği ve Algoritma Geliştirme Başarı Testi uygulanmıştır.

Bilgisayarca Düşünme Ölçeği (BDÖ): (BDÖ) Korkmaz, Çakır ve Özden (2015) tarafından ortaokul düzeyindeki öğrencileri bilgi-işlemsel düşünme beceri düzeylerini betimlemek amacıyla geliştirilmiştir. Ölçme aracının gerekli faktör, geçerlilik ve güvenirlik analizleri yapılmıştır. Ölçek yaratıcılık, problem çözme, algoritmik düşünme, işbirliklilik ve eleştirel düşünme olmak üzere 5 faktör 22 maddeden oluşan beş dereceli Likert tip bir ölçektir.. Ölçeğin maksimum likelihood tekniği kullanılarak yapılan

doğrulayıcı faktör analizi sonucu maddelerin regresyon değerleri 0.507 ile 0.872 arasında değişmektedir. Madde test korelasyon katsayıları 0.655 ile 0.862 arasında değişmektedir. Bu katsayılara göre maddelerin ölçeğin genel amacına hizmet ettiği söylenebilir. BDÖ' nin güvenirliğini hesaplamak için veriler üzerinde iç güvenirlik analizleri yapılmış olup ölçeğin Cronbach Alpha güvenirlik katsayısı 0.809 olarak belirlenmiştir.

Algoritma Geliştirme Başarı Testi (AGBT): Çalışma için araştırmacılar tarafından 27 maddeden oluşan 4 şıklı bir başarı testi oluşturulmuştur. Milli Eğitim Bakanlığı Bilişim Teknolojileri ve Yazılım dersi öğretim programında yer alan kazanımlara (MEB, 2012) göre oluşturulan başarı testi öğrencilerin algoritma geliştirmeleri, akış şeması geliştirmeleri, hatalı akış şeması ve algoritmaları düzeltmeleri, mevcut algoritmaları istenilen özelliğe göre yeniden dizayn etmeleri ve algoritma ve akış şemalarının temel özelliklerini bilip bilmediklerini ölçmek için geliştirilmiştir. Bu kazanımların ölçülmesi amacı ile geliştirilen AGBT içeriğinde aşağıda yer alan sorular ve benzerleri öğrencilere sorulmuştur.

Tablo 2. Örnek AGBT soruları

1. Öğrencinin e-okul sistemine girmesi için yapması gereken işlemler için oluşturulacak akış şeması sıralaması hangi şıkta doğru olarak verilmiştir?

1- Öğrenci numarası, T.C. kimlik numarası ve doğrulama kodu girilir

2- Tarayıcı açılır 3- Web Sitesine Girilir 4- Giriş düğmesine basılır

5- Başla 6- Bitir

A – 5-2-3-1-4 -6 B – 5-2-1-3-4-6

C – 5-1-3-4-3-6 D – 5-2-4-1-3-6

2. Bilgisayara girilen iki sayıyı toplayıp ekrana yazdıran programın algoritması aşağıdakilerden hangisidir?

A – ADIM 1: Başla
ADIM 2: a ve b'yi oku
ADIM 3: Sayıları Topla Toplam=a+b
ADIM 4: Yaz "a+b"
ADIM 5: Bitti

B– ADIM 1: Başla
ADIM 2: a ve b'yi oku
ADIM 3: Sayıları Topla Toplam=a+b
ADIM 4: Yaz "Toplam"
ADIM 5: Bitti

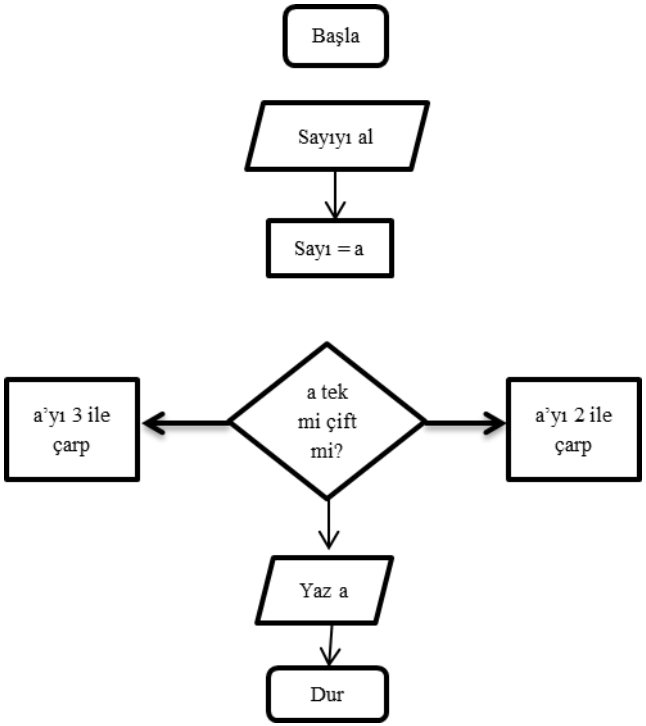
C– ADIM 1: Başla
ADIM 2: a ve b'yi oku
ADIM 3: Sayıları Topla Toplam=a-b
ADIM 4: Yaz "a+b"
ADIM 5: Bitti

D– ADIM 1: Başla
ADIM 2: a ve b'yi oku
ADIM 3: Sayıları Çarp Toplam=a*b
ADIM 4: Yaz "Toplam"
ADIM 5: Bitti

Tablo 2'nin devamı

3. Aşağıdaki algoritmada hatalı bir kısım var ise aşağıdaki şıklardan hangisi yapılırsa algoritma sorunsuz bir şekilde çalışır.
ADIM 1.Başla
ADIM 2. a, b ve c'yi oku
ADIM 3.Eğer(a=b) ve (a=c) ise Yaz"Eşkenar"
ADIM 4.Değilse Eğer (a=c) ve (b=c) ise yaz"ikiz kenar"
ADIM 5.Değilse yaz"çeşitkenar"
ADIM 6.Bitti
A – 3. Satıra İkizkenar yazılması gerekir
B – 5. Satıra Eşkenar yazılması gerekir
C – 4. Satırdaki ve yerine veya yazılması gerekir
D – Algoritmada bir hata yok değişiklik yapılmaması gerekir

4.



Yukarıdaki akış şemasına göre a sayısı 2 ile çarpılıyor olması ne ifade etmektedir?

A – a sayısının tek olduğunu
B – a sayısının çift olduğunu
C – a sayısının negatif olduğunu
D – a sayısının pozitif olduğunu

Yukarıda yer alan sorular ve benzerlerinden oluşan testimiz daha önce bilişim teknolojileri ve yazılım dersinde algoritma geliştirme konusunda eğitim almış 105 6. Sınıf öğrencisi ile yapılan pilot çalışma ile madde ayırt edicilikleri 0.30'un altında bulunan 7 madde testten çıkarılmıştır. Madde ayırt edicilikleri 0.33 ile 0.48 arasında değişen 20 madde ile başarı testi son halini almıştır. Hazırlanan testin madde güçlük indeksi ortalama puanı 0.66, Kr-20 iç tutarlılık kat sayısı ise 0.85 olarak bulunmuştur. Tablo 3'te başarı testinde yer alan maddelerin ayırt edicilik ve güçlük indisleri sunulmuştur.

Tablo 3. AGBT maddelerin ayırt edicilik ve güçlük indisleri

Maddeler	Madde Ayırıcılık Gücü	Madde güçlük İndeksi
M1	0.33	0.65
M2	0.33	0.83
M3	0.37	0.54
M4	0.33	0.66
M5	0.44	0.67
M6	0.33	0.80
M7	0.37	0.58
M8	0.37	0.70
M9	0.40	0.58
M10	0.33	0.67
M11	0.40	0.61
M12	0.37	0.59
M13	0.33	0.83
M14	0.33	0.58
M15	0.48	0.67
M16	0.37	0.67
M17	0.33	0.69
M18	0.37	0.61
M19	0.33	0.62
M20	0.33	0.69

2.3. Deneysel Uygulama

Çalışma için planlanan uygulamalar aşağıda belirtildiği gibi sırası ile öğrencilere uygulanmıştır. Uygulama haftalarında ikinci haftadan itibaren her iki gruba da dersin bir kısmında öğretmen tarafından ders anlatılmış diğer kısımda ise öğrencilerin kendilerinden istenilen görevleri yapmaları için zaman verilmiştir. Öğrenciler uygulamalarını yaparken öğretmen rehberlik yapmış öğrencilere herhangi bir müdahalede bulunmamıştır.

Hafta 1

İlk hafta derste deney ve kontrol grubu öğrencilerine problem kavramını, problem çözmek için ne gerekli olduğunu, problem çözmek için hangi basamakları takip etmemiz gerektiğini, algoritma ve akış şemasının ne olduğu, neden algoritma ve akış şeması

geliştirmemiz gerektiği, algoritma ve akış şeması geliştirmenin faydaları gibi temel bilgileri verildi. Bu bilgilerden sonra basit bir algoritma ve akış şeması örneği verilerek bu örnek üzerinde kullanılan algoritma ve akış şeması basamaklarındaki Başla, Bitir, değişken, adımların takibi ve okların ne işe yaradıkları anlatıldı.

Hafta 2

Uygulamanın ikinci haftası deney grubu öğrencilerine Scratch programının arayüzü anlatıldı ve Scratch ile daha önce yapılmış olan bazı örnekler gösterildi. Dersin devamında öğrencilere algoritma ve akış şemasında başla bitir kavramları tekrar anlatıldı. Bunun Scratch programında kontrol bloklarında karşılığı anlatıldı. Öğrencilere hareket, görünüm ve operatörler bloklarında yer alan özellikler anlatıldı. Öğrencilere değişken kavramı anlatıldı. Bu kavramın Scratch programındaki karşılığı gösterildi. Öğrencilere 1 ile 10 arasında rastgele oluşturulan iki sayının toplamını yine bir değişkene atayarak ekranda nasıl gösterilebileceği Scratch programı ile gösterildi. Daha sonra algoritma ve akış şeması örneklerinde değişken kavramı gösterilerek Scratch yardımı ile yeni değişkenler oluşturmaları sağlandı.

Şekil 1. Scratch programı kod blokları örneği

Kontrol grubu öğrencilere değişken kavramı anlatıldı. Değişken kavramı içeren algoritma ve akış şeması örnekleri gösterildi. Öğrencilere değişken ve akış şemalarının aktif kullanabilecekleri örnek problemler verilerek algoritma ve akış şemasını çizmeleri istendi. Daha sonra üçerli gruplar oluşturularak öğrencilerin yaptıkları örnekleri değiştirmeleri istendi ve arkadaşlarının yaptıkları hatalar varsa tespit etmeleri sağlandı.

Hafta 3 - 4

Deney grubu öğrencilerine Algoritma ve akış şemalarında kullanılan basit döngü örnekleri gösterildi. Gösterilen döngülerin Scratch programında yer alan kontrol bloğundaki bölümden tekrar gösterildi. Öğrencilerin algoritma ve akış şemalarında karşılaştırma işlemlerinin nasıl yapıldığını anlamaları için kontrol, görünüm ve algılama bloklarında yer alan özellikler anlatıldı. Örnek olarak 1 ile 100 arasında rastgele belirlenen iki sayıdan hangisinin büyük olduğunu ekrana gösteren programı Scratch yardımı ile sınıfta oluşturuldu.

Şekil 2. Scratch programı kod blokları örneği

Şekil 3. Çalışan programın ekran görüntüsü

Kontrol grubu öğrencilerine basit döngü örnekleri gösterildi. Döngü mantığı anlatılmaya çalışıldı. Öğrencilere aritmetik ve mantıksal operatörlerin nasıl kullanıldığı hakkında örnekler üzerinde bilgiler verildi. Dersin diğer kısmında öğrencilerin gruplar halinde içerisinde döngü kullanılabilecek bir problem cümlesi bulmaları ve buna uygun bir algoritma ve akış şeması geliştirmeleri istendi. Çözümler öğretmen tarafından toplandı ve grupların problem cümlelerini aralarında değiştirmeleri sağlandı. Daha sonra her grubun bu yeni problem cümlesine uygun bir algoritma ve akış şeması geliştirmesi istendi. Son olarak öğretmen tarafından aynı soruya yanıt olan iki akış şemasıda tahtaya çizilerek iki gruba yaptıkları hataların bulunması ve çözümü için fırsat verildi.

Hafta 5

Deney grubu öğrencilerine algoritma ve akış şemalarında kullanılan sayaç mantığının kavranması için birkaç örnek gösterildi. Daha sonra Scratch programının değişkenler ve

kontrol blokları içerisinde yer alan özelikler yardımı ile sınıfta sayaç mantığına uygun bir örnek tasarlandı. Öğrencilerden öğrenilen bilgiler ışığında Scratch programında kendi fikirleri ile bir uygulama geliştirmeleri istendi.

Kontrol grubu öğrencilerine sayaç mantığını anlatmak için öncelikle lokantalarda kullanılan adisyon kartlarından dağıtılarak sayaç mantığını anlamaları sağlandı. Daha sonra verilen problem cümlesine uygun olarak bir algoritma ve alkış şeması geliştirilerek öğrencilere dikkat etmeleri gereken yerler gösterildi. Ardından oluşturulan problem cümlelerine uygun algoritma ve akış şemalarını deftere çizmeleri istendi, daha sonra öğrencileri gruplara ayırarak her grubun sıra ile çıkarak verilen örnekteki algoritma ve akış şemasını basamaklar halinde oluşturması sağlandı. En sonunda ise hatalı basamakların tespiti sağlandı.

Hafta 6

Deney grubu öğrencilerine daha önceden hazırlanmış ve istenilen şekilde çalışmayan Scratch proje örnekleri verildi ve düzeltilmesi istendi. Kontrol grubu öğrencilerine ise hatalı algoritma ve akış şeması örnekleri verildi ve bunları düzeltmeleri istendi. Daha sonra her öğrencinin eline akış şeması içerisinde yer alan bir blok verildi ve bunları sıraya dizerek probleme uygun bir akış şeması geliştirmeleri istendi.

Deney ve kontrol grubu öğrencilerine sontest olarak AGBT testi ile BDÖ uygulanarak çalışma sonlandırıldı.

2.4 Verilerin Analizi

Verilerin analizi için SPSS 12.0 programı kullanılmıştır. Grupların başarı testleri arasında istatistiksel olarak bir fark olup olmadığını anlamak için bağımsız gruplar t testi uygulanmıştır. Bu test iki ilişkisiz örneklem ortalaması arasında anlamlı bir farklılık olup olmadığı ya da grupların sürekli değişken üzerinden aldıkları değerlerin karşılaştırılması için kullanılır (Seçer, 2013). t testinin bu çalışma grubu üzerinde uygulanıp uygulanamayacağına dönük varsayımlar test edilmiş ve grubun normal dağılım gösterdiği, varyanslarının eşit olduğu tespit edilmiştir. Bu çerçevede t testinin kullanılabileceği sonucuna varılmıştır.

3. Bulgular

3.1.Grupların Algoritma Geliştirme ve Bilgi-işlemsel Düşünme Becerilerinin Denkliğine İlişkin Bulgular

Öğrencilerin AGBT öntest puanları arasında anlamlı bir farklılık olup olmadığını anlamak için öntest verilerine bağımsız t testi uygulanmıştır. Test sonucu elde edilen veriler Tablo 4'te verilmiştir.

Tablo 4. AGBT Öntest puanlarının gruplara göre analizi

Grup	N	$\bar{X}$	S	*Sd*	*t*	*p*
Deney	31	31.45	9.05	60	1.81	.074
Kontrol	31	26.93	10.46			

Tablo 4'te görüldüğü gibi AGBT öntest puanları arasında kontrol ve deney grubu arasında istatistiksel olarak anlamlı bir farklılık bulunmamaktadır, $t(60) = 1.81, p > .05$. Öğrencilerin BDÖ öntest puanları arasında anlamlı bir farklılık olup olmadığını anlamak için öntest verilerine bağımsız t testi uygulanmıştır. Test sonucu elde edilen veriler Tablo 5'te verilmiştir.

Tablo 5. BDÖ öntest puanlarının gruplara göre analizi

Grup	N	$\bar{X}$	S	*Sd*	*t*	*p*
Deney	31	86.58	18.66	60	.933	.354
Kontrol	31	90.51	14.23			

Tablo 5'te görüldüğü gibi BDÖ öntest puanları arasında kontrol ve deney grubu arasında istatistiksel olarak anlamlı bir farklılık bulunmamaktadır, $t(60) = .933, p > .05$.

3.2. Algoritma Geliştirme Becerilerine ilişkin Bulgular

Grupların öntest puanları arasında anlamlı olmamakla birlikte, bir farklılaşma gözlenmektedir. Deney Grubu ($\overline{X}$ =31.45), Kontrol Grubu($\overline{X}$ =26.93). Bu farkın kontrol altına alınabilmesi için bu kısımda sontest-öntest fark puanları kullanılmıştır. Öğrencilerin AGBT öntest sontest başarı puanı farkları arasında anlamlı bir farklılık olup olmadığını anlamak için başarı puanı farklarına bağımsız t testi uygulanmıştır. Test sonucu elde edilen verilere Tablo 6'da verilmiştir.

Tablo 6. AGBT öntest sontest puan farkının gruplara göre analizi

Grup	N	$\bar{X}$	S	*Sd*	*t*	*p*
Deney	31	42.58	7.83	60	10.63	.000
Kontrol	31	23.87	5.87			

Tablo 6'da görüldüğü gibi AGBT öntest sontest puanı farkları incelendiğinde deney ve kontrol grubu arasından istatistiksel olarak anlamlı bir farklılık bulunmaktadır, $t(60) = 10.63, p < .01$. Deney grubunun AGBT öntest sontest puan farklarının ($\bar{X} = 42.58$), kontrol grubuna ($\bar{X} = 23.87$) göre daha yüksek olduğu görülmektedir. Bu durumda 5. Sınıf bilişim teknolojileri ve yazılım dersinde Scratch kullanımının algoritma geliştirme üzerinde normal müfredata göre işlenen derse göre daha olumlu bir sonuç verdiği söylenebilir.

3.3. Bilgi-işlemsel Düşünme Becerilerine ilişkin Bulgular

Öğrencilerin BDÖ öntest sontest puan farkları arasında anlamlı bir farklılık olup olmadığını anlamak için başarı puanı farklarına bağımsız t testi uygulanmıştır. Test sonucu elde edilen verilere Tablo 7'de verilmiştir.

Tablo 7. BDÖ öntest sontest puan farkının gruplara göre analizi

BDÖ Alt Boyutları	Grup	N	$\bar{X}$	S	Sd	t	p
Yaratıcılık	Deney	31	1.80	3.94	60	.809	.422
	Kontrol	31	1.09	2.87			
Algoritmik Düşünme	Deney	31	1.41	4.18	60	.987	.328
	Kontrol	31	0.51	2.90			
İşbirliklilik	Deney	31	2.48	4.13	60	1.602	.114
	Kontrol	31	1.00	3.07			
Eleştirel Düşünme	Deney	31	0.96	4.38	60	.065	.948
	Kontrol	31	1.03	3.31			
Problem Çözme	Deney	31	2.77	6.30	60	2.044	.045
	Kontrol	31	-1.74	10.56			
Toplam	Deney	31	9.45	13.57	60	2.202	.032
	Kontrol	31	1.90	13.41			

Tablo 7'de görüldüğü gibi BDÖ alt boyutlarının deney grubu öğrencilerinde ve kontrol grubu öğrencilerinde bir yükselme olduğu görülmektedir. Bu yükselmeler bütün alt boyutlarda deney grubu lehine olduğu görülmektedir. Bu fark Yaratıcılık ($t(60) = .809, p > .05$), Algoritmik Düşünme ($t(60) = .987, p > .05$), İşbirliklilik ($t(60) = 1.602,\ p > .05$) ve Eleştirel Düşünme ($t(60) = 2.044, p > .05$) alt boyutlarında anlamlı bir farklılık ifade etmemektedir. Problem çözme alt boyutunda ise deney ve kontrol grubu arasındaki fark deney grubu lehine istatistiksel olarak bir anlam ifade etmektedir ($t(60) = 2.044, p < .05$). BDÖ'nin sontest öntest toplam puanları arasındaki farka baktığımızda deney grubu ($\bar{X} = 9.45$) öğrencilerinin kontrol grubu ($\bar{X} = 1.90$) öğrencilerine göre daha yüksek bir puan farkına sahip oldukları görülmektedir. İki grup arasındaki bu farkı incelendiğinde deney ve kontrol grubu arasında istatistiksel olarak anlamlı bir farklılık olduğu görülmektedir, $t(60) = 2.202, p < .05$.

4. Tartışma ve Sonuç

Programlama becerisinin öğrenilmesi zor olduğunun düşünülmesinde problem çözme yaklaşımları ve çözüm üretebilme becerilerinin etkin biçimde öğrenilememesi sebep olmaktadır (Pillay & Jugoo, 2005). Programcıların problem çözme ve farklı çözüm yolları üretebilmeleri için algoritma mantığını iyi bir şekilde kavramaları gerekmektedir. Bu sebeple programlama öğretiminde öncelikle üst düzey programlama dili ve bunun kuralları yerine algoritma mantığının kazandırılmasına önem verilmelidir (Yükseltürk ve Altıok, 2016). Bu çalışmada da bilişim teknolojileri ve yazılım dersinde Scratch programlama aracı kullanımının algoritma geliştirme ve bilgi-işlemsel düşünme becerisi geliştirme üzerindeki etkisi incelenmiştir.

Deney grubu öğrencilerinin BDÖ öntest sontest puan farkının kontrol grubu öğrencilerinin puan farkından yüksek olduğu bulunmuştur. Veriler incelendiğinde deney grubu lehine olan BDÖ puanları arasındaki farkın istatistiksel olarak anlamlı olduğu görülmektedir. Bu durumda Scratch kullanımının bilgi-işlemsel düşünmeyi geliştirmede

olumlu bir etkisi olduğu söylenebilir. Öğrencilerin algoritma geliştirme testi öntest sontest puanlarının farkları incelendiğinde deney grubu öğrencilerinin puan farkının kontrol grubu öğrencilerine göre daha yüksek olduğu görülmektedir. İki grubun AGBT puan farkları istatistiksel olarak bir anlam ifade etmektedir. Scratch algoritma becerileri ve bilgi-işlemsel düşünme becerilerine katkı sağlamaktadır. Begosso ve Silva (2013) yaptıkları çalışmada 11 – 13 yaş aralığındaki 10 öğrencinin 3 aylık bir çalışma ile Scratch kullanarak algoritma ve programlama öğrenme durumları incelenmiştir. Çalışma sonucu Scratch kullanımının programlama ve algoritma öğretmede kullanılabilir bir araç olduğu sonucuna varılmıştır. Maloney ve arkadaşları (2010) 8 – 16 yaş aralığındaki öğrencilerle yaptıkları çalışmada Scratch programlama ortamında animasyonlu hikayeler oluşturarak programlama ve algoritma geliştirmede olumlu etkiye sahip olduğu bulmuşlardır. Kaučič ve Asič (2011) ilkokul öğrencileri ile yaptıkları çalışmada Scratch görsel programlama ortamının öğrencileri programlama öğretici ve ilgi çekici bir ortam olduğunu belirtmişlerdir. Su ve arkadaşları (2015) 37 Tayvanlı 6. Sınıf öğrencisine haftalık 3 saatten 4 aylık bir süreçte Scratch Programlama isimli bir ders planlayarak çalışma yapmışlardır. Yapılan çalışma sonucu Scratch programlama ortamının programlama öğretimi için yenilikçi ve kullanılabilir bir ortam olduğu sonucuna varmışlardır. Ozoran ve arkadaşları (2012) ise yaptıkları çalışmada üniversite öğrencilerine algoritma eğitimi ve programlamaya başlangıç için Scratch temelli bir eğitim düzenleyip bu becerilerin kazandırılmasında Scratch programının olumlu bir etkisi olduğunu belirtmişlerdir. Federici (2011) mühendislik fakültesinde eğitim gören 40 öğrenci ile yaptıkları çalışmada Scratch programlama aracının öğrencilerin C, Java gibi programlama dillerinin küçük ayrıntıları ile uğraşmak yerine başlangıç için daha iyi bir öğrenme ortamı oluşturduğu sonucuna ulaşmıştır. Bu çalışmada da literatürde yer alan çalışmaları destekler sonuçlar elde edilmiştir. Elde edilen sonuçlara göre Scratch görsel programlama aracının algoritma geliştirme eğitimi için kullanılabilir bir araç olduğu söylenebilir. Literatürde Scratch programlama aracı ile oluşturulan projelerin bilgi-işlemsel düşünme becerilerine etkisinin incelendiği çalışmalar olduğu bilinmektedir (Brennan & Resnick, 2012; León & Robles, 2015). Bu çalışmalardan Brennan ve Resnick yaptıkları çalışmada oluşturdukları online grubun yaptıkları çalışmaları incelemişler ve geliştirilen Scratch projelerinin bilgi-işlemsel düşünmeyi ölçmek için kullanılabilir bir araç olduğunu ve bilgi-işlemsel düşünme becerilerini geliştirdiğini belirtmişlerdir. León ve Robles (2015) ise Scratch web sitesinde yayınlanan rastgele aldıkları projeleri değerlendirmişler ve yalnız Scratch projelerinin bilgi-işlemsel düşünme becerilerini ölçmek için yeterli olmayacağını söylemişlerdir. Bu çalışmada elde edilen sonuçlara göre Scratch programlama aracının bilgi-işlemsel düşünme becerilerini geliştirmek için kullanılabilir bir öğrenme aracı olduğu söylenebilir.

Effect of Scratch on 5th Graders' Algorithm Development and Computational Thinking Skills

Extended Abstract

Introduction

Skills that students are expected to possess have been changing along with the technology that is also transforming. Countries feel the need to make some changes in their educational systems to ensure that students acquire these new skills. The changes undertaken for information technologies and software classes can be regarded as one of these changes. This study conducted to have students acquire skills such as informatics literacy, communicating through information technologies, informing sharing, expressing themselves, research, structuring knowledge, working collaboratively, problem solving, programming and developing authentic products is also compatible with the basic skills that students of the 21st century are expected to have. Providing students with training in programming is known to have an important place to ensure the acquisition of the skills. Sub skills of computational thinking skills- creative thinking, algorithmic thinking, critical thinking, collaborative learning and communication skills- are similar to the skills expected from today's students.

Students regard programming as a difficult task to accomplish. Visual programming tools such as Scratch, Alice and Small Basic were developed to facilitate teaching programming. Among the visual programming tools developed for children, Scratch program is the most preferred one since its web site and interface have Turkish language support and the projects prepared with the help of the program can be shared in the internet as open sources. Individuals can learn programming more easily with the help of these tools. In order to acquire programming skills, individuals need to find different solutions to problems and select the fastest path. Finding the fastest path requires that the individual understands the logic of algorithm very well. Scratch is known as a useful tool that helps students acquire the logic of algorithm and algorithmic thinking skills. There are many studies in literature related to the influence of Scratch in teaching algorithm development and programming. However, the number of studies that investigate the effects of Scratch on computational thinking skills is rather limited. Current study investigated the effects of Scratch programming tool on computational thinking and algorithm development skills.

Method

The study utilized a quasi-experimental design that included pretest-posttest and control group. Experimental group students were taught the basic figures and terms used in algorithms and flow charts. 5th graders who participated in the study were taught information technologies and software for two hours per week. The implementation lasted 6 weeks. After program interface was presented to experimental group students, they were instructed about how to use basic operators such as the variables, value operations, list description, mathematical operators, relational operators and logical operators in the

program. In addition to Scratch program, students were taught with examples how these operations could be done in algorithm and flow charts. Sample projects were used in classes to teach students in more detail how the basic features of programming can be used in the Scratch program. The control group students were taught the basic figures and terms used in algorithm and flow charts with the help of a slide show and they were y-taught how to use these terms and figures in the sample algorithm and flow charts. Sample algorithms from easy to difficult were prepared and studied with students so that they could comprehend the logic of algorithms, arrange missing or faulty algorithms and flow charts and prepare new algorithms and flow charts for a new problem. The study group was composed of 5th graders. 31 students were in the experimental group while 31 students were in the control group. Students were administered Computational Thinking Scale and Algorithm Development Achievement Test as pre and post-tests. Administered Computational Thinking Scale was developed to describe secondary school students' computational thinking skill levels. The required factor and reliability analyses for the scale were undertaken and Cronbach Alpha reliability coefficient was found to be 0.809. A 4-option achievement test with 27 items was developed by the researchers for the study. 7 items whose item discrimination values were found to be under 0,30 were eliminated after the pilot implementation undertake n with 105 students. KR-20 internal consistency coefficient for Algorithm Development Achievement Test was found to be 0.85. SPSS 12.0 program was used for data analysis.

Results

Independent Samples t – test was administered to see whether there was a statistically significant difference between the groups' achievement test results. No significant differences were found based on the analysis conducted on computational thinking and algorithm development achievement test scores. Analysis based on pretest-posttest scores pointed to significant differences in the favor of the experimental group. Analysis showed significant differences between the groups' academic achievement. In this context, it can be argued that Scratch program is a learning tool that can be used to develop algorithm development and computational thinking skills.

Conclusion and Discussion

A student's score difference in experimental group is higher than a student's score in control group score by Computational Thinking Scale pre and post test results. Investigation of datum, experimental group results shows that the difference of score is statistically meaningful. By the way, Using Scratch has a positive effect to improve computational thinking. The examination of AGT pre and post test results, it can be seen that experimental group score differences are higher than control group. In terms of AGBT score differences of two groups, results are statistically meaningful. As a result Scratch contributes improving algorithm and computational thinking skills.

Kaynaklar/References

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

Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From Scratch to "real" programing. *ACM Transactions on Computing Education*, *14*(4), 10-25.

Begosso, L., & Silva, P. (2013, October). *Teaching computer programming: A practical review*. Paper presented at IEEE Frontiers in Education Conference (FIE), Oklahoma City, USA.

Brennan, K., & Resnick, M. (2012, April). *Using artifact-based interviews to study the development of computational thinking in interactive media design.* Paper presented at Annual American Educational Research Association Meeting, Vancouver, BC, Canada.

Büyüköztürk, Ş., Kılıç Çakmak, E., Akgün, Ö., Karadeniz, Ş. ve Demirel, F. (2014). *Bilimsel araştırma yöntemleri*. Ankara: Pegem Akademi.

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

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

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

Federici, S. (2011, October). *A minimal, extensible, drag-and-drop implementation of the C programming language.* Paper presented at Proceedings of the 13th Annual Conference on Information Technology Education, New York, America.

Garner, S. (2009, July). *Learning to program from Scratch.* Paper presented at the 9th IEEE International Conference on Advanced Learning Technologies, Riga, Latvia.

Genç, Z. ve Karakuş, S. (2011, Eylül). *Tasarımla öğrenme: Eğitsel bilgisayar oyunları tasarımında Scratch kullanımı.* 5. Internatiol Computer & Instructional Technologies Symposium'da sunulan bildiri, Fırat Üniversitesi, Elazığ.

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

Günüç, S., Odabaşı, H. ve Kuzu, A. (2013). 21. yüzyıl öğrenci özelliklerinin öğretmen adayları tarafından tanımlanması: Bir Twitter uygulaması. *Eğitimde Kuram ve Uygulama*, *9*(4), 436-455.

International Society for Technology in Education [ISTE]. (2015). *Computational thinking.* Retrieved March 12, 2015 from http://www.iste.org/docs/ct-documents/ctleadershipt-toolkit.pdf?sfvrsn=4.

Karabak, D. ve Güneş, A. (2013). Ortaokul birinci sınıf öğrencileri için yazılım geliştirme alanında müfredat önerisi. *Eğitim ve Öğretim Araştırma Dergisi*, *2*(3), 175-181.

Kaučič, B., & Asič, T. (2011, May). *Improving introductory programming with Stratch?* Paper presented at 34th International Convention Conferance, Opatija, Croatia.

Korkmaz, Ö. (2016). The effects of Scratch-based game activities on students' attitudes, self – efficacy and academic achievement. *International Journal Modern Educationand Computer Science*, *8*(1), 16-23.

Korkmaz, Ö., Çakır, R. ve Özden, M. (2015). Bilgisayarca düşünme beceri düzeyleri ölçeğinin (bdbd) ortaokul düzeyine uyarlanması. *Gazi Eğitim Bilimleri Dergisi*, *1*(2),143-162.

Korkmaz, Ö., Çakır, R., Özden, M., Oluk, A. ve Sarıoğlu, S. (2015). Bireylerin bilgisayarca düşünme becerilerinin farklı değişkenler açısından incelenmesi. *Ondokuz Mayıs Üniversitesi Eğitim Fakültesi De*rgisi, *34*(2), 68-87.

Köse, U. ve Tüfekçi, A. (2015). Algoritma ve akış şeması kavramlarının öğretiminde akıllı bir yazılım sistemi kullanımı. *Pegem Eğitim ve Öğretim Dergisi*, *5*(5), 569-586.

León, J., & Robles, G. (2015, August). *Analyze your Scratch projects with dr. Scratch and assess your computational thinking skills.* Paper presented at 7th International Scratch Conferance, Amsterdam, Netherlands.

Lye, S., & Koh, J. (2014). Review on teaching and learning of computational thinking through programing: What is next for K-12? *Computers in Human Behavior, 41,* 51-61.

Milli Eğitim Bakanlığı [MEB]. (2012). *Ortaokul ve imam hatip ortaokulu bilişim teknolojileri ve yazılım dersi (5, 6, 7 ve 8. sınıflar) öğretim programı.* 15 Ocak 2016 tarihinde http://mufredat.meb.gov.tr/Programlar.aspx adresinden erişildi.

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

Nikou, S., & Economides, A. (2014, April). *Transition in student motivation during a Scratch and an appınventor course.* Paper presented at Proceedings of Global Engineering Education Conference (EDUCON), İstanbul, Turkey.

Ozoran, D., Çağıltay, N., & Topalli, D. (2012, November). *Using Scratch in introduction to programing course for engineering students.* Paper presented at 2nd International Engineering Education Conference (IEEC2012), Atılım Üniversitesi, Antalya.

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

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

Seçer, İ. (2013). *SPSS ve LISREL ile pratik veri analizi.* Ankara: Anı Yayıncılık.

Shin, S., Park, P., & Bae, Y. (2013). The effects of an ınformation-technology gifted program on friendship using Scratch programming language and clutter. *International Journal of Computer and Communication Engineering*, *2*(3), 246-249.

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

Wing, J. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Yükseltürk, E. ve Altıok, S. (2015). Bilişim teknolojileri öğretmen adaylarının bilgisayar programlama öğretimine yönelik görüşleri. *Amasya Üniversitesi Eğitim Fakültesi Dergisi, 4*(1), 50-65.

Yükseltürk, E. ve Altıok, S. (2016). Bilişim teknolojileri öğretmen adaylarının programlama öğretiminde Scratch aracının kullanımına ilişkin algıları. *Mersin Üniversitesi Eğitim Fakültesi Dergisi, 12*(1), 39-52.

Kaynak Gösterme

Oluk, A., Korkmaz, Ö. ve Oluk, H. A. (2018). Scratch'ın 5. sınıf öğrencilerinin algoritma geliştirme ve bilgi-işlemsel düşünme becerilerine etkisi. *Türk Bilgisayar ve Matematik Eğitimi Dergisi, 9*(1), 54-71.

Citation Information

Oluk, A., Korkmaz, Ö., & Oluk, H. A. (2018). Effect of Scratch on 5th graders' algorithm development and computational thinking skills. *Turkish Journal of Computer and Mathematics Education, 9*(1), 54-71.