ALGORİTMİK DÜŞÜNME

Alaattin PUSMAZ

- Matematik ve Fen Bilimleri Eğitimi Bölümü, Atatürk Eğitim Fakültesi, Marmara Üniversitesi, İstanbul/TÜRKİYE
- ORCID: 0000-0003-4755-4089
- Mail: apusmaz@gmail.com

Matematiksel Düşünme

Doksanlı yıllardan itibaren matematik derslerinde öğrencilerin sadece formülleri ezberleme ve prosedürleri uygulama becerisi değil, matematiksel kavramlar, prosedürler ve ilkeler hakkında derin ve birbiriyle ilişkili anlayışlar geliştirmelerinin önemine işaret edilmiştir. Son yıllarda öğrencilerin sadece matematiğin özünü anlama kapasitelerine değil, aynı zamanda "matematik yapma" kapasitelerine de giderek daha fazla vurgu yapılmaktadır. Matematik eğitimcilerine göre derinlikli anlama, matematiksel düşünme süreçlerine katılma kapasitesini yani matematiği yapanların ve kullananların yaptıklarına benzer şeyler yapmaktır: Problemleri anlama ve çözme, örüntü arama, varsayımda bulunma, kısıtlamaları inceleme, verilerden çıkarım yapma, soyutlama, icat etme, açıklama, gerekçelendirme, meydan okuma vb. (Stein, Grover, & Henningsen, 1996). Görüldüğü üzere matematiksel düşünme, öğrencilerin matematikle meşgulken yapabilecekleri birçok sürece işaret eden kapsamlı bir beceridir ve müfredatların en önemli genel hedeflerindendir. Bazı araştırmacılar burada olduğu gibi matematiksel düşünmeyi süreçler perspektifinden ele alırken bazıları da matematiksel kavramların gelişimi olarak ele almışlardır (Çelik, 2016).

Holyoak ve Morrison (2012, s.1) düşünmeyi "genellikle hedeflere hizmet etmek üzere, dünyanın gerçek veya olası durumlarını karakterize etmek için bilginin zihinsel temsillerinin sistematik olarak dönüştürülmesidir" şeklinde tanımlamaktadırlar. Matematik, düşünmeyi geliştirdiği bilinen en önemli araçlardan biri olarak kabul edilir ve bundan dolayı da matematik eğitimi temel eğitimin önemli yapı taşlarından birini, belki de en önemlisini oluşturur (Umay, 2003). Düşünmeye yönelik çalışmalar, düşünme üzerine biraz farklı bakış açılarını yansıtan birbiriyle ilişkili birkaç alt alanı içerir. Bunlardan biri felsefe ve mantık temelli uzun bir geleneğe sahip olan akıl yürütmedir ki (ya da eş anlamlısı muhakeme etme); bazı başlangıç bilgilerinden (öncüller) çıkarımlar (sonuçlar) yapma sürecine vurgu yapar. Standart mantıkta, öncüllerin doğruluğu argümanlar sayesinde sonucun doğruluğunu garanti ediyorsa çıkarım tümdengelimseldir. Öncüllerin doğruluğu sonucun doğruluğunu daha inandırıcı kılıyor ancak kesinlik vermiyorsa, çıkarım tümevarımsal olarak adlandırılır. Bir diğeri değerlendirme ve karar verme; bir seçeneğin değerinin veya belirli bir sonucu sağlama olasılığının netleştirilmesi (değerlendirme) ve alternatifler arasından seçim yapılmasını (karar verme) içerir. Bir başkası ise problem çözme; bir hedefe ulaşmayı sağlayacak bir eylem planının oluşturulmasını içerir (Holyoak & Morrison, 2012).

Matematik eğitimi üzerine yapılan çoğu araştırma, matematik yapmanın ayrılmaz bir parçası olarak matematiksel akıl yürütmenin önemini vurgulamaktadır (NCTM, 2000). Akıl yürütme, bütün etmenleri göz önünde bulundurarak düşünüp akılcı bir sonuca ulaşma sürecidir (Umay, 2003). Bunu etkili bir şekilde yapabilmek için düşünebilme potansiyelimizin güçlü olması, yeterli kavramsal bilgiye sahip olmak ve onu sonuca yaklaştıracak şekilde kullanabilmek gerekir (Holyoak & Morrison, 2012). Bu beceriler sayesinde karşılaştığımız problem durumunu sonuca ulaştırmak mümkün olabilir.

Matematiksel düşünme ve muhakeme becerileri; varsayımlarda bulunma ve tümdengelimli argümanlar geliştirme de dahil olmak üzere, yeni bakış açıları geliştirmek ve daha ileri fikirler üretmek için bir temel oluşturduklarından matematik öğrenmede önemlidir. Öğretmenler bu duruma uygun, derinlikli matematiksel düşünmenin norm olduğu entelektüel bir sınıf ortamı oluşturmaktan sorumludur (NCTM, 2000). Polya (1997) matematik yaparken ortaya çıkan akıl yürütme sürecindeki eylemlere vurgu yaparak tümdengelimli muhakemede matematiksel bilgilerimizi sağlama bağladığımızı, fakat akla yakın muhakeme ile hipotezlerimize destek olmaya çalıştığımızı ifade etmektedir. Polya'ya (1966) göre akla yakın muhakeme ve tahmin matematiksel olarak zor problemlerde bile (örneğin ispat yapma) işe yarardır ve makul olan bir tahmini başka bir tahminden ya da makul bir tahmini daha az makul tahminden ayırmada kullanışlıdır. Polya'nın bu vurgusu muhakemenin daha esnek bir yapısının da olabileceğine işaret etmektedir ve bunun matematik yaparken ya da öğrenirken kullanışlı olduğunu söylemektedir.

Öğrencilerin muhakeme becerilerini geliştirmede öğretmenin elindeki en önemli araç şüphesiz öğrencileri derinlikli düşünmeye sevk edebilecek tarzda hazırlanmış problemlerdir. Schoenfeld'e (1985) göre, hem prosedürleri hem de matematiksel olarak temellendirilmiş akıl yürütmeyi öğrenmek için öğrencilerin hem rutin hem de rutin olmayan problemleri nasıl çözecekleri konusunda pratik yapmaları önemlidir. Rutin görevler standart algoritmalarla (çözüm yolları) çözülmeye daha uygunken, rutin olmayan görevlerde ise çözüme ulaştıracak kesin algoritmalar yoktur ve bu tür görevlerde problem durumunun anlaşılması ve ardından uygun çözümün aranması gerekir (Cai, 2000). Standart bir algoritma ile ifade edilen işlemsel düzeyde ezbere dayalı veya taklitle yapılan çözümdür. Standart olmayan algoritmada ise öğrencilerden problem karşısında olayları incelemesi ve bunlar arasında bağıntı bulması beklenir. Ayrıca çözüm yolu yeni oluşturulacağı için ezber yöntem değil çözüm için gerekli üst bilişsel stratejiler kullanılır (Altun, 2008).

Öğretmenlerin derslerini verimli kılmak için öğrencilerini dikkatle gözlemlemesi, onların fikirlerini ve açıklamalarını dinlemesi de önemlidir. Tüm bunlar öğrencileri matematiksel düşünme ve muhakeme yapmaya motive eder ve farklı potansiyele sahip öğrencilere uygun öğrenme fırsatları oluşturur (NCTM, 2000).

Akıl yürütme, matematiksel fikirler arasındaki bağlantıları ve ilişkileri görme açısından özellikle önemlidir (English, 1998). Literatürde akıl yürütme genellikle yüksek seviyede tümdengelimli mantıksal nitelikte bir beceri olarak tanımlanmakla (Silver, 1997) birlikte farklı tanımlarının ve yaklaşımların da bulunduğu görülmektedir. Lithner'e (2008) göre akıl yürütme görev çözümünde hipotezler üretmek ve sonuç elde etmek için kullanılan düşünce yoludur ve biçimsel bir yolla yapılması zorunlu değildir yani ispatla sınırlı değildir. Ayrıca akıl yürüten kişiye göre mantıklı sebepleri olmak kaydıyla akıl yürütmeyle yanlış sonuçlar da elde edilebilir.

Kişi bazen karşılaştığı bir problemi çözmek için daha önce gördüğü ve hatırladığı benzer bir problemin çözümünde kullanılan stratejiyi kullanarak doğru cevaba ulaşabilir. Ya da çözüme ulaşması kitaptan, öğretmeninden veya arkadaşından gördüğü bir stratejiyi kullanmasıyla da olabilir. Bu bir çeşit taklit ederek problemi çözme yaklaşımıdır. Lithner (2008) bu yaklaşımı taklitçi muhakeme olarak isimlendirerek iki kısma ayırmıştır. İlki daha önce kullanılan bir stratejiyi hatırlayarak yeni problem durumu için kullanmaktan ibaret olan ezbere muhakemedir. Bu tür bir akıl yürütmenin ancak rutin, tanıdık problem durumlarında kullanılabileceği aşikardır. İkincisi ise problemin durumuna göre bazı işlem adımlarını kullanarak sonuca ulaşmaktır ve bu algoritmik muhakeme veya algoritmik akıl yürütme olarak isimlendirilir. Burada yapılan işlemlerin ne tür sonuçlar ortaya çıkardığı veya neden yapıldığı akıl yürüten tarafından bilinmez.

Lithner'in (2008) ezbere muhakemenin karşıtı olarak isimlendirdiği yaratıcı muhakeme ise matematiksel olarak inşa edilmiş ve içsel olarak yapılandırılmış iddiaları kullanarak problem çözmedir. Bu muhakemenin belirleyici üç özelliği; özgünlük, akla yatkınlık ve matematiksel dayanağa sahip olmadır. Özgünlük, akıl yürütme dizisinin akıl yürüten kişi için bilindik değil yeni olması; akla yatkınlık, seçilen veya kullanılan stratejilerin neden seçildiğinin makul bir açıklamasının olması; matematiksel dayanak ise akıl yürütmede kullanılan matematiksel yaklaşımın matematiğin temel prensipleri ile çelişmemesidir.

Matematik eğitimcileri uzun zamandır öğrencilerin matematiksel düşünme ve muhakeme becerilerinin nasıl geliştirilebileceğine, bunu destekleyen bir sınıf ortamının ne tür özelliklere sahip olması gerektiğine, bu hususstaki öğretmen yeterliklerinin tespit edilmesine, öğrencilerin muhakeme süreçlerinde yaşadıkları zorlukların neler olduğuna yönelik araştırmalara odaklanmışlardır. Tüm bunların yanında günümüzde teknolojideki gelişmelerle birlikte teknolojik araçlar insanların matematik yapma biçimlerini yeniden şekillendirmeye başlamıştır. Dijital teknolojilerin sunduğu olanaklar matematik derslerinin de bir parçası olmaya başlamış ve matematik öğretim sürecindeki etkili materyaller arasında yerini almıştır. Matematik öğretimine yönelik tasarlanmış birçok program (örneğin Cabri, Geogebra, Geometer's Sketchpad, Logo, Scratch, TinkerPlots) hem öğrencilerin derse karşı ilgi ve motivasyonlarını artırmış hem de matematiksel becerilerin gelişmesine katkı sağlamıştır. Yapılan bu araştırmalar ve ortaya çıkan yeni gelişmeler matematik eğitimcilerini hesaplamalı düşünme ve algoritmik düşünmeye yönelik araştırmalara da yönlendirmiştir. Bu çalışmalarda genellikle öğrencilerin nasıl algoritma geliştirdikleri, geliştirdikleri algoritmaların yapısı, algoritma sürecinde yaşadıkları zorluklar incelenmiştir. Bu bilgiler ışığında bu çalışmada matematik eğitimine yönelik nispeten son zamanlarda yapılan araştırmaların da konusu olan algoritmik düşünme kavramına odaklanılmış, kavram etraflıca ele alınarak matematik öğretimindeki yeri gerçekleştirilen araştırmaların sonuçları yardımıyla değerlendirilmiştir.

Hesaplamalı Düşünme

Matematik eğitimine yönelik yapılan çeşitli araştırmalarda, hesaplamalı düşünmenin öğretilmesi ve öğrenilmesi teşvik edilerek bunun modern teknolojik dünyayı anlamanın bir parçası olduğu ifade edilmiştir (Moala, 2021). Wing (2006) hesaplamalı düşünmenin önemini ve farklı alanlarda problem çözme süreçlerini etkileme potansiyelini tartıştığı makalesinde hesaplamalı düşünmeyi "bilgisayar biliminin temel kavramlarından yararlanarak problem çözmeyi, sistem tasarlamayı ve insan davranışlarını anlamayı içeren" bir süreç olarak tanımlar. Wing'e (2006) göre hesaplamalı düşünme bir problem çözme yaklaşımıdır.

Hesaplamalı düşünme kavramı ilk olarak 1980'lerin başında çocukların matematiksel ve hesaplamalı fikirleri yaratıcı ve sezgisel bir şekilde keşfetmeleri için LOGO programlama dilini geliştiren Papert tarafından kullanılmıştır. Papert'e göre hesaplamalı düşünme, çocukların LOGO'da

programlama yoluyla matematik öğrenirken uyguladıkları belirli düşünceleri ifade ediyordu (Stephens & Kadijevich, 2020). Benzer şekilde Wings (2006) de hesaplamalı düşünmeyi kodlamanın arka planındaki düşünce olarak ele almaktadır. Shute ve arkadaşlarına (2017) göre literatürde hesaplamalı düşünmenin tanımlanmasında belirgin bir çeşitlilik olmasına rağmen ayrıştırma, soyutlama, algoritmalar ve hata ayıklama hesaplamalı düşünmenin ortak bileşenleri olarak görülmektedir.

Algoritma ve Algoritmik Düşünme

Algoritma kısaca "uygulanabilen ve sistematik olarak bir sonuca kadar takip edilebilen, kesin olarak tanımlanmış rutin bir işlem" olarak tanımlanabilir. Bir algoritmayı anlamak, test etmek, geliştirmek ya da tasarlamak gerektiğinde algoritmik düşünme gereklidir (The Concise Oxford Dictionary of Mathematics, fourth ed, s. 11, akt. Stephens and Kadijevich, 2020). Knuth (1985) algoritmayı "üzerinde işlem yapılan verinin yapısı ve gerçekleştirilen işlemler dizisinin yapısı da dahil olmak üzere, iyi tanımlanmış süreçlerle ilgili tüm kavramları kapsayacak şekilde" kapsamlı bir yapı olarak ele alır.

Algoritmik düşünme ise cebirsel, uzamsal ve geometrik ve istatistiksel gibi birçok şekilde olabilen matematiksel akıl yürütmenin bir şeklidir (Stephens & Kadijevich, 2020). Algoritmik düşünme aynı zamanda bir düşünme yöntemi ve kişinin düşüncesi hakkında düşünmesi için bir araçtır (Mingus & Grassl, 1998; akt. Lehmann, 2023b). Bazı araştırmacılar algoritmik düşünme ve hesaplamalı düşünmeyi birbirinin yerini tutan iki kavram olarak ela almaktadırlar. Stephens ve Kadijevich'e (2020) göre algoritmik düşünme ayrıştırma, soyutlama ve algoritma tasarımı basamaklarını içerir.

Algoritma üretmek bir problem karşısında nispeten yeni bir çözüm yolu oluşturmak demektir. Buna ek olarak oluşturulan bu çözümün doğruluğunu sınamak ve benzer problemlere genelleştirmek de algoritma üretme kapsamında değerlendirilmektedir (Rasmussen vd., 2005). Algoritmik düşünme algoritmaları oluşturmak, anlamak veya değerlendirmek için gereken bir dizi bilişsel beceri ile de ilişkilidir (Lehmann, 2023a). Moala (2021) algoritmik düşünmeyi bir algoritma oluşturma, onu test etme ve ardından elde edilen sonuçlara göre algoritmayı revize etme şeklinde yinelemeli bir süreç olarak ele alır.

Algoritma oluşturma sürecindeki aşamalar problem çözme sürecine benzemektedir. Rutin olmayan bir problemi çözerken başlangıçta istenenin ne olduğunu bilmek ve verilenlerin sonuca ulaşmak için yeterli olup olmadığını kontrol etmek gibi eylemler algoritma görevlerindeki kapsamlı yapıyı ayrıştırma basamağıyla özdeştir. Problemi çözerken uygun bir strateji belirleyip kullanma algoritma oluşturma basamağı ile, problemin çözümünden elde edilen sonucu kontrol etme, gerekirse kullanılan stratejide değişiklik yapma ise algoritmayı revize etme ve genelleme basamağı ile özdeştir.

Algoritma oluşturma sürecine ait özyineleme, yineleme ve algoritmaların karşılaştırılması gibi matematiksel kavramlar, teknolojik bir dünyada artan ilgileri ve faydaları nedeniyle okul matematiğinde daha fazla ilgi görmektedir (NCTM, 2000, s. 16). Matematik öğretimine bilgisayar ortamında algoritma geliştirmeyi entegre etmek büyük önem taşımaktadır. Bu sayede öğrencilerin matematiksel yeterlilik için gerekli olan hesaplamalı düşünme ve problem çözme becerilerini geliştirmek (Resnick vd., 2009), etkileşimli görsel temsiller aracılığıyla kavramsal anlamayı geliştirmek (Brennan & Resnick, 2012; Calder, 2010) mümkün olur. Algoritmik kodlama ve matematik arasındaki bu sinerji ile öğrencilerin hem matematiksel hem de teknolojik yeterliklerini geliştirerek dijital çağın temel becerilerine katkı yapılmış olunur.

Öğrencilerin algoritma üretmelerine yönelik tasarlanmış Scratch, Logo gibi kodlama programları algoritmik düşünme becerisine önemli katkılar sunmaktadır. Bunlardan bir olan Scratch, ilkokul ve ortaokul çağındaki öğrencilerin algoritmik düşünmelerini geliştirmek için kullanılan bir kodlama programıdır (Resnick vd., 2009). Bu program, 2003 ile 2007 yılları arasında Massachusetts Teknoloji Enstitüsü Laboratuvarı tarafından geliştirilmiştir. Scratch programının ara yüzü metin tabanlı değil blok temelli görsel bir ara yüzdür. Öğrenciler Scratch ile proje oluştururken farklı yaklaşımlar sergileyerek ve kendi sorumluluklarını üstlenerek bir interaktif tasarım süreci içerisine girerler (Genç & Karakuş, 2011). Öğrencilerin Scratch gibi bir programda algoritma geliştirmeye çalışmaları onların ilgilerini çekmekte ve matematiksel kavramları öğrenmek için motive edici bir ortam sağlamaktadır (Calder, 2010).

Son zamanlarda yapılan araştırmalar, matematik eğitimi müfredatlarında algoritmik düşünmeye yer verilip verilmemesini tartışmaktadır (Stephens & Kadijevich, 2020). Çeşitli araştırmalarda

öğrencilerin matematiksel algoritma üretme süreçlerine odaklanılmıştır (Calder, 2010; Lehmann, 2023a, 2023b; Moala, 2021; Tupouniua, 2023). Ayrıca kod temelli programlar (örneğin Scratch programı) ile algoritma geliştirmeye odaklanan araştırmaların da (Calder, 2010; Ozoran vd., 2012; Korkmaz, 2016) mevcut olduğu görülmektedir. Bu araştırmalarda öğrencilerin Scratch gibi blok temelli programlarda algoritma oluşturmalarının onlar üzerinde olumlu etkiye sahip olduğu (Korkmaz, 2016), algoritma geliştirme becerilerine olumlu katkı yaptığı (Calder, 2010, Ozoran vd., 2012) ve kavramsal öğrenmelerinin, problem çözme becerilerinin gelişimine yardımcı olduğu (Calder, 2010) sonuçları ortaya konulmuştur.

Matematik öğretmenleri algoritmik düşünmeyi öğretmeleri gereken başka bir şey olarak değil, genel olarak problem çözmeye yönelik pedagojik bir yaklaşım, birçok disipline aktarılabilecek bir beceri olarak düşünmelidir (Clark, 2016; akt. Stephens, 2018). Algoritmalarla başarılı bir şekilde başa çıkmak için algoritmik düşünme, ayrıştırma (bir problemi alt problemlere ayırma) ve soyutlama (temel kavramlar, prosedürler, ilişkiler ve modellerle ilgili belirli örnekleri özetleyen genel ifadeler yapma) dahil olmak üzere farklı bilişsel yetenekler gerekir (Stephens & Kadijevich, 2020). Stephens (2018, s.1) algoritmik düşünmenin, "ayrıştırma (karmaşık bir problemi bileşen alt problemlere ve alt görevlere ayırma), örüntü tanıma, genelleme ve soyutlama" gibi bilişsel becerilerden oluşan özel bir matematiksel muhakeme türü olduğunu ifade etmektedir. Lehmann (2023a) da algoritmik düşünmenin ayrıştırma, soyutlama, algoritma oluşturma ve hata ayıklama becerilerinden oluştuğunu belirtmektedir.

Matematik eğitiminde öğrencilerin algoritmik düşünme becerilerini geliştirme çabaları farklı şekillerde ortaya çıkmaktadır; bunlardan biri, öğrencilerin nispeten yeni problemleri çözmek için bir algoritma oluşturmayı ve ifade etmeyi gerektiren algoritma oluşturma görevleriyle meşgul olmalarıdır (Tupouniua, 2023).

Algoritmik düşünmenin eğitim alanları arasında farklı tanımları bulunmakla birlikte algoritma oluşturmak için gerekli bilişsel becerilerin; ayrıştırma, soyutlama, algoritma oluşturma ve hata ayıklama basamaklarının birçok araştırmacı tarafından kabul gördüğü görülmektedir. Bundan sonraki bölümde algoritma oluşturmanın basamakları ele alınmıştır.

a. Ayrıştırma

Algoritmik düşünme sürecinin ilk basamağı olarak kabul edilen ayrıştırma karmaşık bir problemi alt problemlere ve alt görevlere ayırmayı içerir (Stephens, 2018; Stephens ve Kadijevich, 2020). Kwon ve Cheon'e (2019) göre bir probleme çözüm tasarlamak için ayrıştırma gerekir. Ayrıştırma sürecinde karmaşık bir problem daha küçük parçalara ayrılır ve bu küçük problemlerin her birini ele almak için sistematik süreçler kullanılır (Shute vd., 2017). Lockwood ve arkadaşları (2016, s. 1591) da algoritmik düşünmeyi tanımlarken bu hususu ön plana çıkarmışlardır; "mevcut araçları kullanarak karmaşık bir hedefi bir dizi (sıralı) adıma ayırmak için kullanılan mantıksal, organize bir düşünme biçimi."

Polya (1997) da bir problemin çözümünde öncelikle problemi bir bütün olarak ele almayı ve ardından ayrıntılara odaklanarak problemi ayrıştırmayı tavsiye etmiştir. Bu yaklaşım matematik eğitiminde höristik problem çözme stratejilerden biri olarak kabul edilmektedir ve araştırmalar bu yaklaşımın problem çözmede etkili olduğunu vurgulamaktadır. Höristik, herhangi bir konudan veya sorundan bağımsız olarak, probleme uygun bir şekilde yaklaşmayı, problemi çözmede elindeki kaynakları etkili bir şekilde kullanma konusunda problem çözücüye yardımcı olmayı sağlayan genel bir öneri veya strateji olarak tanımlanabilir (Schoenfeld, 1980).

Bilgisayar bilimleri eğitimi perspektifinden bakıldığında ayrıştırma örneğin Scratch'te bir algoritma oluştururken alt görevleri belirlemekle ilgilidir (Kwon ve Cheon, 2019). Eğitim literatüründe ayrıştırma tanımları ve örnekleri arasında tutarlılık olduğu görülmektedir; ancak öğrencilerin algoritmik düşünme bağlamlarında ayrıştırmayı nasıl kullandıklarına dair sınırlı sayıda ampirik araştırma olduğu görülmektedir (Lehman, 2023b).

b. Soyutlama

Algoritmik düşünme sürecinin ikinci basamağı soyutlama karmaşık sistemlerden temel örüntülerin damıtılması anlamına gelmektedir (Shute vd., 2017, s.150) ve hesaplamalı düşünmenin özü olarak kabul edilir (Wing, 2008). Stephens ve Kadijevich (2020) algoritmik düşünme ile ilgili olarak soyutlamayı "altta yatan kavramlar, prosedürler, ilişkiler ve modellerle ilgili belirli örnekleri özetleyen genel ifadeler oluşturmak" şeklinde tanımlamaktadır (s. 118).

Hesaplamanın zihinsel aracı olan soyutlama, hangi ayrıntıları vurgulamamız gerektiğini, hangilerini göz ardı edebileceğimizi tespit etmede

bize yardımcı olur (Wing, 2008). Soyutlama yapma önemli bir bilişsel beceri olan genelleme yapma ile de ilişkilidir. Soyutlama sürecinde, problemi çözerken örüntüler bulma ve böylece benzer problemlerde kullanılabilecek bir yaklaşım oluşturma, genelleme yapma mümkün olur (Shute vd., 2017).

Sonuç olarak soyutlama, karmaşık bir problemi çözümlemek, verilerden hangilerinin kullanılıp kullanılmayacağını belirlemek, bu probleme ve benzerlerine çözüm olabilecek genel bir yaklaşımı geliştirmek gibi hususları barındıran kapsamlı bir bilişsel süreçtir. Öğrencilerin özellikle rutin olmayan problemlerle meşgulken verilenlerden hangilerinin gerekli hangilerinin gereksiz olduğunu tespit etmesi ve eksik bilginin olup olmadığını anlayabilmesinin ne kadar önemli olduğu göz önüne alındığında soyutlama becerisinin algoritmik düşünme için önemi daha da belirginleşir.

Wetzel ve arkadaşları (2020) üstün yetenekli öğrencilerle yaptıkları en kısa yolu bulma ile ilgili çalışmada öğrencilerin gerçek hayat durumu ile ilgili bir görevde soyutlama yapmada zorlandıklarını tespit etmişlerdir. Öğrenciler ancak kendilerinden istendiğinde problemdeki gereksiz ayrıntıları çıkartmaya odaklanmışlar fakat bunda pek başarılı olamamışlardır. Araştırmacılara göre soyutlamayı uygun bir şekilde yapamamak diğer aşamalarda da zorluklar yaşamaya sebep olabilmektedir. Lehman (2023a) bir grup öğrenci ile yaptığı çalışmasında öğrencilerin bir dizi soyutlama alt becerisi kullandığı sonucuna ulaşmıştır. Bunlar soruya ait temel bileşenleri belirleme, bunları temsil edecek grafikleri oluşturma ve bunları nasıl kullanacaklarını tespit etme. Benzer sonuçlar başka araştırmalarda da ortaya çıkmıştır (Moala, 2021; Moala ve ark., 2019).

c. Algoritma Oluşturma

Algoritmik düşünmede merkezi bir konuma sahip olan algoritma oluşturma süreci aynı zamanda bilişsel beceriye en çok ihtiyaç duyulan aşamadır. Algoritma, girdileri istenilen çıktılara dönüştürecek şekilde adım adım ilerleyen bir prosedürün soyutlanmasıdır (Wing, 2008). Algoritma oluşturma sürecinde problem alt problemlere ayrılarak modüler, parçalı çözümler yazılır (Stephens, 2018). Bundaki amaç büyük veya kapsamlı görevi küçük parçalara ayırarak çözmekdir. Shute ve arkadaşlarına (2017) göre algoritma oluşturma, bir problemi çözmek için bir dizi mantıksal ve sıralı adım tasarlama sürecidir ve birden fazla kaynaktan gelen bilginin belirli sayıda adımla aynı anda gerçekleştirilmesini sağlamalıdır. Ayrıca algoritma problemi çözmek için en az sayıda adım kullanılarak tasarlanmalıdır, yani gereksiz adımlar kullanılmamalıdır. Bu durum Knuth (1997) tarafından etkililik olarak

isimlendirilir. Yani bir algoritmadaki işlemlerin tamamı, bir kişi tarafından kalem ve kağıt kullanılarak yapılabilecek kadar basit olmalıdır. Algoritmaların etkili/verimli olabilmesi soyutlama ve genelleme becerileri ile ilgilidir (Shute vd., 2017).

Algoritma oluşturma görevlerinde, öğrenciler bir algoritma oluşturma, algoritmayı belirli problemler üzerinde test etme ve test sonuçlarına göre algoritmayı revize etme gibi yinelemeli bir sürece girerler (Moala, 2021). Bu durum benzer problemleri çözmek için yeniden kullanılabilir araçların/prosedürlerin geliştirilmesini sağlar. Dolayısıyla oluşturulan algoritma benzer problemleri çözmede kullanılabilecek bir yapıda olmalıdır (Shute vd., 2017). Böylelikle aslında her ne kadar süreç spesifik bir problem durumuyla başlamış olsa da ortaya çıkan yaklaşım yani üretilen algoritma problemin bir parçası olduğu daha kapsamlı genel bir problem kümesinde de kullanılabilir olmalıdır.

Knuth (1997) algoritmaların temel özelliklerini sonluluk, kesinlik, girdi, çıktı ve etkililik olarak ifade etmiştir. Buna göre algoritmalar her zaman sonlu sayıda adımda sona ermeli yani sonsuza kadar çalışmamalıdır. Bu özellik algoritmanın sonlu olması ile ilgilidir. Kesinlik özelliği, algoritmanın gerçekleştireceği her eylemin her durum için kesin ve net bir şekilde belirtilmesini ifade eder. Algoritmaların belli bir nesne kümesinden alınan sıfır ya da daha fazla girdisi vardır. Bu girdilerle belli bir ilişkiye sahip olan bir veya daha fazla çıktısı vardır. Etkililik özelliği ise algoritmanın işlemlerinin bir kişi tarafından sınırlı bir süre içinde yapılabilmesini yani basit olmasını ifade eder.

Matematik öğretiminde algoritmik düşünmeye yönelik yapılan araştırmalarda algoritma oluşturma basamağına özellikle odaklanıldığı görülmektedir. Wetzel ve arkadaşları (2020) üstün yetenekli öğrencilerle yaptıkları çalışmada öğrencilerin çözümlerini geliştirerek uygun bir algoritma üretebildiklerini bulmuşlardır. Lehman (2023a) da bir grup lise öğrencisinin yeni algoritmik kavramları ya da eylemleri bir araya getirerek algoritma oluşturabildiklerini ve ayrıca yineleme/döngüler ve değişken kullanımı gibi kavramları analiz edebilmelerinin algoritma tasarımlarına destek olduğunu bulmuştur. Bu çalışmada öğrenciler iki çeşit algoritma oluşturma yaklaşımı sergilemişlerdir. Bunlardan ilki çözdükleri problemleri adım adım anlatarak bunu algoritmaya dönüştürmeleri şeklindedir (çözüm açıklama yaklaşımı). İkincisi ise önce algoritma taslağı oluşturup daha sonra bu taslağa aşamalı olarak eylemler ekleme (üst düzey algoritma) şeklindedir. Moala ve

arkadaşları (2019) bir grup lisans öğrencisinden bir şekil ile temsil edilen arkadaşlık optimizasyon problemi için algoritma geliştirmelerini ve ardından geliştirdikleri algoritmanın farklı arkadaşlık ilişkileri için de kullanılabilir olup olmadığını tespit etmelerini istemişlerdir. Sonuç olarak öğrenciler geliştirdikleri algoritmayı diğer örnekler için kontrol ederken örneğe uygun olduğunu düşündükleri algoritma basamaklarında (yerel hususlar) herhangi bir değişiklik yapmadıklarını, uygun olmayan basamaklarda ise düzenlemeler yaptıklarını -ki bunlar ekleme veya çıkarma şeklindeki yamalardır- tespit etmişlerdir. Tupouniua (2023) öğrencilerin algoritma oluştururken yaşadıkları zorlukları tespit etmeye yönelik araştırmasında ortaya çıkan zorluklardan bahsetmektedir. Tespit ettiği zorluklardan biri öğrencilerin problemi çözebilmelerine rağmen benzer problemleri çözebilecek algoritma oluşturamamaları, yani yöntemleri ile ortaya çıkan algoritmanın tutarsız olmasıdır. Bir diğeri ise oluşturdukları algoritmayı test ederken revize etmelerine rağmen geçerli bir algoritma üretememeleri yani istenilen sonuca ulaşamamalarıdır. Öğrencilerin Scratch programında algoritma oluşturmaları, onların programlama ve algoritma oluşturma becerilerine olumlu katkı sağlamaktadır (Ozoran vd., 2012).

d. Hata Ayıklama

Bir algoritmanın doğruluğunu ve verimliliğini test etme aşaması hata ayıklama olarak isimlendirilir (Shute vd., 2017). Hata ayıklama, bir algoritmayı, eldeki problemi başarılı bir şekilde çözecek şekilde test etme ve iyileştirme yeteneğidir (Lehman, 2023a) ve algoritmanın doğru bir şekilde çalışıp çalışmadığı hakkında karar verdikten sonra geçilen bir aşamadır. Yani algoritma oluşturulduktan sonra hataların belirlenmesi, tanımlanması ve düzeltilmesi anlamına gelir (Shute vd., 2017). Hata ayıklama, algoritmanın etkisiz olması durumunda alternatif yaklaşımların değerlendirilmesini de içerebilir (Ginat, 2008; akt. Lehman, 2023a). Lehman'a (2023a) göre öğrenciler algoritmayı revize etme ve algoritmaya adım ekleme şeklinde iki tür hata ayıklama yaklaşımı kullanırlar.

Oluşturulan bir algoritmada mutlaka hata olacağı varsayılamaz ama hatanın olup olmadığının sistematik bir şekilde kontrol edilmesi ve varsa bunların tespit edilmesi ve düzeltilmesi gerekmektedir. Knuth (1997) hata ayıklamayı üzerinde çok daha fazla çalışılması gereken bir sanat olarak tanımlar ve bu işlem için öncelikle uygun veri kümesinin belirlenmesini ardından tespit edilen hataların kaydının tutulması gerektiğini belirtmektedir.

Literatür incelendiğinde öğrencilerin algoritma oluşturma sürecindeki hata ayıklama becerilerine yönelik araştırmaların sınırlı olduğunu ve bu çalışmaların öğrencilerin algoritmik kavramları kullanma becerilerinin algoritmalarının hata ayıklamasında etkili olduğunu göstermektedir (Lehman, 2023b).

Algoritma oluşturmaya yönelik tasarlanan programlarda oluşturulan algoritmanın sorunsuz olup olmadığını test etmede programın yapısı da önemli bir rol oynamaktadır. Calder (2010) Scratch programının öğrencilerin problem çözme sürecini nasıl kolaylaştırdığı, matematiksel düşüncelerinin nasıl ortaya çıktığını incelediği çalışmada öğrencilerin algoritmalarındaki hataların kolayca görülmesinin, istenen sonucu elde etmek için daha fazla deneme yapmalarına teşvik etme hususunda olumlu bir etkisi olduğunu ifade etmiştir. Wetzel ve arkadaşları (2020) öğrenciler ancak geliştirdikleri algoritma ile ilgili ya da cevabın doğru olup olmadığı ile ilgili soru sorulduğunda hata ayıklama sürecine yöneldiklerini gözlemlemişlerdir. Tupouniua (2023) da öğrencilerin algoritmaları ile çözüm aşamalarının aynı olup olmadığını veya farklıysa algoritmanın doğru sonucu üretip üretmediğini nadiren incelediklerini belirterek, öğrencilerin algoritmalarını doğrulama hususunda desteğe ihtiyaçları olduğunu ifade etmiştir. Bu durumu problem çözme süreci ile ilişkilendirdiğimizde benzer sonuçların ortaya çıktığını görürüz. Öğrenciler problemi çözdüklerinde çoğu zaman çözümlerinin doğruluğunu test etmeye yönelmemektedirler. Problem çözmede iyi olan öğrenciler ise çözüme bir planla başlayarak bu planı hayata geçirirler. Ayrıca çözümlerini tamamladıklarında geriye dönüp bakarak çözümün doğruluğunu kontrol ederler. Bu tür öğrenciler ayrıca alternatif çözümler üretmeye de hazırdırlar (Cai & Brook, 2006). Farklı bakış açılarıyla probleme yaklaşabilme ve alternatif çözümler üretebilme, rutin olmayan problemler ya da algoritma oluşturma gibi zor görevlerde başarılı olmak için gerekli olan bir yetkinliktir. Bunun gelişmesi ise öğretmenlerin derslerinde bu hususlara dikkatlice eğilmesi ve alternatif yaklaşımlar üretmeyi gereksiz ve zaman kaybı olarak görmemesi ile olabilir.

Üstün yetenekli öğrencilerin hesaplamalı düşünme ile ilişkili becerilerini tespit etmeye odaklanan bir araştırmada katılımcıların algoritmik düşünme süreçlerinden olan hata ayıklama ve genelleme basamaklarında ileri düzeyde beceriler sergiledikleri fakat soyutlama becerilerinin temel düzeyde kaldığı görülmüştür. Yazarlar öğrencilerin algoritmik düşünmenin en önemli bileşenlerinden birisi olan soyutlama becerisindeki düşük bir performanslarına rağmen diğer aşamalarda daha yüksek başarı sergilemelerini

ilginç bir sonuç olarak yorumlamışlardır. Çünkü soyutlamaya yönelik temel becerilerin gözlemlenmesine rağmen öğrencilerin soyutlamaya geçiş yapamamaları beklenen bir durum değildir.

SONUÇ

Algoritmik düşünme, ilkokul ve ortaokul yıllarında matematik öğreniminde doğal olarak ortaya çıkmalı ve matematik öğrenmeyi desteklemelidir (Stephens, 2018). Öğrenciler algoritma geliştirmeye yönelik görevlerle meşgulken, günümüz matematik öğretiminde kritik bir öneme sahip olan ilişkisel düşünme ve üst düzey akıl yürütme becerisine ihtiyaç duyarlar. Algoritma geliştirme sürecinde kolayca üstesinden gelemeyecekleri yeni problemlerin oluşması öğrencileri çözümlerini daha kapsamlı bir bakış açısıyla oluşturmaya teşvik eder.

Bu durum bir program yardımıyla (örneğin Scratch) algoritma oluşturmaya çalışırken meydana geldiğinde, öğrenciler programın kullanmadıkları ve bazen bilmedikleri yeni özelliklerini de öğrenmeye ve yeni bazı matematiksel fikirler ortaya çıkarmaya yönelmelerini desteklemektedir. Örneğin program doğrudan asal sayıları listeleyen bir komuta sahip değilse bu durumda öğrenciler asal sayıların tanımını ve özelliklerini dikkate alarak onların bir listesini oluşturacak algoritma üretmek zorunda kalırlar. Bu tür ortamlar da matematiksel kavramları keşfetmek için motive edici bir ortam sağlar. Algoritma üretmeye yönelik çalışmalarda öğrencilerin grupça çalışmaları iletişim ve işbirliğini sağlamaya yönlendirmesi bakımından da etkili bir araçtır.

Algoritma geliştirmeye yönelik görevlerde öğretmenin ya da kullanılan programın geri bildirimleri, öğrencilerin düşüncelerini yeniden gözden geçirmelerine (Clements vd., 2008) veya eksiklikleri ortadan kaldıracak müdahaleler yapmalarına yardımcı olur. Bu durum izleme ve değerlendirme gibi üst bilişsel becerilerin ve yapılan işlemlerin, kullanılan stratejinin gözden geçirilmesi, hataların ayıklanması gibi problem çözmeye yönelik önemli temel becerin kullanılmasını destekler.

Algoritma geliştirme görevleri öğrencilerin matematiksel kavramlara odaklanmasına yaptığı katkı bakımından da etkilidir. Tupouniua (2023), algoritma oluşturma görevleriyle meşgul olan öğrencilerin matematiksel kavramlara ilişkin kavramsal anlayışlarının geliştiğini ve matematiksel akıl yürütmenin karşı örnek bulma, doğrulama gibi temel süreçlerine yönelik beceriler kazandıklarını ifade etmektedir.

Araştırmalardan elde edilen sonuçlar algoritmik düşünmeye yönelik etkinliklerin matematik derslerinde kullanılmasının önemli faydalar sağlayacağına işaret etmektedir. Bu durumu dikkate alan Avustralya, Yeni Zelanda, Fransa gibi ülkeler matematik müfredatlarına hesaplamalı ve algoritmik düşünmeyi dahil etmişlerdir (Stephens, 2018). Ülkemizde de matematik eğitimi camiasının yukarıda bahsedilen araştırmaların sonuçlarını kapsamlı bir şekilde ele alıp değerlendirerek, matematik öğretim programlarında hesaplamalı ve algoritmik düşünmeye nasıl yer verilebileceğine dair bir tartışma yapması elzemdir.

KAYNAKLAR

Altun, M. (2008). İlköğretim ikinci kademe (6, 7 ve 8. sınıflarda) matematik öğretimi. Aktüel Yayınları: Bursa.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Paper presented at the Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada.

Cai, J. (2000). Mathematical Thinking Involved in U.S. and Chinese Students' Solving of Process-Constrained and Process-Open Problems. Mathematical Thinking and Learning, 2(4), 309-340.

Cai, J., & Brook, M. (2006). Looking back in problem solving. Mathematics Teaching, 196, 42–45.

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

Clements, D. H., Sarama, J., Yelland, N. J. & Glass, B. (2008). Learning and teaching geometry with computers in the elementary and middle school. M. K. Heid & G. W. Blume (Eds). Research on technology and the teaching and learning of mathematics içinde (Vol. 1, s. 109–154).

Çelik, D. (2016). Matematiksel düşünme. Bingölbali, E., Arslan, S., & Zembat, İ.Ö. (Ed). Matematik Eğitiminde Teoriler (s. 17-42). Ankara: Pegem Akademi.

English, L. D. (1998). Editorial Mathematical Reasoning: Nature, Form, and Development. Mathematical Cognition, 4(2), 81-83.

Genç, Z. & Karakuş, S. (2011). Tasarımla öğrenme: Eğitsel bilgisayar oyunları tasarımında Scratch kullanımı. 5th International Computer & Instructional Technologies Symposium (ICITS), (s. 981-987) Elazığ, Turkey.

Holyoak, K. J., & Morrison, R. G. (Eds.). (2012). The Oxford handbook of thinking and reasoning. Oxford University Press.

Knuth, D. E. (1985). Algorithmic thinking and mathematical thinking. The American Mathematical Monthly, 92(3), 170-181.

Knuth, D. E. (1997). The art of computer programming. Vol. 1: Fundamental algorithms. Reading, MA: Addison-Wesley.

Korkmaz, Ö. (2016). The Effect of Scratch-Based Game Activities on Students' Attitudes, Self-Efficacy and Academic Achievement. Online Submission, 8(1), 16-23.

Kwon, K., & Cheon, J. (2019). Exploring problem decomposition and program development through block-based programs. International Journal of Computer Science Education in Schools, 3(1), 1–14.

Lehmann, T. H. (2023a). How current perspectives on algorithmic thinking can be applied to students' engagement in algorithmatizing tasks. Mathematics Education Research Journal, 1-35.

Lehmann, T. H. (2023b). Using algorithmic thinking to design algorithms: The case of critical path analysis. The Journal of Mathematical Behavior, 71, 101079.

Lithner, J. (2008). A research framework for creative and imitative reasoning. Educational Studies in mathematics, 67, 255-276.

Lockwood, E., DeJarnette, A. F., Asay, A., & Thomas, M. (2016). Algorithmic Thinking: An Initial Characterization of Computational Thinking in Mathematics. North American Chapter of the International Group for the Psychology of Mathematics Education.

Moala, J. G. (2021). Creating algorithms by accounting for features of the solution: the case of pursuing maximum happiness. Mathematics Education Research Journal, 33(2), 263.

Moala, J. G., Yoon, C., & Kontorovich, I. (2019). Localized considerations and patching: accounting for persistent attributes of an algorithm on a contextualized graph theory task. The Journal of Mathematical Behavior, 55, 100704.

NCTM. (2000). Principles and standards for school mathematics. Reston, VA: National Council of Teachers of Mathematics.

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

Polya, G. (1966). Matematik ve Akla Yakın Muhakeme. 1. Cilt, 1. Kısım, Matematikte Endüksiyon ve Benzetme (Çev. O.Ş.İçen). İstanbul: Türk Matematik Derneği Yayınları

Polya, G. (1997). Nasıl Çözmeli (Çev. F. Halatçı). İstanbul: Sistem Yayıncılık.

Rasmussen, C., Zandieh, M., King, K., & Teppo, A. (2005). Advancing mathematical activity: A practiceoriented

view of advanced mathematical thinking. Mathematical Thinking and Learning, 7(1), 51–73.

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

Schoenfeld AH. Mathematical problem solving. Orlando (FL): Academic Press; 1985.

Schoenfeld, A. H. (1980). Heuristics in the classroom. In S. Krulik & R. E. Reys (Eds.), Problem solving in school mathematics 1980 yearbook içinde (s. 9–22). Reston, VA: National Council of Teachers of Mathematics.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. Educational research review, 22, 142-158.

Silver, E. A. (1997). Fostering creativity through instruction rich in mathematical problem solving and problem posing. Zdm, 3(29), 75-80.

Stein, M. K., Grover, B. W., & Henningsen, M. (1996). Building Student Capacity for Mathematical Thinking and Reasoning: An Analysis of Mathematical Tasks Used in Reform Classrooms. American Educational Research Journal, 33(2), 455-488.

Stephens, M. (2018). Embedding algorithmic thinking more clearly in the mathematics curriculum. Proceedings of ICMI study, 24, 483-490.

Stephens, M., & Kadijevich, D. M. (2020). Computational/algorithmic thinking. Encyclopedia of mathematics education (s. 117–123). Springer.

Tupouniua, J. G. (2023). What challenges emerge when students engage with algorithmatizing tasks? Journal of Pedagogical Research, 7(2), 93-107.

Umay, A. (2003). Matematiksel muhakeme yeteneği. Hacettepe Üniversitesi Eğitim Fakültesi Dergisi, 2003(24), 234-243.

Wetzel, S., Milicic, G., & Ludwig, M. (2020). Gifted students' use of computational thinking EDULEARN20 Proceedings, (s. 6936-6944). IATED.

Wing, J. M. (2006). Computational thinking. Commun. ACM, 49(3), 33–35.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717-3725.

MATEMATİK VE FEN BİLİMLERİ EĞİTİMİNDE YENİ YAKLAŞIMLAR

2023-III



EDİTÖR
PROF. DR. AYŞEGÜL GÖKHAN
DOÇ. DR. FATMA ERDOĞAN

EFEAKADEMİ YAYINLARI

MATEMATİK VE FEN BİLİMLERİ EĞİTİMİNDE YENİ YAKLAŞIMLAR
2023-III

Editör
Prof. Dr. Ayşegül GÖKHAN ORCID (0000-0001-5648-6343)
Doç. Dr. Fatma ERDOĞAN ORCID (0000-0002-4498-8634)

ISBN: 978-625-6600-26-3
E-ISBN: 978-625-6600-27-0
DOI: https://doi.org/10.59617/efepub202345
1. Baskı Ekim 2023

Bu eserin; yayın, satış ve kopyalama hakları EFE AKADEMİ'ye aittir.

Kütüphane Kartı

Matematik ve Fen Bilimleri Eğitiminde Yeni Yaklaşımlar 2023-III
GÖKHAN, Ayşegül & ERDOĞAN, Fatma
1. Basım 354 s., 160 x 235 mm. Kaynakça var, Dizin yok.
Türü : Özgün Bilimsel Kitap
Anahtar Kelimeler :
1. Araştırma 2. Eğitim 3. Fen bilimleri 4. Matematik 5. Öğrenme

Dizgi / Design	İsa Burak GÜNGÖR (burakgungor@efeakademi.com)
Kapak Tasarım / Cover Design	Gamze DUMLU (gamzedumlu.2710@gmail.com)
Sertifika No / Certificate No	49168
Matbaa Sertifika /Printing Certificate	49168
Efe Akademik Yayıncılık / Efe Akademik Publishing	Cağaloğlu Yokuşu Cemal Nadir Sokak Büyük Milas Han No: 24/125 Fatih/ İSTANBUL 0212 520 52 00 - www.efeakademi.com
Efe Akademik Yayıncılık Matbaa Adres: / Efe Akademik Publishing Printing Adress:	Cağaloğlu Yokuşu Cemal Nadir Sokak Büyük Milas Han No: 24/125 Fatih/ İSTANBUL 0212 520 52 00 - www.efeakademi.com

EFEAKADEMİ YAYINLARI