Problem Çözme Süreci ve Öz-Yeterlik Algısı Üzerinden Programlama Öğretiminin İncelenmesi

Mustafa Serkan Abdüsselam[*], Ebru Turan Güntepe[**], Ümmü Gülsüm Durukan[***]

Makale Geliş Tarihi: 25/06/2020 *Makale Kabul Tarihi: 03/12/2020*

DOI:10.35675/befdergi.758137

Öz

Bu araştırmada programlama öğretimi sürecinde, öğretmen adaylarının bu süreçte izlediği adımların ve yeterliliklerinin belirlenmesi amaçlanmıştır. Araştırma, iç içe karma desen çerçevesinde 24 öğretmen adayı ile yürütülmüştür. Veri toplama aracı olarak araştırmacılar tarafından hazırlanan Programlama Sürecinde Problem Çözme Bilgi Toplama Formu (PPBTF) ile Programlamaya İlişkin Öz yeterlilik Algısı Ölçeği (PÖYAÖ) kullanılmıştır. PPBTF'den elde edilen veriler içerik analizi ile analiz edilirken; PÖYAÖ'den elde edilen veriler için adayların toplam puanları hesaplanmıştır. Bu araştırmada öğretmen adaylarının ifadelerinden ortaya çıkarılan programlama süreci alanyazındaki problem çözme aşamaları ile benzer içeriğe sahip olduğu belirlenmiştir. Öğretmen adayının PPBTF'de programlama süreci için kendilerine verdikleri yeterlik puanlarının, PÖYAÖ'den aldıkları puanlardan daha yüksek olduğu; programlama öğretim sürecindeki bireysel çabanın öğretmen adaylarının yeterlik algılarını doğrudan etkilediği söylenebilir.

Anahtar Kelimeler: *Öz-yeterlik algısı, problem çözme süreci, programlama öğretimi*

Investigation of the Teaching Process of Programming in regards to Problem Solving Process and Perception of Self-Efficacy

Abstract

This study aims to determine the contents that teacher candidates (TCs) follow during the process of programming instruction and their efficiencies. The study was conducted with 24 TCs within the framework of the embedded mixed research. As the data collection tools, the "Knowledge Collection Form of the Problem Solving in Programming Process" (KCFPSP) and the "Computer Programming Self-Efficacy Perception Scale" (CPSEPS) were used. KCFPSP was analyzed by content analysis, the total scores of the TCs were calculated for

[*] Giresun Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü, Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Giresun, Türkiye, mustafa.serkan@giresun.edu.tr, ORCID: 0000-0002-3253-7932

[**] Giresun Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü, Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı, Giresun, Türkiye, ebrutgntp@gmail.com, ORCID: 0000-0002-4858-2180

[***] Giresun Üniversitesi, Eğitim Fakültesi, Matematik ve Fen Bilimleri Eğitimi Bölümü, Fen Bilgisi Eğitimi Anabilim Dalı, u.g.iyibil@gmail.com, ORCID: 0000-0002-9279-2812

Kaynak Gösterme: Abdüsselam, M. S., Turan Güntepe, E., & Durukan, Ü. G. (2021). Problem çözme süreci ve öz-yeterlik algısı üzerinden programlama öğretiminin incelenmesi. *Bayburt Eğitim Fakültesi Dergisi, 16*(31), 149-173. https://doi.org/10.35675/befdergi.758137.

CPSEPS. The programming process revealed from the statements of the TCs had similar content to the problem solving process in the literature. It was found that the self-efficacy scores in KCFPSP were higher than the scores in CPSEPS; so, it can be said that the received education and the individual effort in this teaching process directly affect the TCs' perceptions of self-efficacy.

Keywords: *Perception of self-efficacy, problem solving process, programming instruction*

Giriş

Dünya çapında yazılım geliştirmenin önem kazandığı bilinmektedir. Özellikle Amerika Birleşik Devletleri, Hindistan ve Çin gibi ülkeler yazılım geliştirebilen birey sayısını sürekli arttırma gayreti içindedirler. Bu süreçte de 2018'de dünya çapında 23 milyon olan yazılım geliştirebilen birey sayısının, beş yıl içinde 27,7 milyona ulaşması öngörülmektedir (EDC, 2018). Bu doğrultuda, birçok ülkede genç nesilleri dijital çağa hazırlamak için öğretim programlarını güncelleme ihtiyacı duymaktadır. Bu güncelleme sürecinin temelinde; öğrencilerin mantıksal düşünme, problem çözme ve kodlama becerilerinin geliştirilmesi gibi durumları desteklemeye odaklanılsa da en önemlisi öğrencinin bir bilgisayar bilimcisi gibi düşünebilmesidir (Demir & Seferoğlu, 2017; Yağcı, 2018). Böylece günümüzde öğretim programlarına programlama öğretimi dâhil edilmesiyle kodlama, algoritma vb. terimler ön plana çıkmıştır (Balanskat & Engelhardt, 2015). Ülkemizde ilk kez 2012 yılında "Bilişim Teknolojileri ve Yazılım" dersi kapsamında öğretim programına dâhil edilen "Problem Çözme ve Programlama" adlı ünitede ele alınan içeriklerin öğretilmesiyle; öğrencilere problem çözme, algoritma kurma ve kodlama ile ilgili becerilerin kazandırılması hedeflenmektedir (MEB, 2018). Öğrencilerin programlama ve problem çözme süreçlerini bir bütün olarak görebilmesi için yükseköğretimde de yazılım geliştirme ile ilgili eğitimin proje tabanlı ve problem çözme becerilerine dayanan bir yaklaşımla işlenmesi gerektiği ifade edilmektedir (Demirer & Nurcan, 2015).

Problem Çözme Süreci

Birey, önceden edindiği kavram ve becerileri problemin çözümüne ulaşmak için kullandığı ve yeniden organize ettiği düşünüldüğünde, problem çözme etkinliklerinin etkili bir öğrenme yolu olduğu söylenebilir (Ünsal & Ergin, 2011). Problem çözme sürecini ifade etmek için araştırmacılar tarafından farklı sayıda aşama içeren problem çözme stratejileri ileri sürülmüştür. Bu problem çözme stratejilerinden biri olan Polya (1945)'nın önerdiği dört aşamalı problem çözme süreci (1) Problemi anlama, 2) Bir çözüm planı yapma, 3) Planı uygulama, 4) Geriye bakma, çözümü gözden geçirme) birçok çalışmada yer almaktadır (aktaran Baki & Bell, 1997). Problem çözme süreci, programlama sürecinin önemli bir parçası olmasının yanı sıra (Bagley & Chou, 2007; Han & Kim, 2016; Yağcı, 2018), programlama eğitimi alan öğrencilerin problem çözme becerilerinin de geliştiği bilinmektedir (Sayın & Seferoğlu, 2016). Bireyler günlük yaşantılarında da aynı bir programlama sürecinde olduğu gibi, bir problemin

çözmek ya da gelecekteki bir eyleme dair plan oluşturmak için programın adımlarını kullanmaktadır. Program oluşturulurken, tespit edilen bir problemin çözümüne yönelik geliştirilen algoritmaların yani adımların takibiyle programlama diline ait kodların yazıldığı bilinmektedir (Yaşar, 2014). Bu bağlamda bir problemin çözümünde olduğu gibi programın oluşturulması sürecinde de yine sıralı adımların takip edildiği anlaşılmaktadır. Sıralı adımların yani algoritmaların oluşturulup, takip edilmesindeki temel amaç; süreci kolaylaştırmak ve basitleştirerek doğru çözüme ulaşmaktır. Problem çözme becerilerinin kazanılması ve geliştirilmesi, öğrencilerin programlamayı öğrenmesinde ve uygulamasında önemlidir (Keller & Pausch, 2005; Gomes & Mendes, 2007; Hung, 2008; Gundurao, Manjunath & Nachappa, 2010). Ayrıca problem çözme becerilerine dayandırılan öğrenme ortamlarının öğrencilerin motivasyonlarını arttırdığı (Fukuzawa, Boyd & Cahn, 2017) ve işbirlikçi çalışmayı desteklediği (Calder, 2010; Maloney, Resnick, Rusk, Silverman & Eastmond, 2010göz önüne alındığında problem çözme becerilerinin programlama öğretiminde önemli bir yere sahip olduğu söylenebilir (Hung, 2008).

Öğrencilerin programlamayı öğrenmesini etkileyen bazı faktörler bulunmaktadır. Bu faktörlerin başında programlamaya yeni başlayanların, problem çözme becerisinin yeterli düzeyde olmaması gelmektedir (Gomes & Mendes, 2007). Programlama sürecindeki görevlerin zor ve karmaşık görünmesi de programlama öğrenimindeki engellerden biri olup, bu durum programlama öğreniminde başarısızlığa neden olabilir (Mazman & Altun, 2013; Noone & Mooney, 2018). Bunun yanı sıra öğrencilerin algoritma kurma mantığını anlamaması (Dönmez Usta & Turan-Güntepe, 2019), kodları anlamak yerine ezberlemesi (Yecan, Özçınar & Tanyeli, 2017) ve programlamaya yönelik düşük veya orta seviyede bir tutum içinde olması da süreci olumsuz etkilemektedir (Aşkar & Davenport, 2009).

Öz-Yeterlik Algısı

Programlama öğretiminde dikkat edilmesi ve geliştirilmesi gereken bir başka unsur da öz-yeterliktir (Askar & Davenport, 2009). Öz-yeterlik, öğrencinin belirli bir hedefe ulaşmak veya bir sorunu çözmek için olan motivasyonu (Bandura, 1977) ya da daha önce belirlenen veya gerekli olan eylemleri gerçekleştirme yeteneğine olan inancını (Yang & Cheng, 2009) ifade eder. Programlama öz-yeterliği ise programlama içerikleri için öğrencinin gösterdiği öğrenme seviyesi ve motivasyonudur (Durak, Yılmaz & Yılmaz, 2019). Öz-yeterliğin, programlamaya yönelik tutum kazanma sürecinde yordayıcı olduğu (Gurer, Cetin & Top, 2019) ve öğrencilerin kodlamaya yönelik öz-yeterlik algılarının arttıkça, programlama öğrenimine yönelik olumlu tutum geliştirdikleri (Korkmaz, Şahin, Çakır & Erdoğmuş, 2019) vurgulanmaktadır.

Bir duruma veya bir konuya ait öz-yeterlik inancı yüksek olan bireylerin daha yüksek beklentiye sahip ve herhangi bir zorlukla karşılaştıklarında bu zorlukla baş etmekte daha başarılı olduğu bilinmektedir (Compeau & Higgins, 1995; Akkoyunlu & Kurbanoglu, 2004; Çakıroğlu & Işıksal, 2009). Bu doğrultuda, öz-yeterliği yüksek

öğrencilerin programlamada başarılı olabileceğinden söz etmek mümkündür. Öğrencilerin programlama ile ilgili öz-yeterlik algılarının düşük olması onların en baştan programlamayı zor olarak algılamaları nedeniyle bu derste başarısız olabileceklerini aktaran çalışmalar, ilgili alanyazında yer almaktadır (Askar & Davenport, 2009; Altun & Mazman, 2012; Mazman & Altun, 2013). Genç bireylere programlama öğretiminde, Bilişim Teknolojileri (BT) öğretmenleri önemli bir rol üstlenmektedir. Bu doğrultuda geleceğin öğretmeni olacak adayların programlama ve programlama öğretimine yönelik algı, tutum ve görüşlerini ortaya çıkarmak için yürütülen çalışmaların, ilgili öğretim sürecini daha iyi tasarlayabilmek için önem arz ettiği düşünülmektedir. İlgili alanyazında BT öğretmen adaylarının, bilgisayar programlamaya yönelik öz-yeterlik algıları ve tutumlarına (Yükseltürk & Altiok, 2017), programlamaya yönelik tutumlarına (Özyurt & Özyurt, 2015; Chen, Haduong, Brennan, Sonnert & Sadler, 2019), bilgisayar programlama öğretimine yönelik görüşlerine (Yükseltürk & Altiok, 2015; Çankaya, Durak & Yünkül, 2017) ve bu süreçte kullanılan araçlara ilişkin algılarına (Yükseltürk & Altiok, 2016; Davenport, 2018) yönelik çalışmaların mevcut olduğu görülmektedir. Ayrıca, programlama sürecini etkileyen öz-yeterlik gibi diğer faktörlerinde dikkate alınması gerekli görülmektedir (Ramalingam, LaBelle & Wiedenbeck, 2004; Jegede, 2009). Bu gereklilik öz-yeterlik ve problem çözme süreçlerinin birbiriyle ilişkili olduğu ve bu ilişkilerin etkin kurulduğunda programlama sürecinde başarının sağlanabileceği ifadesi alanyazında vurgulanmaktadır (Sayın & Seferoğlu, 2016). Alanyazında belirtildiği gibi problem çözme becerileri programlama becerilerini geliştirmekte ve programlama becerilerinin gelişimi de programlama özyeterlik algılarının geliştirildiği düşünülmektedir. Bu sebeple programlama süreci, problem çözme süreci ve programlama özyeterlik algılarının birbirini yordama durumlarının ortaya çıkarılması gerektiği düşünülmektedir. Bu bağlamda programlama eğitimi alan öğretmen adaylarının, bu süreçteki problem çözme sürecinin ve özyeterliklerinin belirlenmesi iyi bir programlama öğretimi sürecinde öğreticilere, program geliştiricilere ve araştırmacılara yol gösterici olacaktır. Bu çalışmada programlama öğretimi sürecinde, öğretmen adaylarının izlediği adımların ve yeterliklerinin belirlenmesi amaçlanmış ve bu amaç doğrultusunda programlama sürecinde aşağıdaki sorulara cevap aranmıştır:

- "Bir problemin çözümü için doğru yolu tasarlama sürecinde adayların izledikleri adımlar nelerdir?"
- "Problemin doğru kodlarla ifade edilmesi sürecinde adayların izledikleri adımlar nelerdir?"
- "Problemin çözümü için hata denetimi sürecinde adayların izledikleri adımlar nelerdir?"
- "Adaylarının bu süreç içerisindeki öz-yeterlik algıları nasıldır?"

Yöntem

Araştırma Modeli

Bu araştırmada iç içe karma desen tercih edilmiştir. İç içe karma desen nitel ve nicel araştırma yöntemlerinin bir arada veya art arda kullanılmasına olanak tanımaktadır (Creswell, 2012; Creswell & Plano Clark, 2014). Araştırmacılar öğretmen adaylarının programlama öğretimi sürecinin incelenmesini nitel araştırma desenlerinden biri olan durum çalışması şeklinde yürütürken, elde edilen puanların karşılaştırmasında ise korelasyon analizi ve betimsel istatistikler kullanmıştır. Bu kapsamda nitel durum çalışması şeklinde yürütülen çalışmada hem nitel hem de nicel veriler eş zamanlı toplanmış ve ayrı ayrı analiz edilmiştir.

Çalışma Grubu

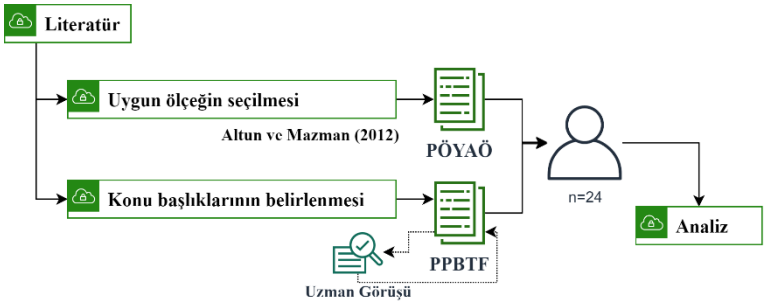
Bu çalışma 2018-2019 eğitim öğretim yılının bahar yarıyılı sonunda Bilgisayar ve Öğretim Teknolojileri Eğitimi (BÖTE) bölümünde öğrenim gören ve Programlama Dilleri I-II derslerini başarı ile tamamlayan 24 öğretmen adayı ile yürütülmüştür. Araştırmada etik kurallar çerçevesinde öğretmen adaylarının isimleri verilmemiştir. Bu öğretmen adayları K1, K2, … K24 şeklinde kodlanmıştır.

Veri Toplama Araçları ve Veri Toplama Süreci

Araştırma kapsamında öğretmen adaylarının programlama öğretimi sürecinin incelenmesine yönelik 10 açık uçlu sorudan oluşan Programlama Sürecinde Problem Çözme Bilgi Toplama Formu (PPBTF) kullanılmıştır. PPBTF'nin geçerliğini sağlamak amacıyla öğretim teknolojileri alanında uzman 3 akademisyenden görüş alınmıştır. Formdaki soruların açıklığı ve anlaşılırlığına yönelik gelen dönütler doğrultusunda düzenlemeler yapılmış ve PPBTF'ye son şekli verilmiştir. PPBTF'de yer alan bu sorulardan 3'ü (S1, S2 ve S3) problemin çözümü için doğru yolu tasarlama sürecini, 2'si (S4 ve S5) doğru kodlarla ifade edilme sürecini açığa çıkarmaya dönük sorulardır. Formdaki sorulardan üçü (S6, S7 ve S8) ise hata denetimi, yazımsal ve mantıksal hatalara yönelik süreci ortaya koymaya yöneliktir. Ayrıca PPBTF'deki bu 8 sorunun, öğretmen adayları tarafından programlama sürecine etkisine/önemine yönelik (1'den 10'a kadar) bir puanlama yapılması istenmiştir. Formda yer alan diğer iki soruda ise adayların programlama sürecindeki yeterliğine ve programlama bilgisine yönelik kendilerini değerlendirdikleri (1'den 10'a kadar) bir puanlama yapmaları beklenmiştir. Veri analizi öğretmen adaylarının yaptıkları bu puanlamalar üzerinden yürütülmüştür.

Araştırma da PPBTF'nin yanı sıra Ramalingam ve Wiedenbeck (1998) tarafından geliştirilen Altun ve Mazman (2012) tarafından Türkçe formu hazırlanan 'Programlamaya İlişkin Öz-yeterlik Algısı Ölçeği' (PÖYAÖ) kullanılmıştır. Bu ölçek 9 madde ve iki faktörden oluşmaktadır. Elde edilen ölçeğin iç tutarlılık katsayısı 0.928 bulunmuş, bu dokuz madde toplam varyansın %80.81'ini açıklamıştır. Araştırma

sürecinde PÖYAÖ ile PPBTF çalışma grubuna eş zamanlı olarak uygulanmış, bu iki veri setinden elde edilen veriler ayrı ayrı analiz edilmiştir. Öğretmen adaylarının veri toplama araçlarını doldurmaları yaklaşık 40 dakika sürmüştür. Araştırma kapsamında kullanılan veri toplama araçlarının uygulama süreci Şekil 1' de yer almaktadır.



Şekil 1. Uygulama süreci.

Verilerin Analizi

Çalışmada kullanılan veri toplama araçları dahilinde elde edilen verilerin analizi sırasıyla aşağıdaki aşamalar çerçevesinde gerçekleştirilmiştir:

- PPBTF'den elde edilen veriler içerik analiz yöntemi ile analiz edilmiştir. Bu form ile öğretmen adaylarının ifadelerinden kodlar belirlenmiş ve bu kodlara yönelik frekans hesaplamaları yapılmıştır. Veri analizinden elde edilen bulgular öğretmen adaylarının ifadelerinden alıntılar ile desteklenmiştir.
- Araştırmada güvenirliği belirlemek adına Miles ve Huberman (1994)'ın uyum yüzdesi formülü ile (Uyum yüzdesi=[Görüş birliği/görüş ayrılığı+Görüş birliği]*100) hesaplanmış, uyum yüzdesi 0.93 bulunmuştur. Bu değerin 0.70'ten yüksek olması verilerin analizinin güvenilir olduğunun göstergesidir.
- PÖYAÖ'den elde edilen veriler kullanılarak adayların toplam puanları hesaplanmıştır. Öğretmen adaylarının ölçekten alabilecekleri toplam puan 9 ile 49 puan aralığında değişmektedir. Ayrıca, korelasyon analizinden önce, PÖYAÖ'den alınan toplam puanların öğretmen adaylarının kendilerine verdikleri puanlar ile arasındaki ilişkinin tespiti için, PÖYAÖ'den alınan toplam puanlar 10 puan üzerinden değerlendirilecek şekilde dönüştürülmüştür. PPBTF'nda yer alan programlama bilgisi ve yeterliği bakımından öğretmen adaylarının kendilerine verdikleri puanlar ile PÖYAÖ'den aldıkları toplam puanlar normal dağılım göstermediği (Grafik 1'de de görüldüğü gibi) için korelasyon analizinde Spearman Rho katsayısı kullanılarak karşılaştırılmıştır.

Bulgular ve Yorum

PPBTF'deki problemin çözümü için doğru yolu tasarlama sürecine odaklanan açık uçlu 3 soruya öğretmen adaylarının verdikleri yanıtlar Tablo 1'de özetlenmiştir.

Tablo 1.
Öğretmen adaylarının bir problemin çözümü için doğru yolu tasarlama sürecine ait yanıtlar

Tema	Kodlar	Katılımcılar	f	Örnek ifadeler
Problemin tanımı	Problemi anlama	K1, K4, K5, K8, K9, K10, K12, K13, K15, K17, K18, K21, K22	13	K4: "Problemi tanımlamak problemi çözmektedir." K1: "Problemin hangi amaca hizmet ettiğini ve çözümü belirlemek adına önemlidir."
	Algoritma kurma	K2,K3,K6,K7,K11, K14,K16,K19, K20, K23,K24	11	K6: "Problemin ne olduğunu bilmeden, sonraki adımlar sağlam atılmaz." K11: "Problemi doğru tanımlamak, çözüm yollarını analiz etmeye yardımcıdır."
Problemin analizi	Algoritmayı doğru kurma	K1, K2, K3, K4, K5, K7, K8, K9, K11, K15, K19, K21, K22, K24	14	K8: "Problemin detaylı incelendiği aşamadır." K3: "Problemi doğru analiz etmek, doğru çözüm için önemlidir."
	Algoritmanın seçimi	K6, K10, K14, K16, K17, K23	6	K23: "Tüm çözüm önerilerinden en kullanılabilir ve uygun yöntemin seçim aşamasıdır."
	Algoritma kurmanın ön adımı	K12, K13, K18, K20	4	K20: "Çözüm için gerekli olan basamakların sıralanışıdır." K13: "Probleme yönelik detaylı inceleme yapıp, algoritma kurmayı kolay hale getiren aşamadır."
Algoritmanın kurulması	Problemin doğru çözümü	K1, K4, K5, K6, K7, K8, K9, K12, K13, K14, K15, K16, K17, K18, K19, K20, K21, K23, K24	19	K21: "Problemi nasıl çözeceğimize ilişkin algoritma kurmak, problemin çözümünü daha anlaşılır kılar." K24: "Doğru analiz sonrası yapılması gereken adımların kurulduğu kısımdır."
	Kodlama sürecinde önemsiz	K3, K10	2	K3: "Algoritma kurulmadan da kodlamayı doğru yapmak mümkündür."
	Kodlamanın ön adımı	K2, K11, K22	3	K11: "Kolay ve hızlı bir şekilde kodlamanın yapılması için gereklidir."

Öğretmen adaylarının bir problemin çözümü için doğru yolu tasarlama sürecine ait yanıtların problemin tanımı, problemin analizi ve algoritmanın kurulması olmak üzere üç tema altında toplandığı görülmektedir (Tablo 1). Problemin tanımı teması altında öğretmen adaylarının birçoğu (f=13) problemi anlama sürecinin önemine değinmişlerdir. Problem analizi teması altında ise öğretmen adayları (f=14) algoritmayı doğru kurmanın önemine değinmişlerdir. Algoritmanın kurulması teması için öğretmen adaylarının (f=19) verdikleri yanıtların daha çok problemin doğru

çözümü teması altında toplandığı görülmüştür. Öğretmen adaylarının bir problemin çözümü için doğru yolu tasarlama sürecinde problemi doğru çözümünün en önemli durum olduğu, bu duruma ilgili olarak ise problemi anlama ve algoritmayı doğru kurma ile öncelikli ilişkilendirebileceği söylenilebilir. PPBTF'deki problemin doğru kodlarla ifade edilme sürecini açığa çıkarmayı amaçlayan açık uçlu 2 soruya öğretmen adaylarının verdikleri yanıtlar ise Tablo 2'de özetlenmiştir.

Tablo 2.
Öğretmen adaylarının bir problemin doğru kodlarla ifade edilme sürecine ait yanıtları

Tema	Kodlar	Katılımcılar	f	Örnek ifadeler
Kodların yazılması	Kodlama sürecinin temeli	K2, K3, K5, K8, K9, K10, K11, K12, K22, K24	10	K5: "Kodların yazılmaya başlandığı aşamadır." K3: "Programı çalıştırmak ve denemek için olmazsa olmaz adımdır."
	Doğru kodlama	K6, K7, K13, K14, K15, K19, K21, K23	8	K13: "Doğru kodların yazılması yazımsal hataların önüne geçer." K23: "Mantıksal olarak doğru kurulan kodlar, doğru çözüm ve kullanışlılık için önemlidir."
	Problemi doğru çözme	K1, K4, K16, K17, K18, K20	6	K16: "Problemin tanımlanıp, analiz edildiğinin ve doğru algoritmanın kurulduğunun göstergesidir." K4: "Probleme doğru çözüm bulsak bile çözüme ulaşmamış oluruz."
Programın çalıştırılması	Hataları tespit etme	K1, K3, K4, K7, K11, K12, K13, K14, K16, K18, K19	11	K4: "Hata denetimi yapılmadan çalıştırılan programın doğru bir sonuç vermesi beklenemez." K7: "Yapılan işlemlerin doğruluğunun test edildiği ve hataların gözüktüğü aşamadır."
	Geri bildirim alma	K2, K6, K8, K9, K15, K20, K21, K24	8	K6: "Yazılımcı için bir dönüttür." K15: "Yazılan kodların çalıştırıldığı bölümdür."
	Çözümün sınanması	K5, K10, K17, K22, K23	5	K5: "Problemin ortadan kalktığı aşamadır." K22: "Süreçteki tüm emeklerinin karşılığıdır."

Öğretmen adaylarının bir problemin doğru kodlarla ifade edilme sürecine ait yanıtları, Tablo 2'de görüldüğü gibi kodların yazılması ve programın çalıştırılması olmak üzere iki tema altında toplanmıştır. Kodların yazılması teması altında öğretmen adaylarının bir kısmı (f=10) kodların yazılmasının, kodlama sürecinin temeli olduğuna değinmişlerdir. Programın çalıştırılması teması altında ise, öğretmen adayları (f=11) bu aşamanın hataların tespitinde önemli olduğunu belirtmişlerdir. Öğretmen adaylarının bir problemin doğru kodlarla ifade edilme sürecinde kodların yazılması bu sürecin ilk basamağı olduğu söylenebilir. Ayrıca doğru kodların

yazılması kadar programda hataları tespit etme durumu da adaylar için önceliklidir. Dolayısıyla bu süreçte kodların yazılması kadar programda olabilecek hataları tespit etmek ve ilgili düzeltmeleri yapmakta önem taşımaktadır. PPBTF'deki problem çözme sürecinin hata denetimini irdeleyen açık uçlu 3 soruya öğretmen adaylarının verdikleri yanıtlar ise Tablo 3'te özetlenmiştir.

Tablo 3.
Öğretmen adaylarının problem çözme sürecinde hata denetimine ait yanıtları

Tema	Kodlar	Katılımcılar	f	Örnek ifadeler
Hata denetimi	Hataları tespit etme	K2, K3, K4, K5, K7, K9, K11, K12, K13, K15, K18, K21, K22, K23, K24	15	K9: "Olası problemlerin çözümü için gereklidir." K12: "Yazılan kodlardaki hataların nerede olduğunu gösterir."
	Problemi doğru kavrama	K1,K6,K8,K10,K14, K16,K17,K19,K20	9	K1: "Problemin doğru kavranılmadığının ya da mantığının kurulamadığının göstergesidir." K17: "Yapılan işlemlerin doğruluğunu test etmek için gereken adımdır."
Yazımsal hatalar	Çözüme yardımcı olma	K1, K2, K3, K4, K5, K6, K7, K8, K9, K10, K11, K12, K13, K14, K15, K17, K20, K21, K23, K24	20	K8: "Yazılan kodlardaki hataların nerede olduğunu gösterir ve dönüt sonucu düzeltme yapılır." K23: "Çözümü kolaylaştıran aşamadır."
	Küçük hatalar	K16, K18, K19, K22	4	K16: "Ufak hatalar olsa da programın çalışması için önemlidir." K19: "Ufak hatalardır."
Mantıksal hatalar	Temel hatalar	K1, K5, K7, K11, K13, K14, K15, K20, K21, K22, K23, K24	12	K22: "Mantıksal hatalar verimliliği azaltır, programın kullanılabilirliğini sınırlar." K23: "Mantığı kavranılmadan yapılan çözümler, en büyük hatalardır. "
	Süreci doğru yapılandırma	K2, K3, K4, K6, K8, K9, K10, K12, K16, K17, K18, K19	12	K8: "Problemin yanlış analiz edildiği ve algoritmanın yanlış kurulduğunun göstergesidir." K9: "Çözümü kısa olan bir problemi uzun yoldan çözmek size zaman kaybettirir."

Tablo 3'te görüldüğü gibi problem çözme sürecinde hata denetimi, hata denetimi, yazımsal ve mantıksal hatalar olmak üzere üç tema altında toplanmıştır. Öğretmen adaylarının hata denetimine ait teması altında, hataları tespit etme (f=15) ve problemi doğru kavrama (f=9) kodlarına yer verilmiştir. Problem çözme sürecinde hata denetimi sürecine yönelik adaylar tarafından ortaya koyulan bir başka kod ise yazımsal hatalardır. Öğretmen adaylarının birçoğu (f=20) yazımsal hataların problemin çözüme yardımcı olduğuna değinmişlerdir. Ayrıca mantıksal hataların, problem çözme sürecindeki en temel hatalar (f=12) olduğunu vurgulamıştır. Adaylar problem çözme süreci doğru yapılandırılmadığında, mantıksal hataların oluşacağını belirtmişlerdir. Program mantıksal olarak doğru kurulsa bile süreçte oluşacak

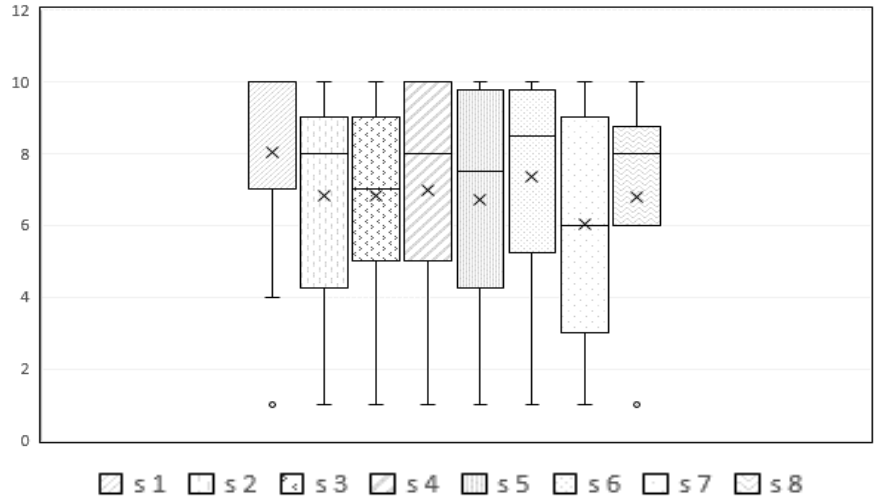
yazılımsal hatalar programın çalışmasını engelleyebilir. Bu bağlamda öğretmen adaylarının problem çözme sürecinde hata denetiminde hataları tespit etme durumu, yapılan işlemlerin doğruluğunu kontrol etmek ve problemi çözmek için yardımcı olabilir. Öğretmen adaylarına kendilerine PPBTF'nda yöneltilen sekiz sorunun program oluşturma sürecine etkisi/önemi konusunda 1'den 10'a kadar yapmış oldukları puanlamalar, Tablo 4'deki sıra frekansları matrisi ile sunulmuştur.

Tablo 4.
Öğretmen adaylarının program oluşturma sürecine ait aşamaların önemi ile ilgili sıra frekansları matrisi

Puanlama	Bir Problemin Çözümü İçin Doğru Yolu Tasarlama Süreci			Bir Problemin Doğru Kodlarla İfade Edilme Süreci		Problem Çözme Sürecinde Hata Denetimi		
	f_{S1}	f_{S2}	f_{S3}	f_{S4}	f_{S5}	f_{S6}	f_{S7}	f_{S8}
1	2	1	1	2	3	2	3	4
2	-	1	1	1	-	1	1	-
3	-	2	1	1	1	-	2	-
4	1	2	2	1	2	-	1	-
5	2	3	2	3	2	3	4	-
6	-	1	3	1	1	1	1	5
7	3	1	3	2	3	3	2	2
8	1	3	3	3	4	2	3	7
9	2	5	3	3	2	6	3	1
10	13	5	5	7	6	6	4	5
Toplam	24	24	24	24	24	24	24	24

Tablo 4'teki matriste 24 öğretmen adayının program oluşturma sürecindeki aşamaları sıralamalarından oluşan sıra değerleri yer almaktadır. PPBTF'de yöneltilen sekiz sorunun program oluşturma sürecine etkisi ve önemi konusunda 1'den 10'a kadar yapmış oldukları puanlamalar incelendiğinde; bir problemin çözümü için doğru yolu tasarlama süreci aşaması altındaki birinci soruya 13, ikinci soruya 5 ve üçüncü soruya 5 öğretmen adayının 10 puan verdiği görülmektedir. Bir problemin doğru kodlarla ifade edilme süreci aşaması altındaki dördüncü soruya 7 ve beşinci soruya 6 öğretmen adayının 10 puan verdiği görülürken; problem çözme sürecinde hata denetimi aşaması altındaki altıncı soruya 6, yedinci soruya 4 ve sekizinci soruya 5 öğretmen adayının 10 puan verdiği belirlenmiştir. Bu puanlamanın aksine bir problemin çözümü için doğru yolu tasarlama süreci aşaması altındaki sorulara 1-2 öğretmen adayının, bir problemin doğru kodlarla ifade edilme süreci aşaması altındaki sorulara 2-3 öğretmen adayının ve problem çözme sürecinde hata denetimi aşaması altındaki sorulara 2-4 öğretmen adayının 1 puan verdiği görülmüştür. Tablo 4 incelendiğinde öğretmen adaylarının PPBTF'de yer alan sorular kapsamında yapmış oldukları puanlamaların çoğunlukla 5 puan ve üzerinde olduğu tespit edilmiştir. Bu durumda programlama sürecinde her aşamanın bir öneme sahip olduğunu

düşündüklerine işaret etmektedir. Öğretmen adayları PPBTF'nda yöneltilen sekiz sorunun programlama sürecine etkisine ve önemine yönelik bir yapmış oldukları puanlamaların yığılımı Grafik 1'de gösterilmektedir.



Grafik 1. Öğretmen adaylarının program oluşturma sürecine ait aşamaların etkisi ve önemine yönelik yaptıkları puanlamaların yığılımı

Öğretmen adaylarının program oluşturma sürecine ait aşamaların etkisi ve önemine yönelik yaptıkları puanlamaların yığılımı Grafik 1'de verilmiştir. Öğretmen adaylarının problemin tanımı olarak temalandırılan birinci soruya verdikleri puanların ortalaması 8'dir. Bu ortalama puan dikkate alındığında öğretmen adaylarının 15'i (%62,5'i) ortalama puanın üzerinde ve öğretmen adaylarının 8'i (%33,3'ü) ortalama puanın altında bir puanlama yaptıkları görülmüştür. Problemin analizi olarak temalandırılan ikinci soruya ait puanlamaların ortalaması 6,8'dir. Bu soru için öğretmen adaylarının yaptıkları puanlamaların %58,3'ü (f=14) ortalama puanın üzerinde ve %41,7'si (f=10) ortalama puanın altındadır. PPBTF'de yöneltilen üçüncü soruya yönelik yanıtlar algoritmanın kurulması şeklinde temalandırılmış ve bu soruya yönelik öğretmen adaylarının yaptıkları puanlamaları ortalaması 6,8 olarak hesaplanmıştır. Hesaplanan bu ortalama puana göre öğretmen adaylarının 14'ü (%58,3'ü) ortalama puanın üzerinde ve öğretmen adaylarının 10'u (%41,7'si) ortalama puanın altında bir puanlama yaptıkları görülmüştür. PPBTF'nda kodlamanın yazılması şeklinde temalandırılan dördüncü soruya yönelik öğretmen adaylarının yaptıkları puanlamaların ortalaması 7'dir. Bu soru için öğretmen adaylarının 13'ü (%54,2'si) ortalama puanın üzerinde ve 9'u (%37,5'i) ortalama puanın altında bir puanlama yaptıkları görülmüştür. Öğretmen adaylarının beşinci soruya verdikleri yanıtlar programın çalıştırılması teması altında toplanmış ve bu soruya yönelik yapılan puanlamaların ortalaması 6,7 olarak hesaplanmıştır. Hesaplanan ortalama puana göre öğretmen adaylarının yapmış oldukları puanlamalar incelendiğinde, bu

puanlamaların 15'i (%62,5'i) ortalama puanın üzerinde ve 9'u (%37,5'i) ortalama puanın altındadır. PPBTF'de yer alan altıncı soruya öğretmen adaylarının cevapları hata denetimi teması altında toplanmıştır. Bu soruya öğretmen adaylarının verdikleri puanlamaların ortalaması 7,3'tür. Öğretmen adaylarının yaptıkları puanlamaların %58,3'ü (f=14) ortalama puanın üzerinde ve %41,7'si (f=10) ortalama puanın altında yer almıştır. Yedinci soruda öğretmen adaylarının verdikleri puanların ortalaması 6,1 olarak hesaplanmıştır. Öğretmen adaylarının verdikleri cevapların yazımsal hatalar olarak temalandırıldığı bu soruda, öğretmen adaylarının 12'si (%50'si) ortalama puanın üzerinde ve 12'si (%50'si) ortalama puanın altında puanlama yapmışlardır. PPBTF'de yer alan son soruda (mantıksal hatalar teması) öğretmen adaylarının yapmış oldukları puanlamaların ortalaması 6,8'dir. Bu soru için öğretmen adaylarının yaptıkları puanlamalar incelendiğinde, öğretmen adaylarının 15'i (%62,5'i) ortalama puanın üzerinde ve 9'u (%37,5'i) ortalama puanın altında puanlama yapmışlardır. Grafik 1'den elde edilen bulgular öğretmen adaylarının yaptıkları puanlamada en yüksek ortalama puana sahip problemi tanımlama sürecinin programlama sürecinde önemli bir paya sahip olduğunu düşündüklerini göstermektedir. Programlama sürecinde öğretmen adaylarının ortalama puana göre önemli gördükleri bir başka durum, hata denetimidir. Grafik 1'deki ortalama puanlar incelendiğinde, öğretmen adaylarının problemi iyi bir şekilde tanımlayıp bir hata ile karşılaşmaları durumunda hata denetimi sürecini iyi bir şekilde yönlendirebilirlerse programlama sürecinde başarılı olabilecekleri düşüncesine sahip oldukları söylenebilir. Öğretmen adaylarının programlama sürecine dair sorulara verdikleri cevapların ardından, PPBTF'de yer alan "Programlama sürecinde kendiniz yeterli görüyor musunuz? Açıklayınız" sorusunu yanıtlamaları istenmiştir. Adayların bu soruya verdikleri yanıtlar Tablo 5'te özetlenmiştir.

Tablo 5.
Öğretmen adaylarının programlama sürecinde yeterlik algıları

Tema	Kod	Katılımcı	f	Örnek ifadeler
Yeterli	Alınan eğitim	K6, K8, K17	3	K8: "Liseden de programlama konusunda biraz bilgim olduğu için kolayca kodları yazabiliyorum"
	Bireysel çaba	K1, K7, K9	3	K1: "Bu alana (derse) karşı bir ilgim var, ve boş zamanlarımda dersim olmasa bile uğraşıyorum." K7: "Konuyu serbest şekilde uygulamayı denediğimde araştırarak takıldığım noktalara çözüm üretebilirim"
Kısmen yeterli	Eksiklerinin fark etmeme	K19, K21	2	K19: "Kendimi programlama konusunda çok yeterli görmüyorum. Ama yetersiz ve bilgisiz olduğumu da düşünmüyorum."
	Eksikliklerini fark etme	K3, K20	2	K3: "Bazı konularda gayet yeterli görsem de bazı konularda (while, diziler) yetersiz görüyorum."

				K20: "Sayısal programlama için yeterli görüyorum fakat sözel programlama için aynı durum geçerli değil."
Yetersiz	Alınan eğitim	K2, K4, K5, K10, K11, K12, K13, K18, K24	9	K4: "… Çünkü bilmediğim o kadar çok kod ve program var ki bu benim motivasyonumu bozuyor. …" K5: "Programlama bilgisi temelimi lise de görmeme rağmen tam oluşturamadığım için üzerine ne kadar yeni bilgiyi eklesem de birşeyler hep eksik kalıyor." K12: "Değişkenleri tam olarak kullanamıyorum, komutların tam olarak hangi sıralamada olacağını anlamıyorum" K13: "Genellikle algoritmaları kurabiliyorum. …. Fakat programlama C# bilgim yetersiz kalıyor. Tasarladığım algoritmayı koda dökemiyorum"
	Bireysel çaba	K14, K15, K16, K22, K23	5	K14: "Programlamanın temelini oluşturamadığımı düşünüyorum. Karmaşık ve zor geliyor. Çalışmama rağmen kavrayamadığım kısımlar fazla. İşleyiş mantığını kavrayamıyorum." K15: "Yeterince iyi olduğumu düşünmüyorum gerekli zamanı ayırıp yeterince tekrar ve çalışma yapmıyorum." K22: "Sınavlardan aldığım notlar sebebiyle kendimi yetersiz görüyorum" K23: "… uğraş versem de mantığını kavrayamadığımı düşünüyorum. Anlayamadığım için mantıklı yolla değil ezberle yapmaya çalışıyorum…"

Programlama sürecinde kendilerini yeterli hissetmeleri konusunda altı öğretmen adayı kendini yeterli, dört öğretmen adayı kendini kısmen yeterli ve on dört öğretmen adayı ise kendiniz yetersiz hissettiklerini ifade ettikleri Tablo 5'te görülmektedir. Kendini programlama sürecinde yeterli hisseden öğretmen adaylarının ifadeleri alınan eğitim ve bireysel çaba kodlarının altında toplanmıştır. Alınan eğitim kodu altında, K8 kodlu adayın ifadesindeki gibi, yükseköğretime yerleşmeden önce aldıkları eğitimin yeterli olduğunu ifade eden adaylar bulunmaktadır. Bireysel çaba kodu altındaki ifadelerde ise (K1 ve K7 gibi) öğretmen adayları programlama ile ilgili temellerinin olduğunu ve programlama dersine çalıştıklarını bu sebeple de kendilerini bu konuda yeterli gördüklerini belirtmektedir. Programlama sürecinde kendini kısmen yeterli gören öğretmen adaylarının çoğunluğu sürece dair kendi eksikliklerinin farkında olmadıklarını, K19'un ifadesinde de belirttiği gibi, ifade etmişlerdir. Kendini bu süreçte kısmen yeterli hisseden öğretmen adaylarının bir kısmı ise K3 ve K20 kodlu adayların ifadelerindeki gibi kendi eksik yönlerinin farkında oldukları görülmüştür. Öğretmen adaylarının kendilerini programlama konusunda kısmen yeterli hissetmeleri ve kendilerini bu süreçte eksik gördükleri noktaları kısmen ifade

edebilmeleri, öğretmen adaylarının programlamaya yönelik alan bilgisi konusunda zorluklar yaşadıklarını düşündürebilir. Programlama sürecinde kendini yetersiz hisseden öğretmen adayları ifadeleri de alınan eğitim ve bireysel çaba kodlarının altında toplanmıştır. Alınan eğitim kodu altındaki ifadelerde öğretmen adayları durumlarını programlamaya dair yeterli bilgiye sahip olmamaları (K4 ve K5 kodlu adayların ifadesi) ve tasarlanan algoritmanın koda dökülememesi (K12 ve K13 kodlu adayların ifadesi) şeklinde açıklamışlardır. Bireysel çaba kodu altında ise, K14 ve K23 kodlu adayların ifadelerindeki gibi programlamanın mantığını kavrayamamalarından ve K15 ve K22 kodlu adayların ifadelerindeki gibi yeterince alıştırma yapmamalarından kendilerini programlama konusunda yetersiz hissettiklerini açıklamışlardır. Bu durum, K22 kodlu adayın ifadesinde de görüldüğü gibi adayların akademik olarak başarılı olmadıkları için kendilerini yetersiz olarak gördüklerini de gösterebilir. Tablo 5'den ulaşılan bulgular doğrultusunda öğretmen adaylarının programlama sürecinde yeterlik algıları, alınan eğitimle ve bireysel çaba ile ilişkilendirilebilir. Ayrıca yeterlilik algıları açısından öğrencilerin aldıkları eğitimi, bireysel çabalarına karşın daha çok ön planda olduğu söylenebilir.

PPBTF'de yer alan bir soruda adaylardan programlama bilgisi ve yeterliği konusunda kendilerine bir puan vermeleri istenmiş ve bu puan Tablo 6'da yeterlik puanı olarak verilmiştir. Bununla birlikte, öğretmen adaylarının PÖYAÖ'nden aldıkları toplam puan ve dönüştürülmüş toplam puan dağılımı da Tablo 6'da yer almaktadır.

Tablo 6.
Öğretmen adaylarının programlama sürecinin aşamalarına dair yeterlik algıları için yapmış oldukları puanlamalar

	Yeterlik Puanı	PÖYAÖ toplam puanı	PÖYAÖ dönüştürülmüş toplam puanı
K1	4	16	3,3
K2	1	47	9,6
K3	6	26	5,3
K4	7	24	4,9
K5	5	28	5,7
K6	6	22	4,5
K7	6	32	6,5
K8	6	29	5,9
K9	5	36	7,3
K10	3	43	8,8
K11	7	27	5,5
K12	3	44	9,0
K13	5	33	6,7
K14	3	33	6,7
K15	6	36	7,3
K16	6	22	4,5

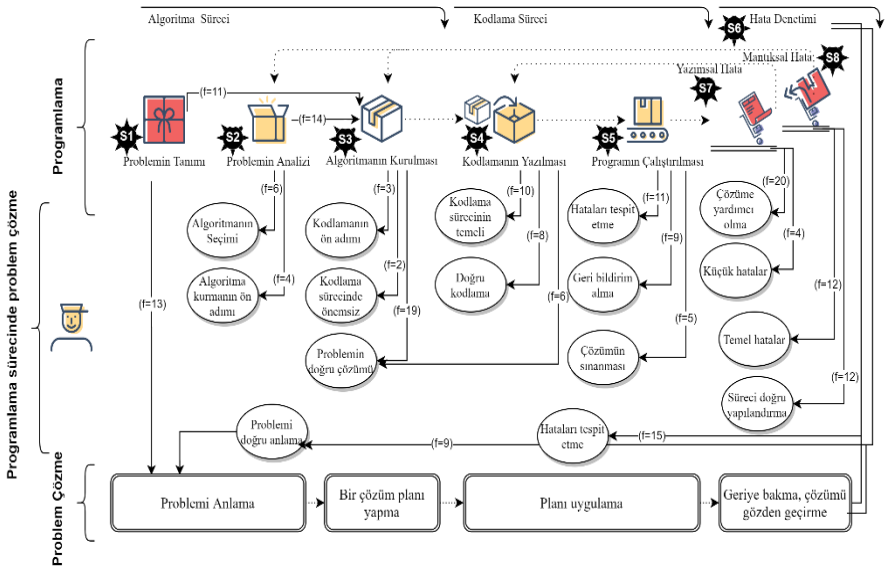
K17	7	32	6,5
K18	6	27	5,5
K19	6	27	5,5
K20	6	24	4,9
K21	7	39	8,0
K22	5	33	6,7
K23	2	48	9,8
K24	6	38	7,8
Ort. Puan	5,2	31,9	6,5

Öğretmen adaylarının programlama bilgisi ve yeterliği bakımından kendilerine verdikleri puanlar 1 ile 7 aralığında değişim göstermektedir. PÖYAÖ'nden öğretmen adaylarının aldıkları toplam puanlar 16 ile 48 aralığında dağılım gösterirken; PÖYAÖ dönüştürülmüş puanları ise 3,3 ile 9,8 aralığında dağılım göstermektedir. Öğretmen adaylarının programlama bilgisi ve yeterliliği ile ilgili kendilerine verdikleri puanların ortalaması 5,2 iken, PÖYAÖ dönüştürülmüş toplam puanlarının ortalaması 6,5 olduğu belirlenmiştir. Öğretmen adaylarının programlama bilgisi ve yeterliği bakımından kendilerine verdikleri yeterlik puanları ile PÖYAÖ dönüştürülmüş toplam puanlarının ortalama puanları birbirine yakın değerlere sahip olduğu görülmüştür. Bununla birlikte, öğretmen adayı bazında bakıldığında bu puanlar arasında birbirinden farklı puanlamalar da bulunmaktadır. Bu noktada, öğretmen adaylarının programlama bilgisi ve yeterliği bakımından kendilerine verdikleri yeterlik puanları ile PÖYAÖ dönüştürülmüş toplam puanları arasında bir ilişki olup olmadığı korelasyon analizi ile test edilmiştir. Bu istatistiksel analize ait sonuçlar Tablo 7'de sunulmuştur.

Tablo 7.
Öğretmen adaylarının PÖYAÖ dönüştürülmüş toplam puanları ile yeterlik puanları arasındaki ilişki

		PÖYAÖ dönüştürülmüş toplam puanı
Yeterlik puanı	Korelasyon Katsayısı	-,475
	P	,019
	N	24

Öğretmen adaylarının kendilerine verdikleri programlama yeterliği puanları ile PÖYAÖ dönüştürülmüş toplam puanları arasında zayıf düzeyde, negatif yönde ve istatistiksel olarak anlamlı bir ilişki bulunmaktadır ($r= -,475$, $p<.05$). Problem çözme süreci, programlama süreci ve öğretmen adaylarının ifadelerinden ortaya çıkarılan programlama süreci Şekil 2'de sunulmuştur.



Şekil 2. Programlama sürecinde problem çözme.

Öğretmen adayları program oluşturma sürecinde öncelikle, problemin çözümü için doğru yolu tasarlamak gerektiğine değinmiştir. Bu kapsamda problemi tanımlamak, problemi analiz etmek ve algoritmaları kurmak, ardından algoritmaları doğru kodlarla ifade etmek için kodlamanın yazılması ve programın çalıştırılması gerekmektedir. Son olarak hata denetimi yani süreçte yazımsal ve mantıksal hataların olma durumlarında hatanın kaynağına erişerek çözüm üretmek ve düzeltmek gereklidir. Program oluşturma sürecinde, problemin kendisini anlamak, çözümü için doğru yöntemi uygulamak ve daha sonra problemi nasıl çözeceklerini bilmeleri ile uygulamalar şekillenmelidir. Bu noktada öğretmen adaylarının ifadelerinden ortaya çıkarılan program oluşturma süreci alanyazında Polya'nın (1945, aktaran Baki & Bell, 1997) önerdiği problem çözme aşamaları (Problemi anlama, Bir çözüm planı yapma, Planı uygulama, Geriye bakma, çözümü gözden geçirme) ile paralellik gösterdiği Şekil 2'de görülmektedir.

Tartışma, Sonuç ve Öneriler

Yapılan çalışma sonucunda problem çözme sürecinin aşamalarının, benzer şekilde programlama sürecinde de takip edildiği belirlenmiştir. Bununla birlikte, bu çalışmada öğretmen adaylarının ifadelerinden ortaya çıkarılan programlama süreci alanyazında Polya'nın (1945, aktaran Baki & Bell, 1997) önerdiği problem çözme aşamaları ile benzer içerikli aşamalar içerdiği görülmektedir. Verilen bir duruma uygun bir program oluşturabilmek için, verilen durum bir problem durumu gibi algılanıp çözüm için aşama aşama çalışılmalıdır (Akpınar & Altun, 2014). Bu noktada, programlama sürecinde bir probleme çözüm bulma etkinliği sırasında problemi tanımlamak, alt

parçalara ayırmak ve temel bir çözüm oluşturmak gerekmektedir (Saeli, Perrenet, Jochems & Zwaneveld, 2011). Dolayısıyla programlama sürecinin problem çözme becerilerini geliştirmeye dönük katkılarından bahsedilebilir (Mulder, 2002; Dasso vd., 2005). Bu durumda programlama öğretiminde problem çözme sürecinin önemli (Bagley & Chou, 2007; Yağcı, 2018) ve yaşanan sorunların birçoğunun problem çözme süreciyle yakından ilişkili (Kelleher & Pausch, 2005) olduğunu göstermektedir.

Problem çözme sürecinin ilk iki basamağı olan problemi anlama ve bir çözüm planı yapma sürecinin, algoritma süreci ile ilişkili olduğu görülmüştür. Algoritma sürecinde ise problemi anlama, problemi doğru çözme ve algoritma kurma gibi kodlar öne çıkmaktadır. O halde problemi tanımlanma süreci iyi bir şekilde yapılandırılırsa programlama öğretimi ve öğreniminde kolaylıklar sağlanacaktır. Ayrıca öğrenenler programlamadaki algoritma kurma sürecini doğru yapılandırabilecek ve mevcut programın problemini kolaylıkla çözebilecektir. Ancak öğrencilerin genellikle algoritma oluşturma sürecinde sorun yaşadığı ve kodları anlamak yerine ezberlediği bilinmektedir (Yecan, Özçınar & Tanyeli, 2017). Bu bağlamda problemi tanımlama sürecine daha çok zaman ayırarak doğru bir algoritma kurmak ve etkili bir programlama öğretimi yürütmek mümkün olabilir.

Problem çözme sürecinin üçüncü adımı olan planı uygulama sürecinin, programlamada kodlamanın yazılması ve programın çalıştırılması süreciyle ilişkili olduğu belirlenmiştir. Bu doğrultuda doğru kodların yazılması için kodlama sürecinde öğrencilerin daha aktif rol alacakları öğrenme ortamları hazırlanmalı ve süreç etkileşimli bir şekilde yönetilmelidir. Benzer şekilde, Hawi (2010) programlama öğretiminde öğrenenlerin aktif rol almaları gerektiğini vurgulamaktadır. Ayrıca programlama sürecinde öğrencilerin akranlarıyla etkileşim kurarak, problem çözme becerilerinin geliştirebildikleri de bilinmektedir (Calder, 2010; Maloney, Resnick, Rusk, Silverman & Eastmond, 2010). Bu durum, öğretmen adaylarının birbirleriyle etkileşim kurarak, ilk elden süreci deneyimlemesine fırsat sunan öğrenme ortamlarının kodlama sürecini olumlu etkileyeceği şeklinde yorumlanabilir.

Problem çözme sürecinin son adımı olan geriye bakma ve gözden geçirme, programlama sürecinin son adımı hata denetimi ile ilişkilidir. Bu aşamada, kodlar yazıldıktan sonra programın çalıştırılarak mevcut hataların belirlenmesine odaklanmak gerekmektedir. Öğretmen adayları karşılaştıkları hataları küçük ya da temel hatalar olarak iki gruba ayırmıştır. Küçük hatalar kodlama sırasında yapılan yazımsal hatalardır. Bu hataların problemin çözümünde yardımcı olabileceği ifade edilmiştir. Temel hatalar ise, problemin analizi çerçevesinde iyi yapılandırılmamış ya da yanlış kurgulanmış algoritmadan kaynaklanmaktadır. Bu durum probleme ait çözümün doğru yapılandırılamadığını ve problemin tam olarak anlaşılamadığını ortaya koymaktadır. Bu aşamada programlamayı öğrenen bireylerin, programı çalıştırdıktan sonra karşılaştıkları geri bildirimleri dikkatle izlemeleri gereklidir. Böylece programlamayı öğrenen bireylerin yaptıkları hatalardan öğrenmeleri

mümkün olabilir. Alanyazında da programlama öğretimi sürecinde öğrencilerin yaptığı hatalardan veya akranlarından öğrendiği (Law, Lee& Yu, 2010; Garner, 2007) ve bu süreçte onlara verilecek geri bildirim veya dönütlerin programlama öğretiminde oldukça etkili olduğuna değinilmiştir (Kordaki, 2010; Qian & Lehman, 2019).

Öğretmen adaylarının PPBTF'nda yöneltilen sekiz sorunun programlama sürecine etkisine yönelik bir yapmış oldukları puanlamalar incelendiğinde, ortalama puanların 6 ile 8 aralığında değiştiği görülmektedir. Bu durum programlama sürecindeki her aşamanın öğretmen adayları tarafından etkili ve önemli görüldüğü şeklinde yorumlanabilir. Grafik 1'de de görüldüğü gibi, program oluşturma sürecindeki her bir aşamayı değerlendiren öğrencilerin 'problemin tanımı' olan birinci kısım için oldukça yüksek puanlar verdikleri görülmüştür. Bu durum, ilgili puanların ortalamasının ve yığılımının diğer kısımlara göre daha yüksek değere sahip olduğundan kaynaklanabilir. Benzer şekilde, alanyazında da bir problemin çözümünün en önemli adımının problemi tanımlamak olduğuna değinilmiştir (Çoşğun & Çoşğun, 2018; Vatansever, 2018). Bu nedenle, bir program yazılmadan önce çözülecek problemin iyice irdelenmesinin, programlama öğretiminde oldukça önemli olduğu söylenebilir.

PPBTF'de öğretmen adaylarının önemli bir kısmının programlama süreci için kendilerine verdikleri yeterlik puanlarının, PÖYAÖ'den aldıkları puanlardan daha yüksek olduğu görülmektedir. Bu çerçeveden bakıldığında, programlama sürecinde alınan eğitimin ve bu öğretim sürecindeki bireysel çabanın öğretmen adaylarının yeterlik algılarını doğrudan etkilediği söylenebilir. Bununla birlikte, özellikle programlama dersinin, programlamaya yeni başlayanlar için oldukça zor algılanması (Askar & Davenport, 2009) ve programlamaya ilişkin ön deneyimi olmayan öğrencilerin kendilerini süreçte yetersiz hissetmesi (Mazman & Altun, 2013) öğrencilerin programlama öz-yeterliği algılarını olumsuz etkilediği düşünülmektedir. Ayrıca yeterlik puanları ile PÖYAÖ'den aldıkları puanlar arasındaki korelasyon analizinde de negatif yönde anlamlı bir ilişkinin olması, aslında öğretmen adaylarının süreç sonunda uygulanan PÖYAÖ ölçeğiyle kendi yeterliklerinin farkına vardıkları şeklinde yorumlanabilir. Bu durum, öğrenenlerin programlama konusunda kendi yeterlik algılarının, öğretim süreci ile ilişkili olduğu ve başarılı bir şekilde yapılandırılan öğretim sürecinin öğretmen adaylarının programlama öz-yeterlik algısını etkilediği şeklinde ifade edilebilir. Benzer şekilde öz-yeterliğin, programlama performansını belirleyen önemli bir faktör olduğuna alanyazında da değinilmektedir (Ramalingam, LaBelle &Wiedenbeck, 2004; Jegede, 2009). Ayrıca bu çalışmada ortaya çıkan programlama ve problem çözme süreçlerinin paralel yapısı düşünüldüğünde, bireylerin programlamaya ilişkin öz-yeterlik algılarını etkileyen faktörler arasında bireyin sahip olduğu problem çözme becerileri de sayılabilir.

Programlama ve problem çözme süreçlerinin ilişkisi göz önüne alındığında, programlama konusunda öğrencilerin başarılı ve yüksek öz-yeterlik algılarına sahip olabilmeleri için özellikle problem çözme sürecinin programlama eğitimine entegre edilmesi önerilmektedir. Ayrıca öğrencilerin programlamadaki öz-yeterlik algıları

onlara verilen programlama eğitimi ile ilişkili olduğundan söz konusu eğitimin niteliğinin geliştirilmesi gerekir. Bununla birlikte, programlama dersini yürüten öğreticilerin bilgi aktaran rolünden ziyade, öğrenenlere rehber olarak onların bilgiyi yapılandırmasına fırsat vermesi gereklidir. Programlama dersini yürüten öğreticilere öğretim sürecinde öğrencilerine yardımcı olabilmeleri adına ders sürecinde ve kodlamada açıklayıcı geri bildirimlerin yanı sıra program çalıştırıldıktan sonra oluşabilen mantıksal ve yazımsal hatalarla ilgili de geri bildirimlerde bulunmaları bu süreci zenginleştirmeye katkı sağlayabilir. Dolayısıyla programlama öğretiminde farklı aşamalarda verilen geri bildirimlerin öğrenme üzerindeki yansımalarını inceleyen çalışmaların yanı sıra tutum, akademik başarı gibi değişkenlerin programlama öğretimine yönelik etkisinin incelendiği çalışmalar yapılması önerilmektedir.

Çıkar Çatışması ve Etik Bildirimi

Makalede bilimsel, etik ve alıntı kurallarına uyulmuş; bilimsel gerçekler üzerinde herhangi bir çarpıtma yapılmamıştır. Etik ihlal sorumluluğunun yazarlara ait olduğu ve bu makalenin daha önce başka bir akademik platformda yayınlanmamış ve paylaşılmamış olduğu yazarlar tarafından taahhüt edilmiştir. Bu makalenin araştırılması, yazarlığı ve/veya yayınlanmasına ilişkin yazarların kendi içinde ve diğer kişi/kurum/kuruluşlarla herhangi bir çıkar çatışması söz konusu değildir. Tüm araştırmacıların çalışmaya katkısı eşit düzeydedir.

Kaynakça

Akkoyunlu, B., & Kurbanoğlu, S. (2004). Öğretmenlerin bilgi okuryazarlığı öz-yeterlik inancı üzerine bir çalışma. *Hacettepe Üniversitesi Eğitim Fakültesi Dergisi*, *27*, 11-20.

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

Altun, A., & Mazman, S. G. (2012). Programlamaya ilişkin öz yeterlilik algısı ölçeğinin Türkçe formumun geçerlilik ve güvenirlik çalışması. *Eğitimde ve Psikolojide Ölçme ve Değerlendirme Dergisi*, *3*(2), 297-308.

Askar, P., & Davenport, D. (2009). An investigation of factors related to self-efficacy for Java Programming among engineering students. *Online Submission*, *8*(1), 26-32.

Bagley, C. A., & Chou, C. C. (2007, June 25-27). Collaboration and the importance for novices in learning java computer programming [Conference session]. *12th annual SIGCSE conference on Innovation and technology in computer science education.* Dundee, Scotland. https://doi.org/10.1145/1268784.1268846

Baki, A., & Bell, A. (1997). *Ortaöğretim matematik öğretimi*. YÖK Dünya Bankası.

Balanskat, A., & Engelhardt, K. (2015). *Computing our future: Computer programming and coding-Priorities, school curricula and initiatives across Europe*. Belgium: European Schoolnet.

Bandura, A. (1977). Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, *84*(2), 191-215. https://doi.org/10.1037/0033-295X.84.2.191

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

Chen, C., Haduong, P., Brennan, K., Sonnert, G., & Sadler, P. (2019). The effects of first programming language on college students' computing attitude and achievement: a comparison of graphical and textual languages. *Computer Science Education*, *29*(1), 23-48. https://doi.org/10.1080/08993408.2018.1547564

Compeau, D. R., & Higgins, C. A. (1995). Application of social cognitive theory to training for computer skills. *Information systems research*, *6*(2), 118-143. https://doi.org/10.1287/isre.6.2.118

Creswell, J.W., & Plano Clark, V.L. (2014). *Karma yöntem araştırmaları: Tasarımı ve yürütülmesi* (2. baskı). (Çeviri ed., Y. Dede & S.B. Demir), Anı Yayıncılık.

Creswell, J.W. (2012). *Educational research: Planning, conducting, and evaluating quantitative and qualitative research* (4th ed.). Pearson Education.

Çakıroğlu, Ü., & Öztürk, M. (2017). Flipped classroom with problem based activities: exploring self-regulated learning in a programming language course. *Educational Technology & Society*, *20*(1), 337-349.

Çankaya, S., Durak, G., & Yünkül, E. (2017). Education on programming with robots: examining students' experiences and views. *Turkish Online Journal of Qualitative Inquiry*, *8*(4), 428-445. https://doi.org/10.17569/tojqi.343218

Çoşğun, Ü. Ç., & Çoşğun, V. (2018). Programlama öğretiminin ortaokul öğrencilerinin öz-düzenleme stratejileri ve motivasyonel inançları üzerindeki etkisi. *Gaziantep Üniversitesi Eğitim Bilimleri Dergisi*, *2*(2), 59-71.

Dasso, A., Funes, A., Riesco, D. E., Montejano, G. A., Peralta, M., & Salgado, C. H. (2005). Teaching programming. In I *Jornadas de Educación en Informática y TICs en Argentina*, 183-186

Davenport, C. E. (2018). Evolution in student perceptions of a flipped classroom in a computer programming course. *Journal of College Science Teaching*, *47*(4), 30-35. https://doi.org/10.2505/4/jcst18_047_04_30

Demir, G. Ö., & Seferoğlu, S. S. (2017). Yeni kavramlar, farklı kullanımlar: Bilgi-işlemsel düşünmeyle ilgili bir değerlendirme. H.F.Odabaşı, B. Akkoyunlu & A. İşman (Eds.), *Eğitim teknolojileri okumaları 2017* içinde (ss. 801- 830). Pegem Akademi.

Demirer, V., & Nurcan, S. A. K. (2015). Türkiye'de Bilişim Teknolojileri (BT) eğitimi ve BT öğretmenlerin değişen rolleri. *Uluslararası Eğitim Bilimleri Dergisi*, (5), 434-448. https://doi.org/10.16991/INESJOURNAL.181

Durak, H. Y., Yilmaz, F. G. K., & Yılmaz, R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology*, *10*(2), 173-197. https://doi.org/10.30935/cet.554493

Dönmez-Usta, N., & Turan Güntepe, E. (2019). Bilişim teknolojileri rehber öğretmenlerinin programlama araçlarına ilişkin deneyimlerinin incelenmesi. *Amasya Üniversitesi Eğitim Fakültesi Dergisi, 8*(2), 373-396.

EDC (2018). *Developer population growth shifts toward China, India and emerging countries*, https://evansdata.com/press/viewRelease.php?pressID=268 adresinden 20 Şubat 2020 tarihinde alındı.

Fukuzawa, S., Boyd, C., & Cahn, J. (2017). Student motivation in response to problem-based learning. *Collected Essays on Learning and Teaching*, *10*, 175-188. https://doi.org/10.22329/celt.v10i0.4748

Garner, S. (2007). A program design tool to help novices learn programming. *ICT: Providing choices for learners and learning*, 321-324.

Gomes, A., & Mendes, A. J. (2007, September 3-7). Learning to program-difficulties and solutions [Conference session]. *International Conference on Engineering Education–ICEE*. Coimbra, Portugal.

Gundurao, H. K., Manjunath, N. S., & Nachappa, M. N. (2010). *Computer technology and computer programming*. Himalaya Publishing House.

Gurer, M. D., Cetin, I., & Top, E. (2019). Factors affecting students' attitudes toward computer programming. *Informatics in Education*, *18*(2), 281-296. https://doi.org/10.15388/infedu.2019.13

Han, S. J., & Kim, S. S. (2016). The effects of app programing education using m-Bizmaker on creative problem solving ability. *The Journal of Korean Association of Computer Education*, *19*(6), 25-32.

Hawi, N. (2010). The Exploration of student-centred approaches for the improvement of learning programming in higher education. *Online Submission*, *7*(9), 47-57.

Hung, Y. C. (2008). The effect of problem-solving instruction on computer engineering majors' performance in Verilog programming. *IEEE Transactions on Education*, *51*(1), 131-137. https://doi.org/10.1109/TE.2007.906912

Jegede, P. O. (2009). Predictors of java programming self efficacy among engineering students in a Nigerian university. *International Journal of Computer Science and Information Security, 4*(1&2).

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys*, *37*(2), 83-137. https://doi.org/10.1145/1089733.1089734

Kordaki, M. (2010). A drawing and multi-representational computer environment for beginners' learning of programming using C: Design and pilot formative evaluation. *Computers & Education*, *54*(1), 69-87. https://doi.org/10.1016/j.compedu.2009.07.012

Korkmaz Ö., Şahin, H., Çakır, R., & Erdoğmuş, F. U. (2019). Bilişim teknolojileri öğretmenlerinin kodlamaya dönük tutumları, öz-yeterlilikleri ve kodlama öğretimi için kullandıkları yöntemler. *Ondokuz Mayıs Üniversitesi Eğitim Fakültesi Dergisi, 38*(2), 1-16. https://doi.org/10.7822/omuefd.612449

Law, K. M., Lee, V. C., & Yu, Y. T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education*, *55*(1), 218-228. https://doi.org/10.1016/j.compedu.2010.01.007

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004, January 30-31). Scratch: a sneak preview [education]. *Second International Conference on Creating, Connecting and Collaborating through Computing, 2004*. Kyoto. https://doi.org/10.1109/C5.2004.1314376

Mazman, S. G., & Altun, A. (2013). Programlama–I dersinin BÖTE bölümü öğrencilerinin programlamaya ilişkin öz yeterlilik algıları üzerine etkisi. *Öğretim Teknolojileri & Öğretmen Eğitimi Dergisi*, *2*(3), 24-29.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis*. Thousand Oaks, Sage

MEB (2018). *Bilişim teknolojileri ve yazılım dersi Öğretim programı (Ortaokul 5 ve 6. Sınıflar)*, http://mufredat.meb.gov.tr/ adresinden 12 Ocak 2020 tarihinde alındı.

Mulder, F. (2002). Computer science: from a BÈTA to a DELTA subject. *Informatica, Tinfon*, *11*(2), 48.

Noone, M., & Mooney, A. (2018). Visual and textual programming languages: A systematic review of the literature. *Journal of Computers in Education, 5*(2), 149-174. https://doi.org/10.1007/s40692-018-0101-5

Özyurt, H., & Özyurt, Ö. (2015). A study for determining computer programming students' attitudes towards programming and their programming self-efficacy, *Journal of Theory & Practice in Education (JTPE), 11*(1), 51-67.

Qian, Y., & Lehman, J. (2019). An investigation of high school students' errors in ıntroductory programming: a data-driven approach. *Journal of Educational Computing Research*, 0(0), 1-27. https://doi.org/10.1177/0735633119887508

Ramalingam, V., & Wiedenbeck, S. (1998). Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. *Journal of Educational Computing Research*, *19*(4), 367-381. https://doi.org/10.2190/C670-Y3C8-LTJ1-CT3P

Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004, June 28-30). Self-efficacy and mental models in learning to program [Conference session]. *9th annual SIGCSE conference on Innovation and technology in computer science education*. Leeds, United Kingdom. 1008042 https://doi.org/10.1145/1007996.1008042

Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2011). Teaching programming in Secondary school: A pedagogical content knowledge perspective. *Informatics in education*, *10*(1), 73-88.

Sayın, Z., & Seferoğlu, S. S. (2016, Şubat 3-5). Yeni bir 21. yüzyıl becerisi olarak kodlama eğitimi ve kodlamanın eğitim politikalarına etkisi. *Akademik Bilişim Konferansı*, Aydın.

Ünsal, Y., & Ergin, İ. (2011). Fen eğitiminde problem çözme sürecinde kullanılan problem çözme stratejileri ve örnek bir uygulama. *Savunma Bilimleri Dergisi*, *10*(1), 72-91.

Vatansever, Ö. (2018). *Scratch ile programlama öğretiminin ortaokul 5. ve 6. sınıf öğrencilerinin problem çözme becerileri üzerindeki etkisinin incelenmesi* (Tez No. 501053) [Yüksek lisans tezi, Uludağ Üniversitesi-Bursa]. Yükseköğretim Kurulu Başkanlığı Tez Merkezi.

Yağcı, M. (2018). Lise öğrencilerinin bilgi-işlemsel düşünme beceri düzeylerinin incelenmesi. *International Online Journal of Educational Sciences*, *10*(2), 81-96. https://doi.org/10.15345/iojes.2018.02.006

Yang, H. L., & Cheng, H. H. (2009). Creative self-efficacy and its factors: An empirical study of information system analysts and programmers. *Computers in Human Behavior*, *25*(2), 429-438. https://doi.org/10.1016/j.chb.2008.10.005

Yaşar E. (2014). *Algoritma ve programlamaya giriş* (5. Baskı). Ekin Basım ve Dağıtım.

Yecan, E., Özçınar, H., & Tanyeri, T. (2017). Bilişim teknolojileri öğretmenlerinin görsel programlama öğretimi deneyimleri. *Elementary Education Online*, *16*(1), 377-393. http://dx.doi.org/10.17051/io.2017.80833

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

Yükseltürk, E., & Altiok, S. (2016). Bilişim teknolojileri öğretmen adaylarının programlama öğretiminde scratch aracının kullanımına ilişkin algıları. *Mersin University Journal of the Faculty of Education*, *12*(1), 39-52. https://doi.org/10.17860/efd.94270

Yükseltürk, E., & Altiok, S. (2017). An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology*, *48*(3), 789-801. https://doi.org/10.1111/bjet.12453

Extended Abstract

Information technology teachers play an important role in teaching programming to young individuals. In this respect, it is thought that the studies carried out to reveal the perceptions, attitudes, and views of the candidate teachers towards programming and programming teaching are important for better designing the related teaching process. Also, it is emphasized that self-efficacy and problem solving processes are related and that success in the programming process can be achieved when these relationships are established effectively (Sayın & Seferoğlu, 2016). In this respect, the aim of the study is to determine the process followed by teacher candidates and their competencies in the programming education process. For this purpose, the following questions related to the programming process were tried to be answered: "Which steps the teacher candidates follow in the process of designing the right way to solve a problem?"; "Which steps the teacher candidates follow in the process of expressing the problem with correct codes?";"Which steps the teacher candidates follow in the error checking process to solve the problem?" and "What are the teacher candidates' perceptions of self-efficiency in this process?".

In this study in which the embedded mixed research design is preferred, the study of the programming teaching process of teacher candidates was carried out in the form of a case study, which is one of the qualitative research designs, and correlation analysis and descriptive statistics were used in the comparison of the scores. In the spring semester of the 2018-2019 academic year, 24 teacher candidates who studied in the Department of Computer and Instructional Technologies Education and took the Programming Languages II Course formed the study group. "Knowledge Collection Form of the Problem Solving in Programming Process" (KCFPSP) was used in the programming process, which consists of 10 open-ended questions for the investigation of the programming education process of teacher candidates. Three of these questions in KCFPSP aimed at unlocking the process of designing the right way to solve the problem, also, two of which aimed at revealing whether the process was expressed with the correct codes. Three of the questions on the scale are intended to demonstrate the process for checking errors and revealing software and logical errors.

In addition, the teacher candidates were asked to make a score for these 8 questions in terms of the impact/importance on the programming process. The remaining two questions consist of the candidates' self-efficacy in the programming process and the scores given to themselves for the programming knowledge. In addition to KCFPSP, the "Computer Programming Self-Efficacy Perception Scale" (CPSEPS), developed by Ramalingam and Wiedenbeck (1998) and adapted to Turkish by Altun and Mazman (2012), was used in the present study. While the data obtained from KCFPSP was analyzed by the content analysis method, the total scores of candidates were calculated using the data obtained from the CPSEPS. Because the scores given to themselves by the candidate teachers for programming knowledge and self-efficacy in KCFPSP and the total points they received from the CPSEPS did not show normal distribution, they were compared with Spearman Rho coefficient in the correlation analysis.

It is observed that the responses of the teacher candidates to the process of designing the correct way to solve a problem are gathered under three themes: The definition of the problem, analysis of the problem, and establishment of the algorithm. Under the theme of problem definition, many of the teacher candidates (f=13) noted the importance of the process of understanding the problem; Also, under the theme of problem analysis, teacher candidates (f=14) addressed the importance of establishing the algorithm correctly; And under the theme of establishing the algorithm, teacher candidates (f=19) noted the correct solution of the problem. Teacher candidates' responses to the process of expressing a problem with correct codes are gathered under two themes: Writing the codes and running the software. Under the theme of writing the codes, some of the teachers (f=10) mentioned that writing codes is the basis of the coding process. Under the theme of running the software, the teacher candidates (f=11) stated that this stage is important in the detection of errors. The error control in the problem solving process is grouped under three themes: Error control, software errors, and logical errors. The code for detecting errors (f=15) and understanding the problem correctly (f=9) are given under the theme of error checking of prospective teachers. Another code that is put forward by teacher candidates for the error-checking in the problem solving process is software errors. Many of the teachers (f=20) mentioned that software errors help solve the problem. Furthermore, the candidates emphasized that logical errors are the most basic errors in the problem-solving process (f=12). Furthermore, the programming process revealed from the statements of the teacher candidates and expressed above was found to have similar content to the problem solving stages proposed by Polya (1945 cited from Baki & Bell, 1997).

When the scores of teacher candidates, which were made to evaluate the impact of the eight problems in the PPBTF on the programming process, it is seen that the average points vary in the range of 6 to 8. Candidates who evaluated each stage in the software development process were found to give quite high scores for the first stage; the average of these scores (8.0) and the accumulation of the scores were evaluated as having a higher value than other stages. Furthermore, it is observed that the self-

efficacy scores given to themselves by a significant number of teacher candidates for the programming process (f=19) in KCFPSP are higher than the scores they received from CPSEPS. The correlation between the self-efficacy scores of the teacher candidates and CPSEPS scores was also determined to show a negative significant correlation. As a result of the study, when considering the relationship between programming and problem-solving processes, it may especially be suggested to carry out studies on developing problem solving process of students and practices and activities to support these process to enable students to have successful and high self-efficacy perceptions in programming.