27. BÖLÜM

PROGRAMLAMA EĞİTİMİNDE YAPA-BOZA ÖĞRENME ETKİNLİKLERİ

Dr. Murat ÇINAR - Milli Eğitim Bakanlığı
Dr. Dilek DOĞAN - Ankara Üniversitesi
Prof. Dr. Hakan TÜZÜN - Hacettepe Üniversitesi

Özet

Programlama sürecinde yapa-boza (tinkering) öğrenme yaklaşımı bir algoritma ya da program parçası gibi bilgisayımsal bir model çerçevesinde en asgari düzeyde planlamayla kodlarda ufak değişiklikler yapmayı ve bunları test etmeyi kapsayan manipülatif bir etkinlik dizisidir. Yapıp-bozma hem acemilerin hem de uzmanların programlama yaparken gerçekleştirdikleri iteratif ve otantik bir süreçtir. Yapa-boza öğrenme olarak tanımlanan etkinlikler kurcalama, deneme-yanılma, geri bildirim mekanizmalarını bulma ve kullanma gibi etkinliklerin kombinasyonlarını içerir. Program kodlarıyla etkileşime girmeye ve onları manipüle etmeye dayalı olan yapa-boza yaklaşımı sorgulamayı teşvik eden üretken bir etkinliktir. Özünde tahmin et-sına süreçlerini barındıran yapa-boza yaklaşımı özellikle programlamaya yeni başlayan öğrenenler için kritik bir öneme sahiptir. Öğrenenler bu süreçte kod kümeleri içinde somutlaşan yapı ve kuralların nasıl işlev gördüğü kadar geliştirdikleri bilgisayımsal modellerin nasıl sonuçlandığını keşfetme fırsatı yakalar. Yapa-boza öğrenme bir problemin doğru cevabını bulmaktan ziyade ona ilişkin yeni anlayışlar keşfetmeye odaklanır. Bu bölümde öncelikle bilgisayımsal düşünme (computational thinking) kavramına yer verilerek, programlama eğitiminde karşılaşılan güçlükler adreslenmiştir. Daha sonra yapa-boza öğrenmenin özellikleri, bilgisayımsal düşünmeyle ilişkisi ve programlama eğitimindeki işlevine değinilerek, eğitsel düzenlemeler içerisinde nasıl yönlendirileceğine ve değerlendirileceğine ilişkin önerilere yer verilmiştir. Çalışma kapsamında ayrıca robotik etkinlikler ve blok tabanlı programlama uygulamaları yapa-boza öğrenme süreçlerini destekleme açısından incelenmiştir.

Anahtar Kelimeler: Bilgisayar bilimleri, programlama eğitimi, yapa-boza öğrenme, kodlama, robotik, eğitsel robot, manipülatif öğrenme

Hazırlık Soruları

1. Yapa-boza ya da kurcalayarak öğrenmenin özellikleri nelerdir?
2. Bilgisayımsal Düşünme (BD) etkinliklerinde yapa-boza öğrenmenin işlevi nedir?
3. Programlama eğitiminde karşılaşılan güçlükler nelerdir?
4. Bilgisayımsal Düşünmenin temel bileşenleri nelerdir?
5. Programlamaya yeni başlayanlar için programlama araçlarının seçiminde kullanılan "düşük zemin yüksek tavan" ilkesi ne anlama gelmektedir?

Giriş

Bilgi ve İletişim Teknolojilerinde (BİT) yaşanan gelişmeler sadece mesleki ve akademik alanı değil, aynı zamanda gündelik yaşamı da hemen her yönüyle etkilemektedir. Bu süreçte market alışverişi yapma, film izleme, bankacılık işlemlerini gerçekleştirme, yol/adres bilgisi sorgulama, ders alma gibi hayattaki birçok rutinin yeni bir biçim kazandığı görülmektedir. Gündelik hayatta gerçekleştirdiğimiz birçok etkinliğin arkasında artık bulut bilişim adı verilen dünya geneline yayılmış ağ sistemleri ve bilgisayarlar bulunmaktadır (Denning & Tedre, 2019). Dijitalleşme süreci devam ettikçe, bireylerin çevreyle etkileşimi ve çevreyi şekillendirme biçimi giderek düşüncelerini bilgisayarın işleteceği bir forma dönüştürme becerisine bağlı olmaya başlamıştır (Weintrop, Holbert, Horn & Wilensky, 2016).

Bilgisayımın (computation) hemen her alana yayıldığı günümüz toplumunda bireylerden artık teknolojiyi kullanmakla yetinmeyip, üretim/geliştirme süreçlerine aktif olarak katkı sağlaması beklenmektedir. Bunu başarmak için mevcut teknolojilere eleştirel bir gözle bakılması, teknolojinin insanların dünyayı algılama biçimini nasıl şekillendirdiğinin kavranması ve bilgisayımsal becerilerin işe koşulması gerektiği konusunda artan bir fikir birliği bulunmaktadır (Burke & Kafai, 2014). Bununla birlikte zaman zaman uzmanlar bile otantik yaşam problemlerine çözümler getirmek adına kuramsal bilgilerini pratiğe dökmekte ya da ilişkili bilgileri uygun durumlara yansıtmakta güçlükler yaşamaktadır.

Bilgisayımsal düşünme (Computational thinking) mantıksal becerilerin bilgisayar bilimleri kavramlarından yararlanılarak algoritmik bir dizi adımda sergilenmesini ifade eden bir problem çözme yaklaşımıdır. Tıpkı bilgisayarların çözümü oldukça zaman alıcı karmaşık hesaplama görevlerinden bizleri kurtarması gibi bilgisayımsal düşünme (BD) de fikirlerin algoritmik bir dizi aşama içerisinde kolayca operasyonel hale gelmesine katkı sağlamaktadır. Dahası BD geniş bir yelpazedeki problemler için geliştirilen entellektüel modellerin etkisini test etme fırsatı da vermektedir. Algoritmik süreçler sayısal olarak bir model tasarlamak ve problemler çözmek için iyi bir zemin oluşturmaktadır. Endüstrideki son gelişmelerle BD herkes için yaygın bir beceri olma yolunda ilerlemektedir. Dahası algoritmaları tanımanın 21. yy. okuryazarlığı için dördüncü "r" (reading, 'riting, 'rithmetic, ve 'rithms) olma yolunda küresel olarak ivme kazandığı görülmektedir (Grover & Pea, 2013). BD, çeşitli öğretimsel süreçler aracılığıyla öğrenenilebilecek ve geliştirilebilecek bir beceridir. Guzdial (2008) eğitimcilerin BD'yi herkes için erişilebilir hale getirmesi gerektiğini vurgulamaktadır. Peki, BD öğrenciler için nasıl daha erişilebilir hale getirilebilir? Başka bir ifadeyle BD becerileri hangi yollarla sergilenebilir? Araştırmacılar, BD'nin matematiksel ve mühendisliksel düşünme üzerine kurulu olduğunu ve bu düşünme biçimlerinin zengin mirasından yararlandığını

belirtse de BD'nin diğer düşünme becerilerini benzersiz bir yolla genişlettiğini kabul etmektedir (Wing, 2006). Bilgisayımsal problemlerle uğraşarak elde edilen bilgi ve deneyimler sadece bilgisayar bilimleri için değil doğa ve sağlık bilimlerinden sosyal bilimlere kadar hemen her alan için bir çerçeve sağlayabilir.

BD'nin geliştirilmesi amacıyla kullanılan çok sayıda etkinlik bulunmaktadır. Programlama, oyun oynama ve oyunlaştırma, oyun tasarımı, dijital hikâye anlatımı, bilgisayarsız bilgisayar bilimi etkinlikleri, örüntü bulma, kural oluşturma ve sınıflama (taksonomi), sistem modelleme ve simülasyon, robotik etkinlikler, üç-boyutlu tasarım ve modelleme, bulmaca çözme gibi etkinlikler bunlara örnek olarak gösterilebilir. Aslında BD için bilgisayarlar başta olmak üzere herhangi bir teknolojik araç ya da cihaz kullanımı şart değildir. Tersinden düşündüğümüzde bilgisayarların kullanımı BD süreçlerinin işletilmesini tek başına sağlamaz.

Programlama hem BD kavramının çıkış noktası hem de onun otantik olarak işe koşulduğu bir etkinliktir. Programlama etkinlikleri BD sürecinin yönlendirilmesi için gerekli tüm koşulları sağlama potansiyeline sahiptir. Birçok araştırmacıya göre BD becerisinin gelişimi için programlamadan daha iyi bir yol bulunmamaktadır. Programlama otantik yaşam problemlerinin adreslenmesi için bilgisayımsal süreçlerin işletilmesini zorunlu kılan önemli yeterliklerden birisi olarak kabul edilmektedir. Günümüzde programlama sadece bilgisayar bilimleri için bir ön koşul olmayıp, aynı zamanda farklı alanlarda problem çözme ve sistem tasarlama pratikleri için gerekli bir beceridir (Chao, 2016). Teknik açıdan programlama, problemler için çözümlerin planlanması, planların işletilmek üzere sözdizimsel olarak doğru komutlara dönüştürülmesi ve bu komutların çalıştırılması sonucunda ortaya çıkan sonuçların değerlendirilmesini gerektirmektedir (Chao, 2016). Bir diğer ifadeyle bir bilgisayar programı belli görevleri yerine getirmek için makine komutlarına dönüştürülmek üzere belirli bir programlama dili sözdizimine göre kodlanmış algoritmalardır. Öte yandan, programlama aracılığıyla bir problemin üstesinden gelmek için öncelikle problem detaylı olarak analiz edilmeli, ilgili problemin nasıl çözüleceği konusunda bir hipotez geliştirilmeli ve sonrasında ise geliştirilen hipotezi test etmek için kullanabilecek bir dizi kural inşa edilmelidir. Daha sonra çıktılar gözden geçirilmeli ve gerekirse çözüm yolları revize edilmelidir. Programlama yoluyla çözümlerini bilgisayımsal bir formda ifade etmeyi öğrenen öğrenciler, yalnızca soruları cevaplamakla kalmayıp, BD perspektifinden zorlukları görmeye, aynı zamanda yenilerini de üretmeye başlamaktadır. Bu aynı zamanda bilgisayımın özyinelemeli bir çıktısıdır.

Programlama öğretiminin hâlihazırdaki formu öğrenenlere bir programlama diline özgü sözdizimsel kuralların tanıtılması ve onlardan düşüncelerini bu forma sıkı sıkıya uyacak şekilde ifade etmelerinin istenmesidir. Bu durum aynı zaman-

da programlama görevlerinin/problemlerinin temsil edilme ve öğrencilere sunulma biçimini de şekillendirmektedir. Yapılan araştırmalar temel programlama ya da programlamaya giriş derslerinde öğrencilerin büyük bölümünün problemlerin ayrıştırılması/analizi, çözüm stratejilerinin geliştirilmesi, çözüm adımlarının planlanması ve belirli bir programlama diline kodlanmasında zorluk yaşadığını göstermektedir (de Raadt, 2007; Robins, Rountree & Rountree, 2003). Bu zorlukların nedeni ilk bakışta bilgisayımsal kavramların yeterince anlaşılmamasına atfedilse de aslında temelde öğrencilerin bilgisayımsal problemleri yeterince iyi temsil edecek zihinsel modeller geliştirememesinden ve çözüm önerilerini uygulanabilir planlara yani algoritmalara dökmek için gerekli stratejiden yoksun olmalarından kaynaklanmaktadır (Chao, 2016). Ayrıca programlamaya giriş derslerinde profesyonel yazılım geliştirme araçlarının kullanımı öğrencilerin problem çözme ve tasarım stratejileri geliştirmekten ziyade programlama dili özellikleri konusunda uzmanlaşmaya odaklanmasına neden olmaktadır. Sonuç olarak bu öğretim yöntemi programlama sürecinde öğrenenler tarafından algoritmaların geliştirilmesi ve test edilmesi, soruların üretilmesi ve bunlara çözüm aranması ve bu yolla dünyanın farklı bir bakış açısı aracılığıyla görülmesi için az sayıda fırsat sunmaktadır.

Papert öğrenme sürecinde öğretmenin rolünün "hazır bilgi vermek yerine, buluş şartlarını yaratmak" olduğunun altını çizmektedir. Programlama özü itibariyle yapıp bozarak en uygun çözümlere ulaşılan bir etkinlik dizisidir. Bu açıdan bakıldığında programlama öğrenenlerin yanlışlar yaptığı, bunları gerektiğinde hızlı bir biçimde geri aldığı ve en nihayetinde daha iyi bir sonuç arayışı içerisinde hatalardan öğrendiği bir süreçtir. Bu süreçte bireyler, başarısızlıklarından öğrendikçe çalışmalarını sadeleştirir ve probleme ilişkin daha derin anlayışlar geliştirir. Bir problem için geliştirilen çözüm adımları ya da algoritmalar sürekli olarak test edilir, gerektiğinde sadeleştirilir ve daha mükemmel hale getirilir. Öğrenenler bu süreçte temel problem durumuyla meşgul olurken sıklıkla kendi yarattığı sorunlara cevap arar ve problem içinde yeni problemler keşfeder. Bu açıdan bakıldığında programlama aslında iteratif ve özyinelemeli bir bilişsel çabadır.

Özetle, programlama öğrenenin sınırsız sayıda yanlış yapmasına, bu yanlışların sonuçlarını gözlemlemesine ve gerektiğinde hataları hızlı bir biçimde düzenlemesine olanak tanımalıdır. Programlamada yapa-boza öğrenme anlayışı öğrenenlerin sürekli olarak denemesi, sonuçları gözlemlemesi, bilgiyle kanıtlar arasında dinamik ilişkiler kurması ve üstbilişsel bir farkındalık içerisinde düşünceleri üzerinde derinlemesine anlayışlar geliştirmesi için eşsiz olanaklar sağlamaktadır. Bu kapsamda bu bölümde BD kavramı açıklanarak, programlama eğitiminde karşılaşılan güçlükler adreslenmektedir. Daha sonra yapa-boza öğrenme kavramı tanıtılmakta ve programlama eğitimindeki işlevine değinilmektedir. Ek olarak, robo-

tik ve blok tabanlı programlama etkinlikleri yapa-boza öğrenme süreci açısından incelenmektedir. Son olarak, yapa-boza etkinliklerin nasıl değerlendirileceği ve eğitsel düzenlemeler içerisinde nasıl yönlendirileceğine ilişkin önerilere yer verilmektedir.

Bilgisayımsal Düşünme (BD)

Papert (1980) bilgisayarların güçlü fikirlerin ve kültürel değişim tohumlarının taşıyıcısı olabileceğini, fen bilimleriyle sosyal bilimleri ayıran geleneksel çizgilerin ötesine geçerek bilgiyle (knowledge) yeni ilişkiler kurulmasına yardımcı olabileceğini ileri sürmüştür. Bu düşünce aslında bilgisayar bilimlerinin sadece uzmanlar için değil, herkes için kullanılabilir evrensel bir beceri seti olduğunun altını çizmektedir. Yani farklı disiplinlerdeki araştırmacılar da bilgisayar bilimcileri tarafından önemli görülen becerileri kullanarak önemli yenilikleri keşfedebilir. Bunun için algoritmalar çerçevesinde analiz edilebilecek davranışların farkına varılması gerekmektedir. BD'nin çıkış noktası tam da budur. BD, bilgisayar bilimi kavramlarını otantik yaşam problemlerinin çözümünde kullanabilmek için gerekli olan bilgi, beceri ve tutumlara sahip olma şeklinde tanımlanabilir. Bununla birlikte BD'nin genel kabul gören bir tanımı bulunmamaktadır. Alanyazın incelendiğinde BD'ye ilişkin çok çeşitli tanımlamaların yapıldığı görülmektedir. Bu tanımların büyük bölümü bilgisayar bilimlerindeki kavramların irdelenmesiyle ya da program yazma süreçlerinden ilham alınarak geliştirilmiştir. BD'ye ilişkin yapılan farklı tanımlamalar onun bir disiplin olarak eğitimciler tarafından açık bir biçimde anlaşılmasında ve dolayısıyla kullanılmasında sıkıntılara yol açmaktadır (Grover & Pea, 2013). Dolayısıyla BD'nin etkili bir biçimde nasıl öğretileceği ve nasıl değerlendirileceği gibi sorular halen güncelliğini korumaktadır (Kalelioğlu, Gülbahar & Kukul, 2016).

BD'ye ilişkin ilk tanımlamalar BD'nin ne olduğu üzerinedir. BD kavramının popülerleşmesini sağlayan Wing (2006) BD'yi bilgisayar bilimleri için temel kavramlardan yararlanarak problemler çözmek, sistemler tasarlamak ve insan davranışlarını anlamak olarak kavramsallaştırmıştır. Wing (2006) BD'nin programlamadan ziyade bir kavramsallaştırma biçimi olduğunu, bir işlemin belirtilen sırada izlenmesinden ziyade izlenecek sürecin bireylerin düşünceleriyle şekilleneceği temel bir beceri olduğunu ifade etmektedir. Ek olarak, bilgisayar gibi düşünme çabasından ziyade onu programlayanın bakış açısının kazanılmasıyla ilgili olduğunu, temellerini matematiksel ve mühendisliksel düşünmeden aldığını, üründen ziyade düşüncelere odaklandığını ve sadece programcılar için değil herkes için işlevsel olabileceğini belirtmektedir. Denning (2009) BD'yi, giriş verilerini algoritmik olarak kontrol edilen çıktılara dönüştürme becerisi olarak tanımlamıştır.

BD üzerine yapılan son tanımlamalarda BD'nin işlevine daha çok vurgu yapıldığı görülmektedir. Wing (2011) BD tanımını problemlere ilişkin çözümlerin bir bilgi-işleme ajanı tarafından etkin bir biçimde yerine getirebilecek şekilde formüle edilmesinin altında yatan düşünme süreçleri şeklinde yeniden organize etmiştir. Aho (2012) da benzer biçimde BD'nin problemlerin ve çözümlerinin algoritmik adımlar şeklinde temsilini içeren düşünme süreçleri olduğunu belirtmiştir. Kalelioğlu, Gülbahar, Akçay ve Doğan'a (2014) göre ise BD, varolan bilgiler yoluyla eleştirel düşünme süreçlerinin uygulanması ve karmaşık teknolojik problemlerin çözümü için bilgisayar teknolojisinden yararlanılmasıdır. Uluslararası Teknoloji Eğitimi Birliği (ISTE) BD'yi otomasyon araçlarıyla algoritmik düşünme ve simülasyonların kullanımıyla veri gösterimi olarak tanımlamıştır (Lye & Koh, 2014). ISTE ve Bilgisayar Bilimi Öğretmenleri Birliği (CSTA) BD'ye ilişkin operasyonel bir tanımlama geliştirmek adına yükseköğretim, endüstri ve K12 temsilcileriyle gerçekleştirdiği işbirliği neticesinde BD'nin altı temel karakteristik özelliğini şu şekilde belirlemiştir (ISTE & CSTA, 2011):

1. Problemlerin bilgisayar ve diğer araçlar yardımıyla çözülmesini sağlayacak şekilde formüle edilmesi
2. Verilerin mantıksal organizasyonu ve analizi
3. Verilerin model ve simülasyon gibi soyutlamalar yoluyla gösterimi
4. Algoritmik düşünme yoluyla çözümlerin bir dizi sıralı adımda otomatik olarak işletilmesi
5. Çözüm aşamalarının ve kaynakların en verimli ve etkili şekilde bir araya getirilmesini sağlamak amacıyla olası çözümlerin belirlenmesi, analiz edilmesi ve uygulanması
6. Problem çözme süreçlerinin genelleştirilmesi ve çok çeşitli problemlere transfer edilmesi

ISTE ve CSTA (2011) tarafından BD becerilerinin kazanımını destekleyici eğilim ve tutumları da adreslemiştir. Bunlar: karmaşıklıkla başa çıkma konusunda güven, zor problemlerle çalışmada kararlılık, belirsizlik toleransı, açık uçlu problemlerle başa çıkma yeteneği ve ortak bir hedef ya da çözüme ulaşmak için başkalarıyla iletişim kurma ve çalışma yeteneğidir. Öğretim programlarıyla bütünleştirilmesi ve öğrenme sürecinde daha izlenebilir olması için BD'nin ne olduğundan ziyade hangi bileşenlerden oluştuğunun belirlenmesi gerekmektedir. Bu bileşenler aynı zamanda eğitimciler için öğretimsel hedeflerin ve değerlendirme ölçütlerinin geliştirilmesinde referans noktalarıdır. BD'nin yaygın olarak kabul edilen bileşenleri soyutlaştırma, ayrıştırma (modülerleştirme), örüntü yakalama, kural oluşturma ve genelleme, sistematik bilgi işleme, sembol sistemler ve temsiller, akış kont-

rolünün algoritmik kavramları, iteratif, öz-yinelemeli ve paralel düşünme, koşullu mantık, verimlilik ve performans kısıtlamaları, hata ayıklama ve sistematik hata denetimidir (Grover & Pea, 2013). Soyutlama BD'nin en temel bileşeni olup onu diğer düşünme biçimlerinden ayıran en belirleyici özelliğidir. Soyutlama sürecinde hangi detayların dikkate alınacağına ya da hangilerinin göz ardı edileceğine karar vermek BD'nin özünü oluşturmaktadır (Wing, 2008). Bilgisayar Bilimleri oldukça sembolik olan genel nitelikli soyutlamalar yönünden zengin referanslar barındırmaktadır. Bu soyutlamalar matematik ve fen bilimlerindeki eşdeğerlerinden daha karmaşıktır. Bilgisayar Bilimlerindeki soyutlamalar genellikle matematikteki emsalleri gibi kolayca algılanabilen cebirsel temellere sahip değildir (Wing, 2008). Dahası bilgisayar bilimlerinde kullanılan veri türleri ve değişkenler diğer disiplinlerdekilerden farklı bir kavramsallaştırma gerektirebilmektedir. Örneğin bir bileşen yığını bilgisayımda kullanılan oldukça soyut bir veri türüdür.

BD'nin ardındaki kuramsal çerçeve köklerini yapılandırmacılıktan alan yapıcı ya da inşacı (constructionist) öğrenme anlayışına dayanmaktadır. Yapıcı öğrenme anlayışı çocuğun bilgiyi deneyim yoluyla inşa ettiğini savunup, eğitimde yaparak öğrenme yaklaşımının altını çizmektedir. Papert'ın yapıcılık/kurmacılık (constructionism) yaklaşımına göre, öğrenciler etkileşimli ve gerçekçi nesneler inşa etme sürecinde aktif olarak yer aldığında öğrenme süreci daha etkili olmaktadır. Bu anlayışa göre öğrenciler bir öğrenme topluluğunda kendi projelerini oluşturduklarında anlamlı ve derin öğrenme gerçekleşmektedir.

Teknoloji ve insan arasında çift yönlü bir etkileşim bulunmaktadır. Yani insanlar teknoloji geliştirdikçe, teknoloji de insanları değiştirmektedir. Teknoloji karşılaştığımız birçok soruyu ve sorunu çözmek için bize yeni yollar bulma ve nasıl düşündüğümüzü etkilemek suretiyle yeni düşünce kalıpları geliştirme ortamı sağlamaktadır. Papert (1980) öğrenme sürecinde kurulan zihinsel modellerin bilgiler arasındaki ilişkilerin daha iyi kavranmasında, dahası bu ilişkilerin üzerinde hâkimiyet kurularak yeni bilgi ediniminde kolaylaştırıcı bir unsur olabileceğini belirtmiştir. Yani bu modeller mevcut bilgiyi yeni bir nesneye taşımanın güçlü bir yoludur. Bu zihinsel modeller sadece bilişsel olarak işlev görmekle kalmayıp, aynı zamanda öğrenmenin pozitif bir duygusal tonla donatılmasına yardımcı olmaktadır. BD uygun temsille problemleri anlamayı, birden fazla soyutlama seviyesinde akıl yürütmeyi ve otomatikleştirilmiş çözümler geliştirmeyi kapsayan bir dizi düşünme kalıplarından oluşmaktadır (Wing, 2006). İster mühendislik, ister sağlık isterse sanat ve sosyal bilimler alanında olsun tüm alanlar bilgisayar bilimlerinden etkilenmektedir. Farklı disiplinlerdeki araştırmacılar bilgisayımsal yöntemler sayesinde yeni ve eski problemlere, araştırma tasarımı ve yorumlamasındaki yeniliklere nasıl yaklaşılacağı konusunda yeni perspektifler kazanmaktadır.

BD'nin ortaya çıkmasına ilham veren şey de bu düşüncedir. BD matematik, fen, mühendislik, teknoloji ve sanat gibi farklı disiplinler için fikirlerin sentezlenmesinde yararlı olacak yeni bir düşünme biçimini teşvik etmektedir. BD sadece bilgisayar bilimcileri tarafından değil herkes tarafından dikkate alınması gereken temel bir beceridir (diSessa, 2001; Guzdial, 2008; Wing, 2006). Örneğin programlama matematikteki yöntem ve süreçleri anlamak için benzersiz bir öğrenme olanağı sunmaktadır. Aslında BD'nin temelleri de çocukların ekran ortasında bulunan bir kaplumbağayı bilgisayar kodlarıyla hareket ettirerek formal geometri kavramlarının biçimsel özellikleriyle ilgili aşinalık kazandığı bir bilgisayar programı olan Logo'ya dayanmaktadır (Papert, 1980). Öğrenciler BD sayesinde kurallardan yola çıkarak örüntüler oluşturabilir, mevcut örüntüleri tanıyıp sınıflayabilir, karmaşık denklemleri çözebilir ve geometrik yapılara ilişkin farklı kavrayışlar edinebilir. Modelleme, simülasyon ve baskı araçları aracılığıyla eğik bir düzlem inşa edebilir; hatta çeşitli hesaplama araçlarıyla bu düzlemdeki kuvvet kazancına ilişkin cebirsel hesaplamalar gerçekleştirebilir. Böylelikle fen ve mühendislik kavramlarını somut ögeler aracılığıyla otantik dünya düzenlemeleri içerisinde deneyimleme fırsatı yakalayabilir. Somut robotikler aracılığıyla çevresel ve toplumsal sorunlara çözüm üretebilir. Yaratıcı kodlama etkinlikleri aracılığıyla sanat eserleri oluşturabilir.

BD becerilerini kullanan bireyler birçok gündelik etkinliği etkili bir şekilde gerçekleştirmektedir. Bununla birlikte bireyin bilgisayımsal becerilerini geliştirmek ve sergilemek için fırsatların yetersizliği bilgisayımsal becerilerin transferinde güçlükler yaratmaktadır. Programlama, BD becerilerinin sergilenmesi için son derece uygun bir etkinliktir. Alanyazın incelendiğinde BD becerilerinin geliştirilmesi için yaygın olarak kullanılan stratejinin programlama olduğu görülmektedir. Programlama etkinlikleri bilgisayar bilimleri ile diğer disiplinler arasında bir köprü olan BD'nin iskeletini oluşturmaktadır. Bununla birlikte, programlama dersleri sıklıkla öğrenciler tarafından oldukça prosedürel, dolayısıyla sıkıcı bulunmaktadır. Programlama derslerinde karşılaşılan yüksek ders bırakma oranları da bu durumun bir göstergesi niteliğindedir.

Programlama Eğitiminde Karşılaşılan Güçlükler

BD programlama ya da kodlamadan ibaret değildir, fakat iyi planlanmış programlama etkinlikleri verimli BD çıktıları elde edilmesini sağlayabilir (García-Peñalvo & Mendes, 2018). Öte yandan, programlama hangi yaştan olursa olsun tecrübesiz öğrenenler için zorlu bir görevdir. Problemlere çözümler üretmeye ve programların nasıl çalışacağına ilişkin uygun zihinsel modeller geliştirmeye ek olarak programlamaya yeni başlayanlar, katı sözdizim kurallarına uymak ve ço-

ğunlukla kendilerine anlamlı gelmeyen hatta çoğu zaman kafa karışıklığına yol açan komut setini öğrenmek zorunda kalmaktadır (Kelleher & Pausch, 2005). Aynı anda tüm bu zorluklara meydan okumak zorunda kalmak programlamaya yeni başlayanlar için sıklıkla programlama sürecinin ezber yoluyla mekanikleşmesine yol açmaktadır. Bu durum programlama mantığının gelişimini sekteye uğratmaktadır. Özellikle sözdizimine dayalı genel amaçlı programlama dilleri yeni başlayanlar için dik bir öğrenme eşiği oluşturmaktadır. Yeni başlayanlar için kullanılacak programlama araçları hem temel bilgisayımsal kavramların kazanımını sağlamalı hem de genel amaçlı programlama dillerine geçişi kolaylaştırmalıdır. Blok tabanlı programlama ortamları sözdizimi hatalarını en asgari düzeye indirecek biçimde görsel kod bloklarının birbirine kenetlenmesiyle programlama yapılmasına olanak sağlamaktadır. Düşük zemin-yüksek tavan ilkesiyle, yani hem sınırlı programlama bilgisine sahip olanların bile zorlanmadan kullanabileceği hem de artan programlama becerilerine bağlı olarak karmaşık ve büyük ölçekli programların hazırlanmasına olanak tanıyacak biçimde tasarlanan bu ortamlar yeni başlayanlar için göz korkutmayan bir programlama deneyimine aracılık etmektedir (Berland, Martin, Benton, Smith & Davis, 2013).

BD'nin başta K12 olmak üzere her seviyedeki öğretim programıyla bütünleştirilmesi için ileri sürülen temel argümanlar bu düşüncenin iyi-yapılandırılmamış problemler yoluyla öğrencilere nasıl düşüneceklerini, verileri nasıl yorumlayacaklarını ve bilgisayarları kullanarak dış dünyayla nasıl iletişim kuracaklarını öğrenmelerine yardımcı olabileceği düşüncesidir (Lee, Martin & Apone, 2014). Bununla birlikte öğrenenler çoğunlukla soyut yapıları mantıksal bir düzende bir araya getirerek oluşturdukları algoritmalar için otantik hayatla ilişkilendirilebilecek referanslar bulamamaktadır (Ramadhan, 2000). Belirli bir programlama diline özgü kodları ve sözdizim kurallarını bilme BD için herhangi bir temel sağlamaz. Programcı gibi düşünmek, kullanıcının bakış açısını anlamayı ve onun ihtiyaçlarını en iyi şekilde karşılamayı gerektirir.

Programlama etkinlikleri çoğunlukla ekran bağımlı olarak gerçekleştirilmektedir. Bu durum öğrenenlerin akranlarıyla olan etkileşim ve işbirliğini sınırlamaktadır. Birden fazla öğrenen ekran karşısında ortak çalışma yapmaya başladığında ise genellikle bazı öğrenenler pasif kalmakta ya da diğer öğrenenlerle bilgisayar bileşenlerinin kontrolü konusunda pazarlığa girişmektedir (Burleson vd., 2018). BD için çok çeşitli eğitim teknolojilerine odaklanılmasına rağmen, bunların büyük çoğunluğu hala bir çocuk başına bir cihaz paradigmasına dayanmaktadır. Özellikle düşük yaş gruplarındakiler olmak üzere öğrenenlerin genellikle oyun oynayarak, sosyalleşerek ve çevrelerini manipüle ederek öğrendikleri bilinmektedir (Resnick, 2017; Wyeth, 2008; Zuckerman & Resnick, 2003). Bilgisayımsal becerileri

geliştirmek için programlar, simülasyonlar, hikâyeler, robotikler gibi bilgisayımsal ürünleri tasarlama ve üretme öğrencilere bilişsel, epistemik ve sosyal süreçlere katılma fırsatı verebilir (Farris & Sengupta, 2016). Ayrıca öğrenciler için tekrarlı girişimlerin sonuçlarının hızlı ve kolay bir biçimde anlaşılmasını sağlayan, öğrenen manipülasyonuna açık otantik uygulamalar sorgulamaya dayalı öğrenmenin üretken bir formunu teşvik edebilir (Wagh, Cook-Whitt & Wilensky, 2017).

Yapa-Boza Öğrenme / Kurcalayarak Öğrenme (Tinkering/Bricolage)

Bilgisayar bilimi ve diğer birçok disiplinde sistemler ve problemler genellikle büyük ve karmaşık süreçleri içermektedir. Karmaşık olan herhangi bir şeyde uzmanlaşmak ve düşünceleri etkili bir şekilde harekete geçirmek için yaklaşımlara ihtiyaç duyulmaktadır (Kalelioğlu, Gülbahar & Doğan, 2017). BD yaklaşımlarından birisi olan yapıp-bozma bir problem veya proje kapsamında bireylerin sürekli olarak fikirlerini denediği, bunlar üzerinde iyileştirmeler yaptığı ve sürekli yeni olasılıkları düşündüğü eğlenceli, açıklayıcı ve yinelemeli bir süreç olarak tanımlanmaktadır (Resnick & Rosenbaum, 2013). Yapa-boza öğrenme Maker hareketi bünyesinde kendin yap (Do-it-yourself-DIY) ve diğerleriyle yap (Do-it-with-others-DIWO) kültürlerinin ortak ürünü olarak dijital ve fiziksel kaynakların kullanılmasıyla gerçekleştirilen ve el yapımı uygulamalarla ortaya çıkan bir kavramdır (Brahms, 2014). Kökeni Fransızca "bricoler" eylemine dayanan bu kavram "tinkering", "tailoring" ve "bricolage" kelimeleriyle de ifade edilmektedir (Lévi-Strauss, 1962).

Yapa-boza öğrenme doğrudan deneyim, deney ve keşif yoluyla sorunlara yaklaşmanın ve sorunları çözmenin eğlenceli bir yolu olan zihinsel bir süreç, oyun ve öğrenmeyi kapsayan, sosyallik ve yaratıcılığı birleştiren benzersiz etkinlikler olarak da tanımlanmaktadır (Martinez & Stager, 2013). Yapa-boza öğrenme Berland vd. (2013) tarafından aşağıdaki gibi de tanımlanmaktadır:

- Belirli bir problem alanında kurama dayanmayan ve amaç-yönelimli olmayan bir keşif sürecidir.
- Bir ürünün ya da programın çalışan bir sürümünü, onun çalışması için yapılan her şeyi mutlaka anlayamaya gerek duymaksızın tasarlamaktır.
- Eğlenceli bir deneydir.
- Küçük değişiklikleri test etme sürecidir.
- Deneme yanılma sürecidir.
- Brikolaj (Bricolage) olarak da adlandırılan programcıyla program arasında bir etkileşim, hatalarla gezinme ve bir adım ötesinden biraz daha fazlasını planlama sürecidir. Brikolaj kendi başına da anlamlı olan öge-

lerin yeni bir bağlam çerçevesinde ufak adımlarla birleştirilmesi ve bu parçalardan yeni şeyler üretilmesi olarak da ifade edilmektedir. Bilimsel bir sorgulamada ise brikolaj uzak planlamalar yapmaksızın nesnelerle doğrudan temasa geçme anlamında kullanılmaktadır(Clegg & Kolodner, 2007).

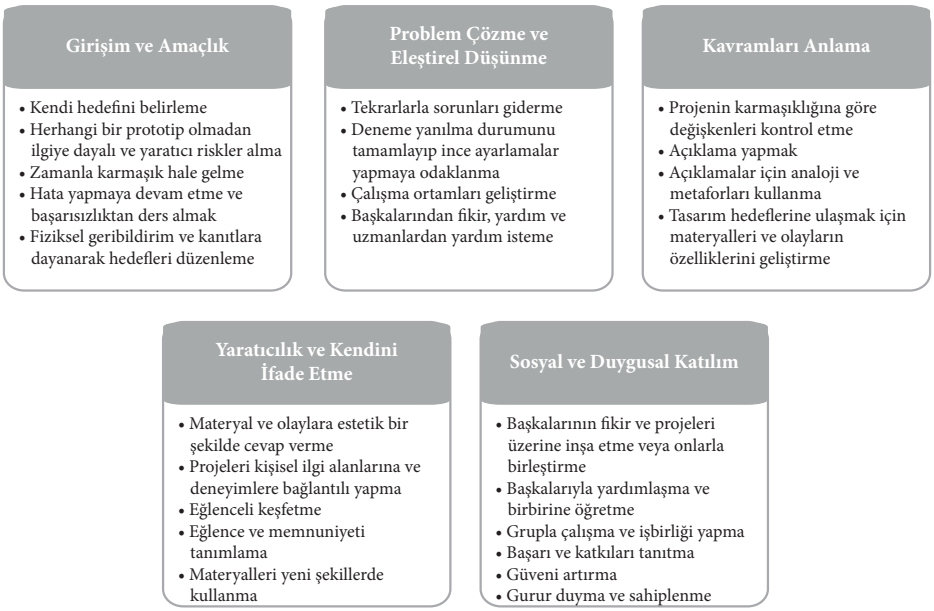
- Fussing olarak da tanımlanan bir öğrenme etkinliğiyle meşgul olurken öğrencilerin eylem ve fikirlerinde tesadüfen gerçekleşen değişkenliklerden doğan ve gelişmiş çıktılara ya da ürünlere ön ayak olan küçük bir değişim sürecidir.
- Tıpkı Vikipedi (Wikipedia) aramalarında olduğu gibi performans anında gerçekleştirilen bir etkinliktir.

Yapa-boza eylemler problemlerin çözümünde doğaçlama gelişen yaratıcı çözümleri içerdiği için fabrikasyon ya da standart üretimlerden farklı bir süreçtir (Bevan, Petrich & Wilkinson, 2014). Bu süreçte bireyler bir dizi materyal veya araç oluşturarak ya da bu materyalleri veya araçları işler hâle getirerek öğrenirler (Papert, 1980). Bireyler yeni bilgiler edindikçe ve yeni çözümler geliştirdikçe projenin ya da problemin amacı değişebilmektedir. Ulaşılacak son nokta ya da ortaya çıkacak ürün bilinmediğinden yapa-boza öğrenme bilim ve mühendisliğin yaratıcı uygulamaları ve keşfedici yönüyle paralellik gösteren bir süreçtir (Bevan vd., 2014). Yapa-boza etkinlikler çok çeşitli materyallerin ve araçların kullanılabileceği fiziksel bir etkinliktir. Öğrenenler bu etkinliklerde materyallerle ve araçlarla oynamaya teşvik edilmektedir. Başka bir ifadeyle öğrenenler kendi projelerini oluşturacakları, onlar için olumlu olacak bir deneyime davet edilmektedir. Deneyimin yaratıcı doğası, öğrenenleri yeni bir projenin, amacın, fikrin geliştirilmesi ve yenilik ruhunun sürdürülmesi için cesaretlendirir. Deneyimin duyusal ve kişiye has doğası ise öğrenenlerin el yapımı fiziksel ürünlerin yapıldığı dijital ve çevrimiçi ortamlarda kaybetme ya da başarısız olma riskini alabilmelerine yardımcı olmaktadır (Harris, Xanthoudaki & Winterbottom, 2018). Bu süreçte çocuklar oyuna dayalı keşfetme ve deneyimleme; gençler ve yetişkinler ise deneme-yanılma ve geliştirme süreciyle amaçlı olarak keşfetme ve birşeyler oluşturmayla öğrenme sürecini gerçekleştirmektedir (Barefoot Computing at School, 2014). Yapa-boza öğrenmede bireyler belli bir amaç gözetmeksizin yola çıkarlar. Bu süreçte bireyler sürekli yeni şeyleri keşfetmeye ve yeni şeyleri denemeye çabalarlar. Yapa-boza eylemi bireylerin yeni durumlarla karşılaştıkça önceki durumları nasıl geliştirdiğini, nasıl uyarladığını ve yinelediğini anlamaya yönelik stratejileri kapsamaktadır (Resnick & Rosenbaum, 2013).

Yapa-boza öğrenmeye dayanak oluşturacak çeşitli öğretim yaklaşımları bulunmaktadır. Yapa-boza öğrenme, öğrenenin bilimsel fikirleri ve olayları kendi

anlayışına göre inşa etmesini sağladığı için oldukça yapıcı bir yaklaşımdır. Öğrenen yeni bir şeyler oluşturma sürecinde önceki bilgileri kullanarak, var olan bilgiler ve yeni kavramlar arasında bağlantı kurarak varolan bilişsel modellerini biraraya getirip yeni bir anlayış inşa etmektedir. Öğrenenler bu süreci bireysel olarak planlamakta, kendi tasarımlarını yapmakta, test etmekte ve yeniden düzenlemektedir. Bununla birlikte yapıp-bozma STEM ve sorgulamaya dayalı yaklaşımlarla da yakından ilişkilidir. Yapa-boza etkinlikler birden fazla sonucu ve beklenmedik durumları olabilecek öğrenme yolculuğunda öğrenenleri kendi sorularını ve mücadele unsurlarını geliştirmeleri, fikirlerini tartışmaları, süreç boyunca karşılaştıkları sorunları fark etmeleri ve bu sorunları dile getirmeleri, çözüm aramaları, süreçteki ilerlemeyi değerlendirmeleri, hipotezlerini oluşturmaları, test etmeleri ve yeniden test etmeleri için zorlamaktadır. Yapa-boza etkinlikler fiziksel, pratik, sürükleyici ve yaratıcı doğası gereği sorgulamaya dayalı ve yapıcı diğer etkinliklerden ayırt edilebilmektedir (Harris vd., 2018).

Yapa-boza eylemlerde öğrenme süreciyle ilgili katılım (engagement), girişim ve amaçlılık (initiative and intentionality), sosyal destek (social scaffolding) ve anlama sürecinin geliştirilmesi (development of understanding) olmak üzere 4 bileşen tanımlanmaktadır. Katılım sürecinde öğrenenler etkinlikler için zaman harcayarak etkinliklere katılmaya çalışarak motivasyonlarını arttırmaya çabalamaktadır. Girişim ve amaçlılık sürecinde öğrenenler kendi amaçlarını belirler, araştırır ve geribildirimlere göre cevaplar oluşturur. Bu süreçte öğrenenler amaca ulaşmakta ısrarcı olup bilgiye dayanan riskler alır. Sosyal destek süreci öğrenenlerin problem çözümünde yardım istemesini, yeni fikirlerden veya yaklaşımlardan ilham almasını ve başkalarının yapmış olduğu çalışmalarla fiziksel bağlantı kurmasını içerir. Anlama sürecinin geliştirilmesinde ise öğrenenler bir kavramı etki ve ifade yoluyla tanımlamaya çabalar, belirlediği strateji ve araçlara göre sonuçları sunar, bilgiyi uygular ve anlamaya çabalar (Bevan vd., 2014). Harris vd. (2018) tarafından yapa-boza etkinliklerinin 5 öğrenme boyutu olduğu ifade edilmektedir (Şekil 27.1).



Şekil 27.1. Yapa-boza etkinliklerinin öğrenme boyutu (Harris vd., 2018)

Özdemir (2019) yeniliklerin ve başarılı girişimlerin arkasında bireylerin gözlemlerinin ve fikirlerinin peşi sıra gerçekleştirdiği işbirlikli yapa-boza eylemlerinin olduğunu vurgulamaktadır. Yapa-boza eylemlere verilebilecek iyi örneklerden birisi varolan bilgisayar kodlarının üzerinde yapılan değişikliklerdir. Bir program çalıştırıldığında ya da derlendiğinde sonuç somut olarak gözlemlenir. Bu durum öğrenenlerin programda yapılan değişiklikler ve sonuç arasındaki bağlantıyı kolayca görmesini sağlarken, program çalışmadığı zaman bir hatanın meydana geldiğini anlamalarına da yardımcı olmaktadır (Kotsopoulos vd., 2017).

Programlama Eğitiminde Yapa-Boza Öğrenmenin İşlevi

Yapa-boza etkinlikler hem tecrübesizlerin hem de uzmanların programlama sürecinde sıklıkla gerçekleştirdiği otantik bir etkinliktir. Programlama hatalara oldukça açık bir süreçtir. Uzmanlık seviyesi hangi düzeyde olursa olsun neredeyse hiçbir programcı ilk seferde kusursuz bir program yazamaz. Programcı yazdığı kodları ya da yaptığı arayüz tasarımını sürekli olarak daha iyi hâle getirmeye çabalar. Bu açıdan bakıldığında programlamayı, yapılan hataların ya da girişimlerin sonuçlarına göre şekillenen bir etkinlik olarak nitelemek yanlış olmaz. Programlamada yapa-boza etkinlikler bir bilgisayar programının oluşturulması ya da değiştirilmesi sürecinde ayrıntılı bir plana sahip olmamayı ya da buna ihtiyaç duymamayı betimler. Yapa-boza öğrenme sürecinde kurcalama, deneme-yanılma, geri

bildirim mekanizmalarını bulma ve kullanma kritik öneme sahiptir. Öğrenciler geribildirimlere göre bir sonraki eylemlerini ve bunların sonucunda ortaya çıkan ürünleri yeniden şekillendirir. Yapa-boza öğrenmenin iteratif ve değişken doğası özellikle yeni başlayanların programlamayı öğrenmesinde yardımcı olabilir. Yapa-boza öğrenmede yer alan tahmin et-sına girişimleri programlamadaki temel ögelerin keşfini destekler (Burke & Kafai, 2010; Perkins, Hancock, Hobbs, Martin & Simmons, 1986; Resnick vd., 2009). Yapa-boza öğrenme programlama sürecinde önemli bir yere sahiptir. Programlamaya yeni başlayan öğrenenler programlama sürecinde yapıp bozdukça programlarında daha az düzeltmeye gittikleri görülmüştür (Berland vd., 2013). Kısa süreli yapıp bozmalar programlamanın son aşaması değildir. Öğrenciler kodları anladıkça yapıp bozmalar daha da azalmakta ve sadeleştirme (kod üzerinde düşünme) safhası üzerinde daha çok durulmaktadır. Başka bir ifadeyle programlama, yapıp bozmaya kesin bir şekilde olanak tanıyan değerli bir öğrenme etkinliğidir (Berland vd., 2013).

Yeni başlayan programcılar yazdıkları program kodlarıyla etkileşime girerek, onları sürekli olarak manipüle eder. Yapılan her manipülasyonun sonucunun gözlemlenmesi öğrencileri yaptıklarıyla sonuçları arasında bir ilişki kurmaya teşvik eder. Bu açıdan bakıldığında yapa-boza eylemlerde kurulan sebep sonuç bağlantıları öğrenenlerin mantıksal akıl yürütme becerilerinin gelişimine önemli katkılar sağlar. Yapa-boza öğrenme keşif ve bilgi edinimini kolaylaştırmasının yanında öğrenenleri diğer yollarla da destekleyebilir. Örnek olarak, yapa-boza öğrenme iyi bir programcı olmak için gerekli planlama becerilerinin gelişimini ve yeni öğrenme yörüngelerinin keşfini destekleyebilir. Yapa-boza etkinlikleri özünde birşeyleri özgürce denemeyi ifade etmektedir. Bu aslında günlük hayatta oldukça sık kullanılan bir öğrenme biçimidir. Yeni bir cihaz (örneğin akıllı telefon) aldığımızı düşünelim. Çoğumuz ürün kullanım kılavuzunun sıkıcı prosedürel talimatlarını okumak yerine, ürünü kurcalar ve özelliklerini test etmeye başlarız. Böylelikle hangi özelliklerin işimize yarayıp hangisinin yaramayacağını sorgularız. Bu süreç önceden öngördüğümüz sonuçlara bilinçli bir şekilde ulaşma çabasından farklıdır. Yapa-boza eylemi sürpriz sonuçlara oldukça açıktır. Bu aynı zamanda öğrenenlerin hatalara karşı toleransının yükselmesine katkı sağlar. Bu onların risk almalarına ve sıra dışı çözümleri çekinmeden uygulamalarına olanak tanır. Yapa-boza öğrenmenin bağımsız öğrenme becerilerinin gelişiminde de olumlu etkisi bulunmaktadır. Bu süreçte öğrenciler başkalarının rehberliğine gerek duymadan eylemlerde bulunur.

Öğrenenler yapıp bozarken birçok şey dener ve denemeleri farklı şekillerde gerçekleştirir. Bu esnada üzerinde çalışılan şeye çok farklı açılardan bakma fırsatı yakalar. Öğrenenler yapa-boza etkinlikler esnasında kullandıkları araçları ya da teknolojileri nasıl kullandıklarına ilişkin bir anlayış da geliştirir. Sonuç olarak, ya-

pa-boza etkinlikler öğrenenlere standart bir ölçme aracıyla doğru cevaba ulaşma çabasından çok daha zengin bir öğrenme deneyimi sunar.

Yapa-Boza Öğrenme Sürecinde Robotik Etkinlikler

İklim değişikliği, kuraklık, yiyeceklerin ziyan edilmesi ve enerji ile ilgili günlük hayattaki problemlere yönelik yaratıcı ve yenilikçi çözümler üretebilecek nesillere ihtiyaç duyulmaktadır. Bu kapsamda yapa-boza eylemler öğrenenlerin problemlerin çözümü ve daha geniş bakış açısıyla düşünebilmeleri için kullanılabilecek güçlü bir yaklaşımdır. Yapa-boza eylemler öğrenmenin gerçekleşmesi için öğrenenlere eşsiz fırsatlar sunmaktadır. Bu süreçte öğrenme şans eseri gerçekleşmez. Öğrenenler bu süreç içerisinde kendilerine sunulan fırsatlarda tasarımla ilgili birtakım kararlar vererek, belirli ilkeler doğrultusunda bu süreci gerçekleştirir (Worker, Ambrose & Mahacek, 2014). Yapa-boza öğrenme etkinliklerinde robotiklerle öğrenenlere yeniden kullanılabilir, değiştirilebilir materyaller üretme fırsatı sağlanırken öğrenenlerin keşfederek öğrenme süreci de desteklenmektedir. Ek olarak programlama sürecinde robotlarla yapılacak olan yapa-boza etkinlikleri soyut sözdizimlerin (program kodları, blokları vs.) etkisinin ya da işlevinin otantik dünyada adım adım somutlaştırılmasına olanak sağlamaktadır. Bu sayede sözdizimsel ve semantik unsurların bütünleştirilmesini kolaylaşmaktadır. Okul öncesi seviyedeki çocuklar ile yapılan birçok çalışma dört-altı yaşlarındaki çocukların basit robotik projeleri geliştirip programlayabildiğini, ayrıca mühendislik, teknoloji ve bilgisayar bilimlerinden köken alan güçlü fikirleri öğrenirken aynı zamanda bilgisayımsal düşünme becerilerini geliştirebildiklerini göstermiştir (Bers, Ponte, Juelich, Viera & Schenker, 2002; Cejka, Rogers & Portsmore, 2006).

Robotik etkinlikler aracılığıyla içerikle ya da materyallerle uğraşan öğrenenlere problem çözme becerilerini pratiğe dökme fırsatı da sunulmaktadır. Robotik etkinlikler, çocukların işbirliği ve takım çalışması yaparken aynı zamanda ince motor becerilerini ve el-göz koordinasyonlarını geliştirmelerine olanak tanımaktadır. Dahası, küçük yaştaki çocuklar robotik etkinlikler aracılığıyla anlamlı bağlamlar içerisinde mühendislik kavramlarını ve öyküsel bağlamlar oluşturarak hikâye anlatımlarını deneyimleyebilirler. Bu tip robotik projelere katılan küçük çocuklar yaratıcı bir ortamda oynarken öğrenebilirler (Resnick, 2003). Öğrenme süresince kullanılan eğitsel robotikler, sadece bilgi yapımı için değil, aynı zamanda belirli alanlarda problemleri olan öğrenciler için yardımcı bir araç olarak da hizmet edebilir. Yani ortak çalışmalar için oldukça elverişli olan robotikler öğrencilerin kabul görmelerine ve aidiyet duygularının yükselmesine olanak tanıyacak biçimde öğrenmeye yönelik tutumlarını değiştirmek için kullanılabilir. Bu da sosyal olarak dışlanma ve ders ya da okul bırakma riskini azaltabilir (Daniela & Lytras, 2019).

Yapa-Boza Öğrenme Sürecinde Blok Tabanlı Uygulamalar

Bireylerin öğrenme sürecinde keşfetmesine ve yaratıcılıklarını uygulamalı deneyimlerle birleştirmelerine olanak sağlayan yapa-boza etkinlikler için kullanılacak araçlardan birisi de blok tabanlı programlama araçlarıdır. Blok tabanlı programlama araçlarındaki açık uçlu görevler problemlerin çözümünü sanal ortamlarda fiziksel olarak oluşturmak için öğrenenleri teşvik etmektedir. Blok tabanlı programlama araçlarıyla yapılan uygulamalarda yapa-boza etkinlikler, öğrenenlerin görevleri gerçekleştirirken hedefe ulaşmak için yaptıklarını test etmesi, araştırma yapması ve kodlarla uğraşırken sergiledikleri davranışları düzenlemesi olarak tanımlanmaktadır. Blok tabanlı uygulamalarla programlama sürecinde yapa-boza etkinlikler 3 kategoriye göre sınıflandırılmaktadır (Dong, Marwan, Cateté, Price, Barnes, 2019). Bunlar;

- *Teste dayalı yapa-boza etkinlik (Test-based tinkering):* Öğrenenlerin düzenlediği kodları parçalar halinde çalıştırabildiği ve düzenleyebildiği etkinlikleri içermektedir. Bu etkinliği yapan öğrenenler kısa sürede kodlarında sıkça düzenlemeler yapma eğilimindedir. Hata ayıklama ve çalıştırma olarak sınıflandırılmaktadır.
- *Prototipe dayalı yapa-boza etkinlikleri (Prototype-based tinkering)*: Bir özelliğin önce bir prototip üzerinde test edildiği etkinlikleri içermektedir. Farklı durumlara ait kod blokları için prototipler oluşturulabilir. Kodlar, ana kod bloğundan ayrı oluşturulup test edildikten sonra ana kod bloğuna eklenebilir. Başka bir ifadeyle parçalar halinde oluşturulup test edilen kod blokları birleştirilerek bütün haline getirilebilir. Bu etkinlik öğrenenlerin uzunluğuna bakmaksızın farklı alanlarda oluşturdukları kod bloklarını tanımlamak için kullanılan yan proje (side-project) ve bir kod bloğunun prototipinin oluşturularak test edildikten sonra yeni özelliklerin eklendiği ve kod bloğunun arttırıldığı genişletilmiş (expanding) proje olmak üzere 2 şekilde sınıflandırılmaktadır.
- *Oluşturmaya dayalı yapa-boza etkinlikler (Construction-based tinkering):* Bu etkinlik parçaların tek tek bütüne eklendiği inşa etme sürecini içermektedir. Etkinlikte öğrenenler kod bloğuna ekleme yapmak, kod bloğunu silmek ve kod bloğunda yeniden düzenlemeler yapmak için çok zaman harcamaktadır.

Blok tabanlı programlama uygulamalarından birisi de Scratch'tir. Scratch, öğrenmesi nispeten kolay, aynı zamanda karmaşık projeler yürütülmesine olanak tanıyan dünya genelinde özellikle K12 seviyesinde BD ve programlama etkinliklerinde oldukça yaygın olarak kullanılan blok tabanlı görsel bir programlama

aracıdır. Scratch öğrencilerin görsel kod bloklarını sürükleyip bırakarak kodlama yapmalarına olanak sağlamaktadır. Öğrenciler bu ortamda kod bloklarını kolaylıkla birleştirip kodlama sonuçlarını hemen gözlemleyebilmektedir. Scratch akran işbirliğini ve yapa-boza etkinlikler aracılığıyla bireysel öğrenme sürecini desteklemektedir (Maloney, Resnick, Rusk, Silverman & Eastmond, 2010). Herkes için programlama sözüyle yola çıkan Scratch öğrenenlere başkalarıyla paylaşabilecekleri oyunlar, animasyonlar, hikayeler, müzik, video ve diğer birçok uygumalamayı tasarlamak ve kullanıcı dostu ve kolay erişilebilir (çevrim içi ve çevrim dışı) bir arayüz sağlamak için tasarım (kod) blok setleri kullanmaktadır (Resnick vd., 2009). Bu yapısıyla da yapa-boza etkinlikler için uygun bir platform sunmaktadır. Scratch uygulamasına alternatif olarak kullanılan Alice, mBot, App Inventor gibi programlar da bulunmaktadır. Ancak görsel programlama araçlarının çoğunda ses kullanılsa da videoların eklenmesi mümkün değildir. Bu nedenle mobil cihazlarda kullanılabilecek Bricoleur isimli görsel programlama araçları da yapa-boza etkinlikler için öğrenenlere bir alternatif sunmaktadır (Hickey, 2019).

Görsel programlama araçları kullanılarak gerçekleştirilen yapa-boza etkinlikler, programlamaya yeni başlayan öğrenenlerin kodun yapısını kolayca anlamasını sağlayarak, kodun araştırılmasında ve düzenlenmesinde köprü görevi görmektedir (Berland vd., 2013; Turkle & Papert, 1992). Blok tabanlı programlama uygulamalarıyla, insanların programlama bloklarını kullanarak fiziksel nesnelerle girdikleri etkileşimle benzer bir hissi yaşamaları sağlanmaktadır (Resnick & Rosenbaum, 2013). Görsel programlama araçları öğrenenlerin programlamaya karşı kaygısını da azaltmaktadır (Chang, 2014). Sürükle-bırak esasına dayanan blok temelli görsel programlama uygulamaları kod yazmak için sözdiziminden kaynaklanan karmaşıklığı ortadan kaldırıp, programlamanın semantik boyutu üzerine yoğunlaşılmasını kolaylaştırmaktadır. Blok temelli görsel programlama araçları sadece K12 düzeyi için değil aynı zamanda programlama deneyimi olmayan üniversite öğrencileri için de temel bilgisayımsal kavramlarının daha iyi anlaşılmasında önemli katkılar sağlamaktadır (Çetin, 2016). Bir programın sonuçlarının görsel programlama araçlarıyla görselleştirilmesinin en büyük yararı, öğrenenlerin programladıkları nesnelerin çalışmasını takip etmesi ve tüm program elemanları arasındaki ilişkileri eşzamanlı olarak gözlemleyerek programın sonuçlarını bir bütün olarak ele alma, böylece programlama mantığını daha iyi geliştirme fırsatı yakalamasıdır. Bu aynı zamanda programın değiştirilmesi ya da yeniden inşa edilmesi sürecinde önemli fikirlerin geliştirilmesini sağlamaktadır (Berland & Wilensky, 2015).

Yapa-Boza Öğrenme Etkinliklerinin Değerlendirilmesi

Yenilikçi ve yaratıcı fikirlerin ileri sürüldüğü bir oyun atmosferinde, öğrenenlerin üretici rolü üstlenerek kendi hedefleri doğrultusunda el emeği ürünler ortaya koyma fırsatı bulduğu yapa-boza etkinlikler öğrenci merkezli bir süreçtir. Peki, öğrencilerin eğlenceli etkinlikler aracılığıyla öğrenmesi nasıl değerlendirmelidir? Yapa-boza etkinlikleri değerlendirme sürecinde öncelikle öğrenenlerin ne oluşturdukları, nasıl oluşturdukları ve neden oluşturdukları sorgulanmalıdır. Papert'ın yapıcı yaklaşımında öğrenenler tarafından oluşturulan ve paylaşılabilir olan ürünlerin öğrencilerin kendi anlamlarını oluşturmasını kolaylaştırdığı vurgulanmaktadır. Bu yaklaşımda geleneksel problem çözme süreçlerinden farklı olarak öğretmenin oluşturduğu ya da verdiği problemin çözülmesinden ziyade, öğrenenden hedefin ve problemin geliştirilmesini içeren tasarımlar yapması beklenir. Çözülmesi gereken problem öğretici ya da uzman kişiler tarafından değil, öğrenen tarafından belirlenir. Bu açıdan bakıldığında yapa-boza eylemlerin değerlendirilmesinde ürün temelli ve süreç temelli yaklaşımların birlikte kullanılması önemlidir (Matusov, 1997).

Gabrielson (2015) tarafından yapa-boza etkinliklerin değerlendirilmesi için şu önerilerde bulunulmaktadır:

- Öğrenenlerin kavramları ne kadar anladıklarını öğrenebilmek için yapa-boza etkinliği süresince öğrenenlere eşlik edilmeli ve onlarla sohbet edilmelidir.
- Öğrenme sürecinde öğrenenlere önceden yapılandırılmış ve standart cevapları olan sorular sorulmamalıdır.
- Öğrenenlerden günlük tutmaları istenerek, yaptıklarının kayıt altına alınması sağlanabilir.
- Süreç bitiminde öğrenenlerin değerlendirme anketi doldurmaları sağlanabilir.
- Projeleriyle ilgili çizimler yapmaları, fotoğraf çekmeleri ve video oluşturmaları sağlanabilir.
- Projeleriyle ilgili düşündüklerini açıklayan videolar oluşturmaları istenebilir.
- Öğreticinin de izleyici olarak dâhil olduğu bir süreçte öğrenenlerin akranlarına ya da farklı kişilere proje ile ilgili süreci ve sonuçları içeren sunumlar yapmaları sağlanabilir.

Sesli düşünme gibi bilişsel görev analizleri eğitimcilere yapa-boza öğrenme sürecine ilişkin olarak oldukça farklı izlenimler kazandırabilir. Yapa-boza öğrenme özünde bir inşa süreci olduğu için gerçekleştirilen çözüm adımlarına öğrenen

tarafından atfedilen anlamların ortaya çıkarılması gerekmektedir. Bu durumda özellikle öz-raporlamaya dayalı sorgulama oldukça işlevsel olabilir.

Yapa-Boza Öğrenme Sürecinde Öğretimsel Rehberliğin Rolü

Öğrenenlerin hayalindeki projeleri ya da materyalleri geliştirmelerini sağlamak kimi zaman zorlayıcı olsa da, öğreticiler için odak noktası projenin son hali, ortaya çıkan materyal veya araç ya da öğrenilmesi beklenen kavramlar değil öğrenenlerdir. Bu süreçte öğretici, öğrenenlere ders anlatmaktan ziyade onların ilgilendikleri şeyle uğraşırken öğrenmesine fırsat vermelidir. Öğrenenler yardım istediğinde ipuçları verilerek onlara yardım edilmelidir (Gabrielson, 2015). Yapa-boza öğrenme sürecinde öğreticilerin rolü Şekil 27.2'deki gibi tanımlanmaktadır (The Tinkering Studio, 2015).

Hedefler	Uygulamalar	Teknikler
İlk ilgiyi oluşturma	• Katılımcılara hoşgeldin diyerek onları etkinlik alanına davet edin. • Etkinliği tanıtın ve etkileşimli bir ortam oluşturun.	• Gülümseyin ve kendinizi tanıtın. • Öğrenenleri varolan araç ve materyallere yönlendirin. • Çalışmaya başlamak için yer önerin. • Açıklama yaparken ya da örnek verirken gözle iletişim kurun. • Farklı düşünceler içeren örnekleri gösterin. • Harekete geçirmek için olasılıklar içeren bir öneride bulunun.
Öğrenenlerin fikirlerini takip ederek katılımı sağlama	• Her türlü fikre, yapılan hatalara, yanlış yönelimlere değer verin. • Öğrenenlerin hatalı yaptığı ve hayal kırıklığı yaşadıkları durumlarda onlara destek olun.	• Yardım etmeden önce öğrencileri gözlemleyin. • Öğrenenlere süreç ile ilgili sorular sorun. • Öğrenenlerin fikirlerini dinleyin. • Durumları ve soruları yeniden şekillendirin. • Yeni materyal ve araçlar önerin. • Cevabı bilmiyorsanız, öğrenenle birlikte çalışın. • Öğreneni yönlendirmek yerine önerilerde bulunun. • Öğrenenin fikirlerine ilgi gösterin.
Bağlantılar oluşturarak anlamı derinleştirme	• Öğrenenlerin yapacaklarının ötesine geçebilmeleri için onlara rehberlik edin. • Öğrenenlerin projeleriyle ilgili farklı alanlarda öğrenme deneyimi yaşayabilmesi için onlara bağlantılar sunun.	• Öğrenenleri ilham alacakları alanlara yönlendirin. • Ortamdaki ortak hedeflere dikkat edin. • İlgi varsa teknik terimler önerin. • Öğrenenlerin düşüncelerini açıklamalarını ve sonraki adımı tanımlamalarını sağlayın. • Öğrenenleri risk almaları ve deney yapmaları için teşvik edin. • Öğrenenlerin kendi yollarında ilerlemesini sağlamak için zorluklar sunun. • Öğrenenlerin deneyimlerinin farklı ilgi alanlarıyla ilişkisini tartışın. • Merak, sürpriz ve sevinç anlarını kutlayın.

Şekil 27.2. Yapa-boza öğrenme sürecinde öğreticilerin rolü (The Tinkering Studio, 2015)

Yapıp-bozma özellikle çocukluk döneminde oyun oynarken sıkça deneyimlenen otantik bir öğrenme etkinliğidir. Çocuklar oyuncak parçalarını bir araya getirip aralarındaki ilişkilerin farkına varıp bazen ilişkileri kendileri var edip yeni bir şeyler ortaya çıkarmaktadır. Aslında bu yaklaşım günümüz toplumunda yetişkenlerin de daha üretken olmasını sağlama potansiyeline sahiptir. Yapılan çalışmalar yoğun ve sistematik bir şekilde (gelişigüzel değil) yapıp bozan acemilerin daha fazla öğrenme kazanımı elde ettiğini göstermektedir (Perkins vd., 1986). Programlama yaparken bazı öğrenenler bir problemle karşılatıklarında, çözüme ulaşmak için sürekli kodları değiştirip, yeniden derleyip, planlama becerilerini işe koşarak programlama hakkında daha fazla çıkarım elde ederken; bazıları sorunlar karşısında programlamadan vazgeçme eğilimi sergilemektedir. Risksiz bir ortamda eğlenceli ve zorluklarla keşfetme özgürlüğüne sahip olmak, bireyin kendisine güvenini artırırken aynı zamanda onu yeni şeyler denemeye motive eder. Ek olarak, öğrenenlere sağlanan manipülatif programlama araçları ve açık uçlu programlama görevleri onların yapım işleriyle meşgul olurken bir amaca ulaşmak için özgün çözümler geliştirmeleri için uygun koşullar sağlar.

Sonuç

Eğitimden beklentilerin başında otantik hayattaki problemleri fark eden, tanımlayan ve bunları yaratıcı biçimde çözen öğrencilerin yetiştirilmesi gelmektedir (Resnick, 2017). Dijital teknolojilerdeki ilerlemeler disiplin ayırt etmeksizin yeni problem çözme stratejilerinin tasarımına, hem sanal hem de otantik dünyada çözümlerin test edilmesine olanak sağlamaktadır (Barr & Stephenson, 2011). Ayrıca sanal ortamda dolaşan verilerin katlanarak artması karşılaşılan problemlerin çözümü için bu bilgilerden yararlanılarak bilgisayımsal yöntemlerin kullanımını gerektirmektedir. Kısaca, teknolojinin baş döndürücü hızına uyum sağlayıp başarılı olmak için bireylerin teknolojik araçları sadece benimsemekle ve kullanmakla yetinmeyip, bu teknolojileri eleştirel bir bakış açısıyla ele alıp, yaratıcı çözümlerine araçsal bir destek olarak görmesi ve üretim odaklı düşünmesi gerekmektedir. Bu da bilgi toplumundan ziyade yaratıcı topluma geçilmesiyle mümkün gözükmektedir.

Günümüzde BİT okuryazarlığı hemen her alanda ihtiyaç duyulan temel bir yetkinlik olsa da bu yetkinliğin sayısal kavramlardan yararlanarak problem çözme becerisini de kapsayacak şekilde güncellenmesi gerektiği konusunda küresel bir eğilim oluşmuştur. Bu kapsamda Türkiye'nin de dâhil olduğu çok sayıda ülke bilgisayımsal okuryazarlık için BİT programlarını revize etmeye başlamıştır (Jun, Han, Kim & Lee, 2014). Öte yandan, BD'ye ilişkin tanımlama ve açıklamaların belirli bir olgunluk düzeyine erişmesini takiben bu düşünme becerisinin nasıl des-

tekleneceğine ve değerlendirileceğine ilişkin pratik sorular ön plana çıkmaya başlamıştır. BD'nin temelinde otantik hayatta karşılaşılan problemlerin bir bilgisayar bilimcisinin bakış açısıyla çözüme kavuşturulabileceği görüşü yatmaktadır. Okullarda BD ve programlama eğitimi çoğunlukla bilişim derslerine yerleşik olarak yürütülmektedir. Bu durum herkes için BD ilkesine ters düşmektedir. BD sadece bilgisayar bilimleri için değil her alanda işlevsel olabilecek bir beceri kümesidir. Bilgisayımsal yaklaşımlar çok geniş yelpazedeki sorunlara ve durumlara uygulanabilir; bir hastalığın yayılma hızı, istiflenen yapı bloklarının stabilitesi, farklı renk karışımlarının ortaya çıkardığı harmoni gibi. Bunun için BD'nin disiplinler üstü bir anlayışla farklı alanlardaki bilgilerin edinilmesinde kullanılacak bir düşünme biçimi olarak görülmesi gerekmektedir.

Öğrencileri bilgisayar bilimi kavramlarıyla tanıştırmanın en etkili yollarından birisi de programlama etkinlikleridir. Programlama ne düşünülmesi yerine nasıl düşünülmesi gerektiğini sorgulatan etkili bir öğretimsel etkinliktir. K12 düzeyinde programlama eğitimi ve problem çözme uygulamaları için geliştirilmiş çok sayıda araç bulunmaktadır. Bununla birlikte bu araçlarla meşgul olmanın ya da programlama yapmanın BD gelişimini tek başına sağlayacağı görüşü oldukça varsayımsaldır. Papert (1980) en iyi öğrenme yolunun öğretilmeksizin öğrenme olduğunu savunmaktadır. Benzer şekilde Resnick (2017) öğrenenlerin neyi, nerede, ne zaman ve nasıl öğreneceklerini kendi seçimleriyle şekillendirmeleri gerektiğinin altını çizerek bunun okul öncesi tarzı öğrenme biçiminin her kademedeki öğretimsel düzenmelere entegre edilmesiyle mümkün olabileceğini belirtmiştir. Resnick, okul öncesi dönemdeki çocukların oyun oynarken hayal etme, üretme, oynama, paylaşma ve yansıtma olmak üzere bir yaratıcı öğrenme sarmalından geçtiğine dikkat çekmiştir. Dahası bunun her yaştan bireyin yaratıcılık kapasitesini geliştirmesi için gerekli olan şey olduğunu belirtmiştir. Okul öncesi dönemdeki öğrencilerin oyun alanları incelendiğinde bu alanlardaki nesnelerin olabildiğince çeşitlilik arz ettiği görülmektedir. Bunun nedeni farklı ilgilere sahip öğrencilerin ihtiyaçlarına cevap verme çabasıdır. Maalesef okul öncesi dönemde bir oyun alanı şeklinde tasarlanan öğrenme ortamları okul döneminde genellikle bir oyun bahçesine dönüşmektedir. Oyun alanları çocukların neyi yaratacaklarına karar verdikleri, bunun için çok sayıda girişimde bulundukları, keşfetmeye açık bir yapıya sahiptir. Oyun bahçeleri ise araç ve gereçlerin kurulu halde bulunduğu, genellikle bu araç-gereçlerle yapılacak etkinliklerin sınırlarının önceden çizildiği fiziksel mekânlardır. Resnick (2017) oyun alanlarının yaratıcı düşüncelerin geliştirilmesi ve sınanması adına daha fazla imkân yarattığını belirtmektedir. Tıpkı okul öncesi dönemde olduğu gibi formal öğretim düzenlemelerinde de bireylerin otantik yaratıcılık kapasitelerinin dışa vurulmasını destekleyecek oyun alanlarının oluştu-

rulması tavsiye edilmektedir. Bunun için öğrencilerin ilgilerine cevap verebilecek şekilde çeşitli ürünlerin tasarımını destekleyecek manipülatif ortam ve araçların varlığı oldukça önemlidir.

Programlama özünde bir yapım sürecidir. Bu yapım sürecini bir oyun alanı biçiminde kurgulamak oldukça makul bir yaklaşım gibi görünmektedir. Çocuklar eğlenceli buldukları sürece denemekten, farklı bir şeyler ortaya çıkarmaya çalışmaktan ve bu süreçte gerekli riskleri almaktan çekinmezler. Yapa-boza etkinlikler bir dizi seri deneme-yanılma, yapma, bozma ve yeniden yapma girişimini içeren eğlenceli bir öğrenme etkinliğidir (Özdemir, 2019). Programlama sürecinde öğrencilerin yapıp bozma eylemlerini sergilemesini kolaylaştıracak ortam ve araçların kullanımı oldukça önemlidir. Eğitsel robotikler ve blok tabanlı programlama araçları programlama sürecinde manipülatif eylemlerin sergilenmesi için oldukça elverişlidir. Çocuklar farklı yapı bloklarından oluşan bir robot yapım setiyle (örneğin LEGO robotikleri) uğraşmaya başladıklarında bir yapıp bozma yani kurcalama sürecine dâhil olurlar. Bu süreçte geliştirdikleri bir fikri denemek, gerçekleştirdikleri eylemleri değerlendirmek ve çözümlerini daha iyi hâle getirmek için fırsatlar yakalarlar. Benzer şekilde blok tabanlı programlama araçlarıyla farklı programlama bloklarını sahneye taşıyıp, sözdizim kurallarına takılmaksızın programlama yapabilirler. Robotik araçların söz konusu programlama yapılarını otantik dünya düzenlemeleri içerisinde somutlaştırması eşsiz nedensellik bağlantılarının kurulmasına olanak sağlar. Benzer şekilde blok tabanlı araçlar zorlayıcı hata ayıklama sürecini kolaylaştırarak programlama sonuçlarını daha izlenebilir hâle getirir. Yapa-boza öğrenme girişimlerinde hatalar yer işaretleri gibi işlev görür. Öğrencilerin aynı zamanda bir problem çözme biçimi olan BD süreçlerini deneyimlerken çok sayıda hata yapmaları olasıdır. Bu süreçte hataların öğretimsel rehberliğinin farkına varılması önemlidir. Hata sonuçlarının gözlemlenmesi önemli bilişsel destek unsuru oluşturarak, gerçekleştirilen eylemlere yönelik bir anlayışın geliştirilmesini ve buna bağlı olarak sonraki eylemlerin revize edilmesini sağlar. Bununla birlikte karşılaşılan başarısız durumların öğrencilerde pes etme eğilimine yol açmaması için gerekli duyuşsal desteklerin sağlanması önemlidir. Ürün geliştirme sürecinde akran işbirliği oldukça önemlidir. Sadece ekrana bağlı programlama ortamları ya da araçları öğrenciler arası iletişim ve etkileşimi sağlama noktasında bir sınırlılığa neden olmaktadır. Öte yandan fiziksel robotikler, ekran dışında da var olduklarından, öğrenciler için daha fazla etkileşim olanağı sunabilir. Programlama ürünlerinin diğer öğrenenlerle paylaşıldığı ortamlar yeni fikirlere ilham verebilir. Ayrıca programcıların birbiriyle etkileşimini sağlayan eşzamanlı ve eşzamansız sanal iletişim platformları (sohbet, forum, wiki vb.) programlamaya ilişkin anlayışların geliştirilmesine ve farklı bakış açılarının oluşturulmasına katkı sağlayabilir.

Yansıma Soruları

1. Yapa-boza öğrenme etkinliklerinin öğrenme sürecindeki işlevini yapıcı (constructionist) öğrenme bakış açısıyla değerlendiriniz.
2. Performansa dönük iteratif (tekrarlı) bir yapıya sahip olan yapa-boza öğrenme sürecinde değerlendirme süreci nasıl olmalıdır?
3. Birçok ülkede K12 düzeyindeki Bilişim derslerinde ortaya çıkan öğretim programı değişikliği ihtiyacının sebepleri nelerdir?

Kaynakça

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835. doi:10.1093/comjnl/bxs074

Barefoot Computing at School. (2014). *Computational thinking.* Retrieved from https://www.barefootcomputing.org/concept-approaches/computational-thinking-concepts-and-approaches

Barr, V. & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *Inroads, 2*(1), 48-54.

Berland, M., Martin, T., Benton, T., Smith, C. P. & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences, 22*(4), 564-599. doi:10.1080/10508406.2013.836655

Berland, M. & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology, 24*(5), 628-647. doi:10.1007/s10956-015-9552-x

Bers, M. U., Ponte, I., Juelich, C., Viera, A. & Schenker, J. (2002). Teachers as designers: Integrating robotics in early childhood education. *Information Technology in Childhood Education Annual, 2002*(1), 123-145.

Bevan, B., Petrich, M. & Wilkinson, K. (2014). Tinkering is serious play. *Educational Leadership, 72*(4), 28-33.

Brahms, L. (2014). *Making as a learning process: Identifying and supporting family learning in informal settings* (Unpublished Doctoral Dissertation). University of Pittsburg, USA.

Burke, Q. & Kafai, Y. B. (2010). *Programming & storytelling: Opportunities for learning about coding & composition*. In Proceedings of the 9th International Conference on Interaction Design and Children (pp.348-351), Barcelona, Spain.

Burke, Q. & Kafai, Y. B. (2014). Decade of game making for learning: From tools to communities. In M. C. Angelides & H. Agius (Eds.), *Handbook of Digital Games* (pp. 689-709). Piscataway, NJ: IEEE Press.

Burleson, W. S., Harlow, D. B., Nilsen, K. J., Perlin, K., Freed, N., Jensen, C. N., . . . Muldner, K. (2018). Active learning environments with robotic tangibles: Children's physical and virtual spatial programming experiences. *IEEE Transactions on Learning Technologies, 11*(1), 96-106. doi:10.1109/TLT.2017.2724031

Cejka, E., Rogers, C. & Portsmore, M. (2006). Kindergarten robotics: Using robotics to motivate math, science, and engineering literacy in elementary school. *International Journal of Engineering Education, 22*(4), 711-722.

Chao, P.-Y. (2016). Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. *Computers & Education, 95*, 202-215. doi:https://doi.org/10.1016/j.compedu.2016.01.010

Chang, C. K. (2014). Effects of using Alice and Scratch in an introductory programming course for corrective instruction. *Journal of Educational Computing Research, 51*(2), 185-204. doi:10.2190/EC.51.2.c

Clegg, T. & Kolodner, J. (2007). Bricoleurs and planners engaging in scientific reasoning: A tale of two groups in one learning community. *Research and Practice in Technology Enhanced Learning, 2*(3), 239-265.

Cetin, I. (2016). Preservice teachers' introduction to computing: Exploring utilization of Scratch. *Journal of Educational Computing Research, 54*(7), 997-1021. doi:10.1177/0735633116642774

Daniela, L. & Lytras, M. D. (2019). Educational robotics for inclusive education. *Technology, Knowledge and Learning, 24*(2), 219-225. doi:10.1007/s10758-018-9397-5

de Raadt, M. (2007). A review of Australasian investigations into problem solving and the novice programmer. *Computer Science Education, 17*(3), 201-213. doi:10.1080/08993400701538104

Denning, P. J. (2009). The Profession of IT: Beyond computational thinking. *Communications of the ACM, 52*(6), 28-30. doi:10.1145/1516046.1516054

Denning, P. J. & Tedre, M. (2019). *Computational thinking*. Cambridge, MA: MIT Press.

diSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy* (1st ed.). Cambridge, MA: MIT Press.

Dong, Y., Marwan, S., Catete, V., Price, T. & Barnes, T. (2019). *Defining tinkering behavior in open-ended block-based programming assignments*. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 1204-1210), Minneapolis, MN, USA.

Farris, A. V. & Sengupta, P. (2016). Democratizing children's computation: Learning computational science as aesthetic experience. *Educational Theory, 66*(1-2), 279-296. doi:10.1111/edth.12168

Gabrielson, C. (2015). *Make tinkering: Kids learn by making stuff (2nd ed.)*. Sebastopol, CA: Maker Media.

García-Peñalvo, F. J. & Mendes, A. J. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior, 80*, 407-411. doi:https://doi.org/10.1016/j.chb.2017.12.005

Grover, S. & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43. doi:10.3102/0013189x12463051

Guzdial, M. (2008). Education: Paving the way for computational thinking. *Communications of the ACM, 51*(8), 25-27. doi:10.1145/1378704.1378713

Harris, E., Xanthoudaki, M. & Winterbottom, M. (2018). *Tinkering and science capital ideas and perspectives. Retrieved from http://www.informalscience.org/sites/default/files/TinkeringAndScienceCapital_LR.pdf.*

Hickey, S. (2019). *Bricoleur: A tool for tinkering with programmable video and audio*. In Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (p. LBW0118), Glasgow, Scotland UK.

ISTE & CSTA. (2011). *Operational definition of computational thinking for K–12 education*. Retrieved from https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

Jun, S., Han, S., Kim, H. & Lee, W. (2014). Assessing the computational literacy of elementary students on a national level in Korea. *Educational Assessment, Evaluation and Accountability, 26*(4), 319-332. doi:10.1007/s11092-013-9185-7

Kalelioğlu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing, 4*(3), 583-596.

Kalelioğlu, F., Gülbahar, Y., Akçay, S. & Doğan, D. (2014). *Curriculum integration ideas for improving the computational thinking skills of learners through programming via Scratch*. Paper presented at the 7th International Conference on Informatics in Schools: Situation, Evoluation and Perspectives (ISSEP), İstanbul, Turkey.

Kalelioğlu, F., Gülbahar, Y. & Doğan, D. (2017). Teaching how to think like a programmer: Emerging insights. In H. Özçınar, G. Wong, & H. T. Öztürk (Eds.), *Teaching Computational Thinking in Primary Education* (pp. 18-35). Hershey, PA: IGI Global Publishing.

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.

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I.K., Somanath, S. Weber, J. & Yiu, C. (2017). Pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education, 3(*2), 154–171.

Lee, I., Martin, F. & Apone, K. (2014). Integrating computational thinking across the K-8 curriculum. *Acm Inroads, 5*(4), 64-71.

Lévi- Strauss, C. (1962). *La pensée sauvage. Paris: Plon*. English translation The savage mind, first published 1966.

Lye, S. Y. & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51-61. doi:https://doi.org/10.1016/j.chb.2014.09.012

Maloney, J., Resnick, M., Rusk, N., Silverman, B. & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, 10(4), 1-15.

Martinez, S. L. & Stager, G. (2013). Invent to learn: Making, tinkering, and engineering in the classroom. Torrance, CA: Constructing Modern Knowledge Press.

Matusov, E. (1997). [Review of the book Constructionism in practice: Designing, thinking, and learning in a digital world, by Y. Kafai and M. Resnick]. *Journal of Educational Computing Research, 16*(4), 397-404.

Özdemir, S. (2019). Soloborative learning: Solo thinking, collaborative tinkering. *International Electronic Journal of Elementary Education, 11*(3), 217-219. doi:10.26822/iejee.2019349246

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

Perkins, D. N., Hancock, C., Hobbs, R., Martin, F. & Simmons, R. (1986). Conditions of learning in novice programmers. *Journal of Educational Computing Research, 2*(1), 37-55.

Ramadhan, H. A. (2000). Programming by discovery. *Journal of Computer Assisted Learning, 16*(1), 83-93. doi:10.1046/j.1365-2729.2000.00118.x

Resnick, M. (2003). Playful learning and creative societies. *Education Update*, 8(6), 1-2.

Resnick, M. (2017). *Lifelong kindergarten: Cultivating creativity through projects, passion, peers, and play*. Cambridge, MA: MIT Press.

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.

Resnick, M. & Rosenbaum, E. (2013). Designing for tinkerability. In M. Honey & D. Kanter (Eds.), *Design, make, play: Growning the next generation of STEM innovators* (pp. 163–181). New York: Routledge.

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

The *Tinkering Studio (2015). Facilitation field guide.* Retrieved from https://www.exploratorium.edu/sites/default/files/files/facilitation_field_guide.pdf

Turkle, S. & Papert, S. (1992). Epistemological pluralism and the revaluation of the concrete. *Journal of Mathematical Behavior, 11*(1), 3-33.

Wagh, A., Cook-Whitt, K. & Wilensky, U. (2017). Bridging inquiry-based science and constructionism: Exploring the alignment between students tinkering with code of computational models and goals of inquiry. *Journal of Research in Science Teaching, 54*(5), 615-641. doi:10.1002/tea.21379

Weintrop, D., Holbert, N., Horn, M. S. & Wilensky, U. (2016). Computational thinking in constructionist video games. *International Journal of Game-Based Learning, 6*(1), 1-17. doi:10.4018/ijgbl.2016010101

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. doi:10.1145/1118178.1118215

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. doi:10.1098/rsta.2008.0118

Wing, J. M. (2011). Research notebook: Computational thinking—what and why?. *The Link, Spring.* Retrieved from https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Worker, S., Ambrose, A. & Mahacek, R. (2014). *Making and tinkering in robotics (and 4-H).* Retrieved from http://4h.ucanr.edu/files/201950.pdf

Wyeth, P. (2008). How young children learn to program with sensor, action, and logic blocks. *Journal of the Learning Sciences, 17*(4), 517-550. doi:10.1080/10508400802395069

Zuckerman, O. & Resnick, M. (2003, 5-10 Apr). *A physical interface for system dynamics simulation.* Paper presented at the Conference on Human Factors in Computing (CHI 2003) New Horizons, Fort Lauderdale, FL.