O3 – Eğitmen Destek İçeriği

O3/A2 – C4G CİDDİ OYUN YAKLAŞIMI KULLANICI KILAVUZU

Belge Bilgileri

Çıktı Adı: O3/A2 - CODING4GIRLS Ciddi Oyun Yaklaşımı Kullancı Rehberi

Fikri Çıktı Numarası: O3 – Eğitmen Destek İçeriği

Fikri Çıktı Lideri: South-West University "Neofit Rilski" (Bulgaristan)

Katkı Sağlayanlar

İstanbul Valiliği – Türkiye

Ahu Şimşek, Kadir Fatih Mutlu, Abdurrahman Saygın

South-West University "Neofit Rilski" - Bulgaristan

Daniela Tuparova, Boyana Garkova, Ivanichka Nestorova, Rositsa Georgieva

UTH – Yunanistan

Hariklia Tsalapata, Olivier Heidmann, Kostas Katsimentes, Christina-Roxani Taka, Sotiri Evangelou, Nadia Vlachoutsou

University of Rijeka – Hırvatistan

Nataša Hoić-Božić, Martina Holenko Dlab, Ivona Franković, Marina Ivašić Kos

EU-Track – İtalya

Michela Tramonti, Alden Meirzhanovich Dochshanov, Luigi Tramonti

Virtual Campus - Portekiz

Carlos V. Carvalho, Rita Durão

University of Ljubljana - Slovenya

Joze Rugelj, Mateja Bevcic, Spela Cerar, Matej Zapushek

Feragatname

Bu proje Avrupa Birliği Erasmus + Programı tarafından finanse edilmiştir.

Bu yayında belirtilen bilgi ve görüşler yazar(lar)a aittir ve Avrupa Birliği'nin resmi görüşünü yansıtmayabilir. Avrupa Birliği kurumları ve kuruluşları veya onların adına hareket eden herhangi bir kişi burada yer alan bilgilerin kullanımından sorumlu tutulamaz.

İÇİNDEKİLER

1. GİRİŞ

C4G Projesi dijital dönemle iç içe açık ve yenilikçi eğitimi konu alır. Proje, teknoloji tabanlı bir toplumda çokça talep edilen programlama becerilerine odaklanır. Günümüzde algoritma tasarlama, kodlama ve programlama becerileri önemli kişisel beceriler haline gelmiştir. Çünkü bu beceriler mantıksal düşünme ve sorun çözme yetenekleriyle bağlantılıdırlar. Dolayısıyla bilgisayar bilimi becerilerinde kariyer yapmak yenilikçiliğe dayalı sektörlerdeki işverenlerin ihtiyacı olan beceriler arasında yer aldığından oldukça talep görmektedir. Bu sektörler önümüzdeki yıllarda da sürdürülebilir bir ekonomik büyümeye yol açarak yüksek istihdam sağlamaya devam edecektir.

Şu anda bilgisayar bilimi becerisi olan çalışan eksiği olduğundan şirketler gerekli BİT becerileri olan işçileri bulmakta zorlanmaktadır. C4G Projesinin amacı etkili biçimde bilgisayar bilimi becerilerini ilkokulun son yıllarında ve ortaokulun ilk yıllarında işleyerek bu boşluğu doldurmaktır. Bu yıllarda birçok öğrenci bilgisayar bilimine ilgisini yitirmektedir. Üstelik bu alanda çokça bilinen bir cinsiyet dengesizliği vardır. Çünkü kızlar genellikle bilgisayar bilimi ile ilgili kariyer peşinde olmakla ilgilenmemektedirler.

C4G Projesi, cazip kariyer imkânı sunan bilgisayar biliminde kızlar ve erkekler arasında eşit fırsatları teşvik ederek kapsayıcı eğitim ve öğretimi ele almaktadır. ~~alır~~. Proje, öğrencilerin, öğretmenlerin ve ebeveynlerin bilgisayar bilimi kariyerleriyle ilgili yanlış algılarını ve tutumlarını ifade etmektedir. . Aynı zamanda bilgisayar bilimlerinin kız ve erkek çocuklara olan faydalarını, gerçek hayatla olan bağlantılarını göstererek ve bu alandaki kariyerin cinsiyet veya kültürel altyapıdan bağımsız olarak ödüllendirildiğini vurgular.

Proje, okul öğrencilerine gelecekteki akademik (bilgisayar bilimindeki yüksek öğrenim) ve kariyer (bilgisayar bilimiyle ilgili mesleki veya profesyonel aktiviteler) uğraşlarında başarılı olmalarını sağlayacak gerekli altyapıyı oluşturmada yardımcı olur. CODING4GIRLS ayrıca, eğitimciler ve öğrencileri bilgisayar bilimi becerilerini geliştirmede güçlendirerek öğretmen yeterliliklerinin ve öğretmenlik meslek profilinin gelişimini hedeflemektedir. Ayrıca eğitimcilerin akademik veya profesyonel hayatta cinsiyet eşitliği perspektifini teşvik eden girişimlere liderlik etmelerini desteklemektedir.

Öğretmenler için kullanıcı kılavuzu, proje çerçevesinde geliştirilen bilişsel çıktıların bir parçasıdır. Öğretmenler, C4G Proje metodolojisinde vücut bulan ana kavramlar, oyun tabanlı öğrenme, ciddi

oyunlar, tasarımsal düşünme; ICT'de kızların yeri ve onların ciddi oyun hakkındaki algıları; C4G oyun ortamının iki öğesiyle birlikte özellikleri, öğretmene ve öğrenciye düşen görevler; programlama öğretmeye yönelik oyun tabanlı öğrenme yaklaşımlarının incelemeleri ve faydalı programlama ortamlarının örnekleri hakkında bilgi edineceklerdir. Kılavuz aynı zamanda derslerin öğrenme sayfalarıyla ve onların C4G Proje metodolojisine ve geliştirilen oyun tabanlı ortama uyarlanmasıyla da ilgili bilgiler sunmaktadır.

2. TEMEL KAVRAMLAR

2.1. Ciddi Oyunlar

Oyun geniş bir kavramdır. Bu, herhangi bir etkinliği oyun yapan en önemli özelliklerin belirlenmesini de zorlaştırmaktır. Thorton'a göre oyunun en önemli özelliği interaktif olmasıdır. Johnston oyunu, her katılımcının uyması gereken kuralları olan, bir veya birden fazla hedefi olan, etkileşim ve dinamik görselleri olan bir aktivite olarak görür (Johnston & de Felix, 1993). Malone (1981), teorisinde her oyunun en önemli öğelerinin hayal, merak, zorluk ve kontrol olduğunu dile getirir. Bu konudaki en kapsamlı tanımı yapan Garris v.d. (2002) oyunun bileşenlerinin rekabet, meydan okuma, sosyal etkileşim ve iletişim olduğunu belirtir. Prensky'nin oyun tabanlı öğrenme teorisi en etkili olanlardan biridir. Ona göre bir oyun şu öğeleri barındırır. Kurallar, hedefler ve görevler, sonuçlar ve geri dönüş, çatışma durumları (rekabet, görev, karşıtlık), etkileşim ve hayal çerçevesi (Prensky, 2001). "Ciddi oyunlar" kitabının yazarları oyunu, oyuncunun bütün dikkatini üstüne çeken ve bütün oyuncuların uyması gereken kurallara göre oynanan, gerçek dünyadan ayrı (gerçek dünyayla bir ilişkisi olan veya olmayan hayali dünya) bir gönüllü aktivite (bir özgürlük biçimi) olarak tanımlar (Michael &Chen, 2006).

Oyunun en çok ortaya atılan özelliklerinden biri etkileşimdir. Bu bilgisayarla, rakiple (bilgisayar veya insan kontrolünde) veya diğer takım arkadaşlarıyla etkileşime girmek manasında kullanılır. Bilgisayar oyunları tek oyunculu, çok oyunculu ve devasa çok oyunculu oyunlara aynı anda giren oyuncu sayısına göre ayrılırlar. Bildiğimiz oyun türleri atari, aksiyon, savaş, rol yapma, strateji, platform, spor, simülasyon ve macera oyunlarıdır. Bilgisayar oyunlarının bilinirlik şemaları, en çok oynanan oyunların devasa çok oyunculu rol yapma oyunları, starteji ve aksiyon oyunları olduğunu gösteriyor.

Oyunun gücünü tam manasıyla anlamak için öncelikle çocuk oyununa bakalım. Çocuk oyunu, yaşam boyu önemli becerileri geliştirmeye yardımcı olan en önemli aktivitelerden biri olarak bilinir. Çocuk, oyun oynayıp gerçek dünyada ilk temel kavramları keşfederek ve onlar arasında pratik bağlantılar kurarak bilişsel kabiliyetlerini geliştirir. Jean Piaget'ye göre oyun yeni materyallerin var olan bilişsel yapıya eklenmesi ve yeni öğrenilen davranışın sağlamlaştırılmasıdır. Freud oyunun duygusal faydalarının altını çizer ve onun içgüdüsel anksiyeteyi azalttığını iddia eder. Bu, çocuğa

duygusal sorunları çözmede yardım eder. Sosyal yapılandırmacılar, oyunun sosyal becerileri geliştirmesinden dolayı önemli olduğuna inanırlar.

Oyunla öğrenme, erken yaşta görülen en yaygın etkinliklerden biridir. Ancak oyun oynamanın ve oyunların öğrenme üzerindeki olumlu etkileri formal bir eğitimde genellikle göz ardı edilir. Oyun temelli öğrenme alanında, öğrenme materyalinin bir bilgisayar oyunu formatında sunulması durumunda motivasyon üzerinde olumlu etkileri olduğunu gösteren çok sayıda araştırma yapılmıştır. Oyun, öğrencilerin öğrenme materyallerine dikkat çekmelerine yardımcı olan ve aynı zamanda onlara zevk veren bir eğlence kaynağıdır. Bu iki bileşene bir öğrenme sürecinde sahip olmak, bize rahat ve motivasyonu yüksek, dolayısıyla öğrenmeye daha istekli bir öğrenci sunmaktadır. Diğer pedagojik faydalar işbirliğine dayalı öğrenme, deneysel ve aktif öğrenme, probleme dayalı öğrenmedir. Günümüzde resmi eğitimde BİT kullanımı yaygındır. Bu da bize öğrenmeyi teşvik etmek için kaliteli malzemeler geliştirme fırsatı vermektedir.

Ciddi oyunlar genellikle eğitim, reklam veya simülasyon için kullanılan oyunları ifade eder. Öğrencilerin güvenlik, maliyet veya zaman gibi farklı nedenlerle gerçek dünyada imkânsız olan durumları deneyimlemelerine olanak tanır. Öğrencilerin tanımlanmış öğrenme çıktılarına sahip olmaları beklenir. Ciddi oyunların oyuncuların yeni bilgi veya farklı beceriler geliştirmeleri üzerinde olumlu etkileri olduğu iddia edilmektedir. İyi bir ciddi oyun tasarlamak için iyi bir yazılım tasarlayıp üretebilmemiz gerekir. Ancak ciddi oyunlar yazılımdan çok daha fazlasıdır. Aynı zamanda iyi bir öğretim tasarlayıp üretebilmeliyiz. Ciddi oyunların öğrenme için potansiyel değeri yüksek görünmektedir. Ancak sınıfta oyun kullanımına karşı direnç devam etmektedir. Daha fazla öğretmeni oyunları denemeye ikna etmenin makul bir yolu, mevcut oyun tasarımlarının unsurlarını kabul edilen öğrenme ve öğretim teorileriyle birleştiren pedagojidir.

Rieber, Smith ve Noah 1998'de eğitimde oyunların iki farklı uygulaması olduğunu belirtmişlerdir. Oyun oynama ve oyun tasarımı. Oyun oynama, öğrencilere hazır oyunların sunulduğu geleneksel yaklaşımdır. Oyun tasarımı, oyunun başkaları için ilginç olup olmamasına bakılmaksızın bir oyun inşa etme eyleminin başlı başına bir öğrenme yolu olduğunu varsayar. "Tasarlayarak öğrenme" fikri, tasarım ve geliştirme sürecine aktif katılımın bir şeyi öğrenmenin en iyi yolu olduğu varsayımına dayanır. Bu yaklaşım, bilgisayar tabanlı tasarım ve yazma araçlarının yaygınlaşması nedeniyle artan bir önem kazanmıştır.

2.2. Oyun Temelli Öğrenme

Eğitimcinin dikkatini oyunlara çeken bazı nedenler vardır (Gee, 2007). Örgün eğitimde, öğretime odaklanan geleneksel didaktik modelden aktif öğrenen rolünü vurgulayan öğrenci merkezli modele geçiş bu nedenlerden biridir. Bir diğer neden ise öğrenme hedeflerini sadece bilgiyi hatırlamak gibi daha düşük taksonomik seviyelerden yeni ortamlarda bilgiyi bulma ve kullanma gibi daha yüksek seviyelere taşımayı başarabilmemizdir. (Rugelj, 2016).

Prensky (2001), Gee (2003) ve Whitton (2009), oyun temelli öğrenmeyi dijital oyunların kullanımıyla öğrenme süreci olarak tanımlamışlardır. Oyunlar öğrenme için motivasyon sağlayabilir. Böylece istenen öğrenme sonuçlarının elde edilme şansını artırabilir. Öğrenme, deneyim veya uygulama yoluyla bilgi veya becerilerin kazanılması ve bir oyun yoluyla öğrenmekten daha iyi bir yol olarak tanımlanır (Pivec ve Kearney, 2007). Oyun temelli öğrenmeyle ilgili hemen hemen tüm araştırmalar öğrenme materyalleri bir bilgisayar oyunu formatında sunulduğunda öğrencilerin oldukça motive olduklarını ve bunun bir öğrenme sürecinin hızlanmasında olumlu etkileri olduğunu göstermektedir (Zapušek ve Rugelj, 2013). Öğrencilerin öğrenilmesi gerekenlere odaklanmak için motivasyona ihtiyaçları vardır. Ancak herhangi bir kaliteli öğrenmenin gerçekleşmesi için bu yeterli değildir. Bilgisayar oyunlarıyla öğrenen öğrencilerin öğrenme kazanımlarıyla daha farklı öğrenme materyalleri ile öğrenen öğrencilerin öğrenme kazanımları karşılaştırıldığında, belirgin bir fark olmadığı görülmektedir. Bilgisayar oyunu temelli öğrenmenin öğrenci motivasyonunda artış sağlarken öğrenme kazanımlarında bir fark oluşturmamasının sebebi kullanılan oyun tasarımlarının uygun olmamasıdır.

Oyunlar öğrenciler için çok çekici olabilir, ancak eğlendirir ve öğretmezlerse, oyunların eğitimde kullanılması pek bir anlam ifade etmez. Öyleyse, bilgisayar oyununu eğitici bir bilgisayar oyunu yapan unsurların neler olduğunu bulmalıyız.

Gross (2003), eğitim amaçlı dijital oyunların iyi tanımlanmış öğrenme hedeflerine sahip olması ve öğrencilerin bilişsel ve entelektüel yeteneklerini artırmak için önemli stratejilerin ve becerilerin geliştirilmesini teşvik etmesi gerektiğini iddia etmektedir.

Malone (1981b) ve Garris'e (2002) göre, dijital oyunların eğitimsel değerlerine katkıda bulunan unsurlar, duyusal uyaranlar (öğrenme materyalinin görsel ve işitsel temsilleri), fantezi (hayali ortamda sunulan bağlam), meydan okuma (zorlayıcı veya teşvik edici durum) ve meraktır (bilme

veya öğrenme arzusu). Bu unsurlar, hedefleri ve kuralları yapılandırmak, anlamlı bir öğrenme bağlamı, çekici bir hikâye, anında geri bildirim, yüksek düzeyde etkileşim, meydan okuma ve rekabet, rastgele sürpriz unsurları ve öğrenme için zengin ortamlar yapılandırmak için entegre bir platformda birleştirilmelidir.

Bir oyunun öğrenme aracı olabilmesi için zihinsel uyarım (bazen fiziksel) içermesi ve pratik beceriler (karar almak, seçim yapmak, öncelikleri belirlemek, sorun çözmek, vb.) geliştirmesi gerekmektedir. Oyunlar, bazen büyük dağınık toplulukları içeren sosyal ortamlar olabilir. Kendi kendine öğrenme yeteneklerini ima eder (Baptista & Vaz de Carvalho, 2010). Oyunlar, öğrenmenin diğer gerçekliklerden aktarılmasına izin verir ve doğası gereği çoklu duyuların katılımıyla deneyimseldir.

Van Eck (2006), oyunların eğlenceli olduklarından değil, sürükleyici, oyuncunun sık sık önemli kararlar vermesini gerektirdiği, akılcı hedeflere sahip, oyuncuya bireysel olarak uyum sağlaması ve sosyal ağ içerdiği için etkili öğrenme ortamları olabileceğini savunmaktadır.

Garris, Ahlers ve Driskell'in (2002) oyun temelli bir öğrenme modelini ele alırsak, eğitici oyunun temel özelliği, öğretim içeriğinin oyun özellikleri içinde bulanıklaşmasıdır. Öğrenciler, oyunda başarılı olabilmeleri için uyarlamaları gereken yeni konseptler sürekli olarak sunulsa da, deneyimin “öğrenme” kısmını unutarak oyunu oynuyorlar ve eğleniyorlar. Döngüleri tekrarlayan oyun tasarımı aracığıyla oyuncunun motivasyonu güçlendirmeliyiz. Oyuncudan, oyundaki etkileşim ve geri bildirimin sonucu olarak duygusal ve bilişsel tepkilere dayalı arzu edilen davranışları göstermesi beklenir.

Gee (2003), yüksek öğrenme potansiyeline sahip video oyunlarının özelliklerinin iki kategoriye ayrıldığını ileri sürer. Oyun dışı bağlamlarda da görülebilen “oyun dışı” özellikler ve daha çok oyun ile ilgili olan “oynanabilirlik" özellikleri. Bu ayrıma rağmen, "oyun dışı" özelliklerin "oyundan" ayrılmış olsalar da öğrenmede işe yarayacağı varsayılmamalıdır.

Yüksek öğrenme potansiyeline sahip oyunların "oyun dışı" özellikleri şunlardır:

• Karmaşık sistemler için empati - Değişkenlerinin nasıl etkileşimde bulunduğunu anlamak için karmaşık sisteme içeriden bakmak.

• Deneyim simülasyonları ve eylem hazırlıkları.

Video oyunlarında oyuncular sanal dünyayı, oyunu kazanmak için gerçekleştirmeleri gereken faaliyetlere göre anlamlandırırlar.

• Akıllı aracın oluşturulmasıyla dağıtılmış zekâ

İyi video oyunları, durumun makro ve mikro görünümünü temsil etmek için gerçek dünyadaki bir kişi ile yapay olarak zeki sanal karakterler arasında bağ kurdurur.

• Çapraz işlevli ekip çalışması

İyi video oyunları, okullar ve işyerleri gibi kurumlar için işbirliğini ve işlevler arası ekip çalışmasını öğretebilir. Oyuncular birbirleriyle gerçek dünya özellikleri açısından değil, işlevsel oyun kimlikleri aracılığıyla etkileşim kurarlar. Ayrıca gerçek dünyadaki ırklarını, sınıflarını, kültürlerini ve cinsiyetlerini stratejik kaynaklar olarak kullanmayı seçebilirler. Ancak önceden belirlenmiş ırk, cinsiyet, kültürel sınıf kategorilerine zorlanmazlar.

- Konumlanmış anlam

İnsanların kelimeleri gerçek eylemler, işlevler ve problem çözme ile ilişkilendirebilmesi için diyalog ve deneyim gereklidir. Video oyunları deneyim simülasyonları olduğundan, dili belirli bağlamlara yerleştirebilirler.

Etkili öğrenmeyle ilgili "iyi bir oyunun" özellikleri şunlardır:

• Motivasyon

Video oyunları, oyuncular için son derece motive edicidir. Bu oyunlar eğer öğrenme için bir temel oluşturacaksa bu motivasyonun kaynaklarını anlamak önemlidir.

• Başarısızlığın rolü

Başarısızlık sıklıkla öğrenmeye giden bir yol olarak görülür. Oyunlardaki bu başarısızlık özellikleri, oyuncuların daha yüksek riskler almasını sağlar.

• Rekabet ve işbirliği

Gençler de dâhil olmak üzere birçok oyuncu, oyunlarda diğer oyuncularla rekabet etmekten zevk alır. Ancak rekabeti okulda zevkli ve motive edici olarak görmeyebilir. Video oyunlarındaki rekabet, oyuncular tarafından kazanmak ve kaybetmek kadar sosyalleşme olarak görülüyor.

Oyunların tasarımı aşağıdaki özellikler ile ilgilidir:

• Etkileşim - Oyuncu bilgiyi pasif olarak tüketmekle kalmayıp içerik üzerinde kontrole sahiptir.

• Özelleştirme - Öğrenme türlerini temel alarak başarıya giden farklı yollar sağlar;

• Güçlü Kimlikler - İyi oyunlar, oyuncu tarafında derin bir yatırımı tetikleyen ve sanal dünyada gerçekleştirilmesi gereken işlevler, beceriler ve hedeflerle açıkça ilişkilendirilen oyunculara kimlikler sunar.

• Doğru Sıralanmış Problemler - Bağlantılı öğrenme yaklaşımlarında, karmaşık alanlarda etkili öğrenme için sıralamanın çok önemli olduğu bilinmektedir.

• Düşük Hayal Kırıklığı Seviyesi - Oyun zorluklarının bazı oyuncuların oyunu zorlu ama başarılabilir olarak deneyimleyebileceği şekilde ayarlanması

• Bir Uzmanlık Döngüsü - Tekrarlanan genişletilmiş uygulama döngüleri ve ustalık testleri

• "Derin" ve "adil" - Oyun zorlayıcı olmalı ancak başarıya götürecek şekilde düzenlenmelidir.

Oynanış öğeleri başlangıçta basit ve öğrenmesi kolay olmalı ve oyuncu daha fazla ustalaştıkça daha karmaşık hale gelmelidir.

Eğitim ortamında kullanılan bilgisayar oyunlarının motivasyonu artırdığı kanıtlanmıştır. Ancak yüksek motivasyona sahip öğrenciler, kaliteli bir öğrenmenin gerçekleşmesi için yeterli değildir. Yüksek motivasyonla birlikte iyi öğrenme materyallerine ihtiyacımız vardır. Böylece öğrenciler bir bilgisayar oyunu biçiminde sunulan materyallerden yeni bilgiler kazanacaklardır.

Oyunun motivasyon üzerindeki olumlu etkisine rağmen öğretmenler hala öğretimde ciddi oyunlar kullanmaktan çekinmektedir. Oyunlar sınıfta kullanılmak için çok zaman alıcı olabilir. Bu nedenle öğretmenler ciddi öğrenme oyunlarını genellikle ev tabanlı bir öğrenme aktivitesi olarak veya giriş derslerinde bir motivasyon dersi olarak kullanabilirler.

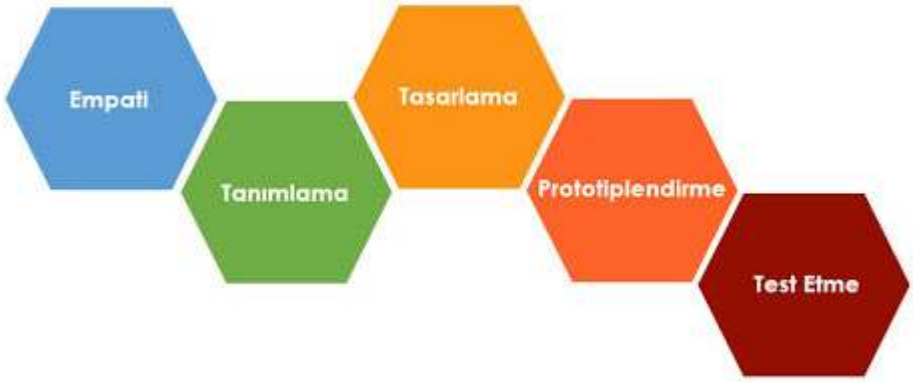
2.3. Tasarım Odaklı Düşünme Metodolojisi

Tasarım Odaklı Düşünme fikirleri 1960'ların sonunda ortaya çıktı. Ancak bir kavram olarak "Tasarım Odaklı Düşünme", 1980'lerin sonlarından itibaren yoğun olarak 1990'larda Stanford Üniversitesi'nde ortaya çıktı. Yöntem, iş dünyasında IDEO'dan(https://www.ideo.com/eu) Tim Brown tarafından tanıtıldıktan sonra yaygınlaştı. "Tasarım Odaklı Düşünme", insanların ihtiyaçlarını, teknoloji yeteneklerini ve iş başarısı gereksinimlerini entegre etmek için tasarımcının araçlarından yararlanan, inovasyon içeren insan merkezli bir yaklaşımdır 21. yüzyılın başında "Tasarım Odaklı Düşünme" son derece popüler hale geldi. BT şirketleri de ürünlerini geliştirirken bu yaklaşımı

uygulamaya başladı. Örneğin, SAP 2005'ten beri bir problem çözme felsefesi olarak "Tasarım Odaklı Düşünme" uyguladı. Sonuç olarak yeni ürünlerin geliştirilmesinde yenilikçi bir son kullanıcı odaklı yaklaşım ile olumlu sonuçlar elde etti. P&G, IBM Design, GOOGLE, AMAZON ve CİSCO gibi şirketler Tasarım Odaklı Düşünme'yi tüm işlerine entegre ettiler.

Stanford Üniversitesi tarafından geliştirilen konsept, "Tasarım Odaklı Düşünme"'yi "yaratıcı problem çözme ve inovasyon yaratmada lider bir metodoloji" olarak tanımlamaktadır. Her gün dünya çapında binlerce şirket ve kuruluş tarafından bu metodoloji kullanılmaktadır. Amaç, gençlerde 21. yüzyılın becerilerini olan takım çalışması, problem çözme, yaratıcılık, empati, güven, sabır, konsantrasyon ve deney becerilerini geliştirmektir.

Stanford modeli 5 aşamadan oluşmaktadır.



Şekil 2.1: Stanford Modelindeki Aşamalar.

Detaylı Bilgi için aşağıdaki linkte yer alan dokümanı inidirp okuyabilirsiniz https://static1.squarespace.com/static/57c6b79629687fde090a0fdd/t/5b19b2f2aa4a99e99b26b6bb/1528410876119/dschool_bootleg_deck_2018_final_sm+%282%29.pdf

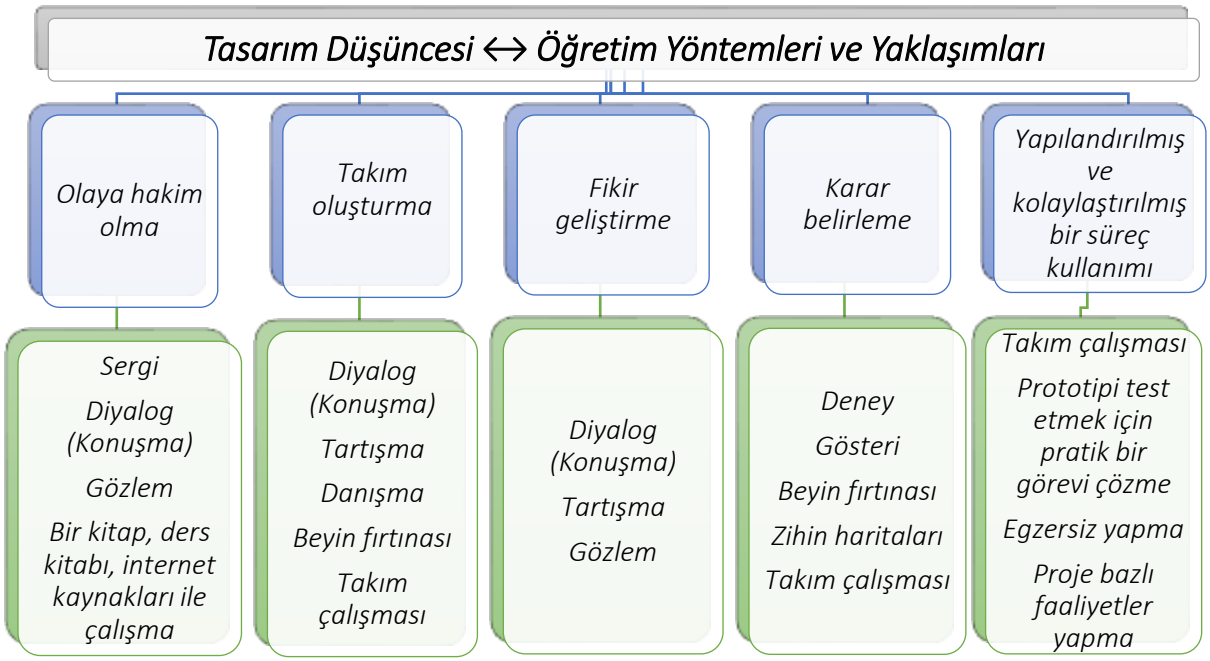
"Tasarım Odaklı Düşünme"'nin ana aşamaları ve özellikleri aşağıda özetlenmektedir.

Uygulama	Prensipler	Aşamalar		
		Kısa süreç	Uzatılmış süreç	Standford Okulu Modeli
Kullanıcı ihtiyaçları	Keşif - Yeni bir	Kullanıcıların davranış ve		Empati

hakkında derinlemesine empatik bir anlayış geliştirmek	meydan okumayla karşılaşmak. Nasıl çözeceksin?	deneyimlerini gözlemlemek ve incelemek. Empati yoluyla bir sorunu ve bakış açısını araştırmak, tanımlamak.	yapmak
Heterojen takımlar oluşturmak	Yorum - Ne öğrendim ve nasıl yorumlayacağım?	Kısa sürede birçok fikir üretmek	Tanım
Diyalog temelli konuşmalar	Fikir - Bir fırsat görüyorum. Onu nasıl bir fikir haline getirebilirim?	Fikirlerin seçimi ve sınıflandırmak	Fikir
Deney yoluyla kararlar üretmek	Deney yapma - Nasıl inşa edeceğime dair bir fikrim var mı?	Prototip oluşturma - Kullanıcılara hızlı geri bildirim alan bir ürün tasarlatmak	Prototip
Yapılandırılmış ve kolaylaştırılmış bir süreç kullanmak	Evrim - Yeni bir şey denedim. Onu nasıl geliştirebilirim?	Test / Geri Bildirim - Prototip geliştirilmesi	Test
			Fikrin işe yarayıp yaramadığını test etmek
			Geri bildirime dayalı öğrenme ve iyileştirme

Tablo 2.1: “Tasarım Odaklı Düşünme”’nin Özellikleri ve Aşamaları.

“Tasarım Odaklı Düşünme” yalnızca işletmeye değil aynı zamanda sivil toplum, eğitim ve sağlık sektörlerine de fayda sağlar. Giderek daha fazla eğitim kurumu “Tasarım Odaklı Düşünme” yöneliyor. Yavaş yavaş eğitimde “Tasarım Odaklı Düşünme”’nin uygulanmasına yönelik iyi uygulamalar artmaktadır (Georgieva, Tuparova). “Tasarım Odaklı Düşünme” yaklaşımının uygulanmasında öğretmenin “Tasarım Odaklı Düşünme” modelindeki bireysel aşamalar ile uygulanabilecek uygun öğretim yöntemleri arasında bir bağlantı kurması gerekir. Bu anlamda “Tasarım Odaklı Düşünme” çok çeşitli öğretim yöntemlerinin uygulanmasını gerektirir.



Tablo 2.2.: Öğretim Yöntemleri ve "Tasarım Odaklı Düşünme" (Georgieva, Tuparova).

2.4. Öğrenme Teorileri ve Oyun Temelli Öğrenme

Geçmişte pek çok eğitici bilgisayar oyunu davranışsal öğrenme teorisine göre tasarlanmıştı (Rugelj ve Lapina, 2019). Bu oyunlar programlanmış talimatlar şeklinde uygulanmışlardı. Öğrencilere, bir soru veya çözülecek başka bir görev veya problem şeklinde bir uyarıcı sunulmaktadır. Öğrenciler sunulan cevaplardan birini seçerek hemen yanıt verir. Cevap doğruysa, oyun olumlu bir karakter tepkisi veya olumlu duyguları harekete geçiren mutlu bir melodi şeklinde olumlu tepki verir. Bu etki-tepki çifti örneği, bir soru ile doğru cevap arasındaki bağlantıyı zorlaştırmaktadır. Yanlış cevap verilmesi durumunda olumsuz uyaran şeklinde tepki verilir ve bağlantı zayıflatılır. İşaretle ve tıkla oyunlar ve sınavlar, alıştırma ve uygulama konseptine sahip olup davranışçılığa dayalı oyunların tipik temsilcileridir. Temel aritmetik işlemleri öğrenmek veya kesikli verilerin ezberlenmesini desteklemek için uygundur. Yani Bloom Taksonomisinin en düşük seviyelerindeki öğrenme hedeflerine uygundur.

Bilişsel öğrenme teorisi, öğrencinin bilişsel aktivitesini ve uygun zihinsel modellerin oluşumunu vurgular. Öğrenciler, öğretmeninden temel kavramları öğrenir veya öğrenme kaynakları oluşturur. Ardından yeni bilgiler edinmek için mantıksal çıkarımı kullanır. Bilişsel yaklaşımın karar verme ortamlarına örnek olarak bulmacalar ve strateji oyunları düşünülebilir. Bilişsel teori tabanlı oyunların en gelişmiş biçimleri, kişiselleştirilmiş öğrenme materyali sağlamak için öğrenci ve uzman bilgilerini modellemek için makine öğrenimi algoritmalarının kullanıldığı akıllı eğitim sistemlerine dayanmaktadır.

Yapılandırmacılık, öğrenenlerin kendi bilgilerini oluşturmalarını öneren alternatif bir görüştür. Öğrenme, kullanıcının hâlihazırda sahip olduğu bilgi üzerine yinelemeli olarak inşa edilen aktif bir inşa etme sürecidir (bilgi edinme yerine). Bir inşa sürecinde, yeni uygulanabilir zihinsel modeller oluşturmak için duyusal veriler mevcut bilgilerle birleştirilir ve bunlar da daha fazla inşa için temel oluşturur.

Yapılandırmacı öğrenme, öğrencilerin kendi bilgilerini oluşturdukları bir ortama (bilgisayar oyunuyla modellenebilecek) yerleştirilmeleri gerektiğini savunarak keşif ve sorgulayıcı öğrenmeyi vurgular. Yapılandırmacı öğrenme görüşünü üç temel ilke tanımlar:

- Her kişi kendi bilgi temsilini oluşturur.
- Öğrenme, öğrencilerin mevcut bilgi ile deneyimleri arasında bir tutarsızlık ortaya çıktığında gerçekleşir.
- Öğrenme sosyal bir bağlam içinde gerçekleşir ve öğrenciler ile akranları arasındaki etkileşim öğrenme sürecinin gerekli bir parçasıdır.

Öğrenme materyalleri, bilgiyi davranışçı bir tarzda beyan etmekten ziyade bilgi oluşumunu destekleyenöğretim sağlar. Bilgisayar oyunu simülasyonları, çeşitli gerçek hayat senaryolarını bilgisayar oyunu formatında sunar. Öğrencinin belirli bir role sahip olduğu soyutlanmış gerçeklik modelini kullanırlar. Öğretmenin görevi, öğrenci öğrenirken yani uygulanabilir zihinsel modeller inşa ederken öğrenciye rehberlik ve geribildirim sağlamaktır.

Oyunlar, oyuncunun tutumlarında, davranışlarında ve becerilerinde değişikliklere yol açabilir. Shute ve Ke (2012), iyi bir oyunun temel unsurları ile üretken öğrenmenin özellikleri arasında benzerlikler olduğunu bulmuştur. Oyun tasarımının bize öğrenme hakkında öğreteceği çok şey vardır ve çağdaş öğrenme teorisi bize daha iyi oyunlar tasarlama konusunu öğretebilir (Shute, Rieber & Van Eck, 2012). Kanadalı iletişim teorisi filozofu olan ve World Wide Web'i geçen yüzyılın

altmışlı yıllarında küresel köyden söz ederken öngören Marshall McLuhan şunu ifade etmektedir: “Oyunlar ve öğrenme arasında bir ayrım yapan kimse, her ikisini de bilmiyor. "

Whitton ve Moseley (2012), aktif bir öğrenme perspektifinden ciddi oyun tasarımında iyi uygulama için bir çerçeve önermiştir. Çerçevedeki yönergeleye göre oyun ortamı keşif, problem çözme ve sorgulamayı teşvik ederek aktif öğrenmeyi desteklemeli, açık ve ulaşılabilir hedeflerle katılım sağlamalı, öğrenme bağlamına uygun olmalı, yansıtma için fırsatlar sağlaayarak desteklemelive tüm öğrenciler için eşit fırsatlar sunmalıdır.

Çeşitli araştırmalar, ciddi oyunların motivasyon, katılım ve eğlence ile öğrenmeyi destekleyebileceğini göstermiştir (Hijon-Neira ve diğerleri, 2015). Programlamayı öğrenmek, mantıksal düşünme, problem çözme ve soyut kavramları anlama yeteneği gibi birçok yetkinliği gerektirir. Bu nedenle, birçok öğrenci bilgisayar programlamayı öğrenmeyi zor bulmaktadır. Bu gerçek, programlamaya giriş dersleri için düşük motivasyona yol açabilir. Motivasyonu artırmak ve öğrencilerin programlamaya yönelik öğrenme tutumlarını geliştirmek için öğretmenler, öğrenmeye yönelik uyarıcı yaklaşımlar aramaktadırlar.

Programlama en iyi uygulama ile öğrenilir ve eğer öğrenciler etkili bir şekilde öğreneceklerse, bu uygulamanın en azından bir kısmının kendi kendilerini yönetmeleri veya akranlarıyla işbirliği içinde olması gerekecektir. Öğretmenin kilit rolü, öğrencileri bunu yapmaya ikna etmek ve böylece onları motive etmektir (Feldgen & Clua, 2004).

Giriş seviyesi bilgisayar programlama dersindeki öğrenciler programları tasarlar ve geliştirir (Rugelj & Lapina, 2019). Bu tipik bir aktif öğrenme yaklaşımıdır. Programlamayı öğrenmenin çok etkili bir yolu olduğu kanıtlanmış olan gruplar halinde proje çalışmasını başlatırsak (Nančovska Šerbec, Kaučič & Rugelj, 2008), aslında deneme amaçlı öğrenme ilkesinden bahsedebiliriz. Deneme mantıklı öğrenme planı, öğrencilerin sistematik bir süreçte ortak eseri (yani bizim durumumuzda bilgisayar programı) birlikte geliştirdiği, değiştirdiği veya yarattığı öğrenme biçimlerinden oluşur (Kafai, 1995). Yalnızca insanlar arasında ("diyalojik” yaklaşım) veya kişinin zihninde ("monolojik" yaklaşım) değil, somut eserlerin yaratılmasıyla ortaya çıkan etkileşime odaklanır (Paavola ve Hakkarainen, 2005).

3. KIZLAR, BİT VE CİDDİ OYUNLAR

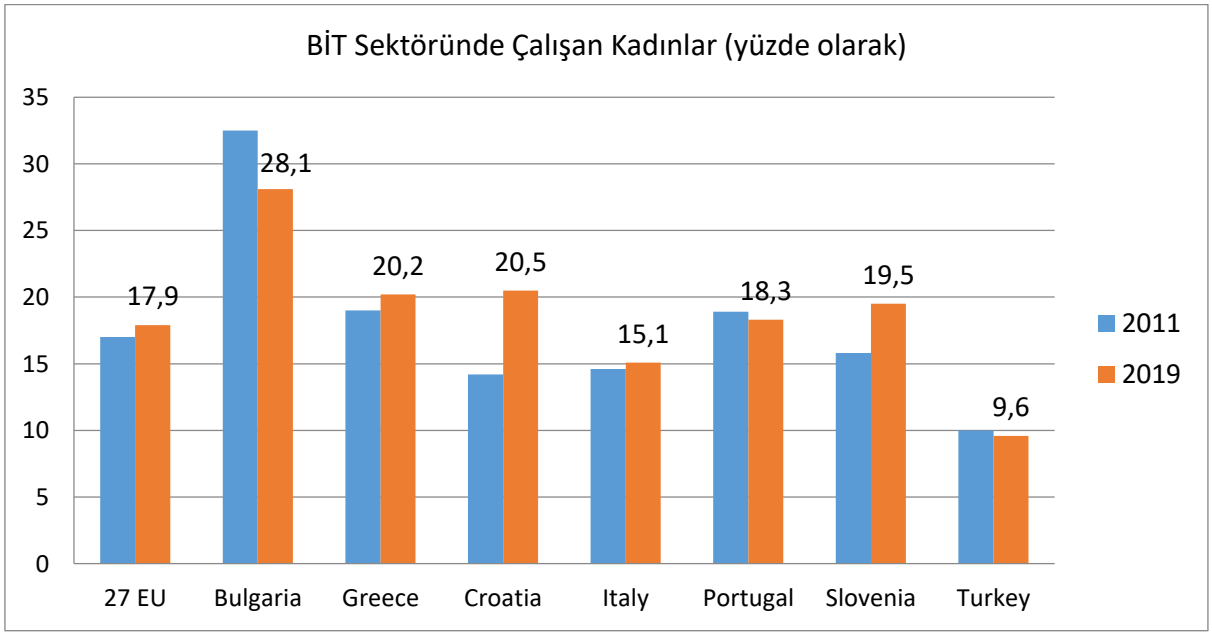
3.1. Kadınların BİT Sektöründeki Konumu

Dijital toplum, modern insanların yetkinliklerine artan talepler doğurmaktadır. Yalnızca dijital okuryazar olmayan, aynı zamanda insan faaliyetinin çeşitli alanlarında yazılım uygulamaları oluşturma ve sürdürme becerisine sahip kişilere artan bir ihtiyaç vardır. Medya, bilgi ve iletişim teknolojileri (BİT) sektöründeki nitelikli uzman eksikliğini sürekli olarak dile getirmektedir. Bu sektörde sürekli bir büyüme mevcuttur.

2019'da Avrupa Birliği'nde (AB) yaklaşık 7,8 milyon kişi BİT uzmanı olarak çalışıyordu. Ancak bu, AB'de toplam istihdamın yalnızca%3,9'udur.

Tablo 3.1'e göre 2011'de AB-27'de BT sektöründeki kadınlar %17 ve 2019'da yaklaşık %17.9'du. 2007-2019 dönemi EUROSTAT verilerine göre, ortalama olarak 27 AB üye ülkesinde, BT sektöründeki kadınlar %22.5'ten %17.9'a düştü. Kadınlar, AB'de BİT eğitimi almış toplam kişi sayısının %17,3'ünü oluştururken 2009'da %20,2'ye düştü.

Coding4Girls proje ortağı ülkelerde BİT sektöründe kadın istihdamındaki durum Tablo 3.1'de sunulmuştur.



Tablo 3.1: 2011 ve 2019'da BİT Sektöründe Çalışan Kadınlar (yüzde olarak).

3.2. Kızların Ciddi Oyunlara Yaklaşımı

Vermeulen v.d. (2011) tarafından yapılan araştırmanın sonuçları, cinsiyet farklılıklarının tutarlı bir şekilde mevcut olduğunu, ancak önceki deneyimlerin bulguları önemli ölçüde etkilediğini göstermiştir (Rugelj ve Lapina, 2019). Kadınlar erkeklere göre daha sık ama daha kısa oyun oynarlar. Soyut, kısa ve sıradan (ör. Tetris) ve sosyal ağ oyunları gibi ustalaşması kolay oyunları tercih ederken, erkeklerin "temel" türleri oynama olasılığı daha yüksektir. Bu kategori, zaman alan ve genellikle atıcılar, dövüş, aksiyon-macera, spor, yarış, strateji, rol yapma ve MMO oyunları gibi yüksek kaliteli üç boyutlu grafiklere sahip olan beceri tabanlı oyunları ifade eder. Çekirdek olmayan türler arasında platform, macera, simülasyon, parti, ciddi, klasik ve gündelik oyunlar bulunur. Ana türler erkekler tarafından kadınlardan daha fazla oynanır. Çalışma, kadınların bulmaca oyunlarından daha çok hoşlandığını, yarış, ritim, simülasyon ve sanal oyunların hem kadınlar hem de erkekler tarafından oynandığını ortaya koydu. Başka bir araştırma, kadınların parti oyunlarını (müzik ve dans oyunları gibi) ve klasik "retro" oyunları sevdiğini tespit etti.

Alserri (2018), dijital oyun oyuncularının motivasyon unsurları, etkili eğitici oyun unsurları ve kadın tercih unsurları gibi etkili ciddi oyun unsurları hakkında kapsamlı bir anket yaptı. Anket sonuçlarını, ciddi oyunlarda cinsiyet temelli katılım için kavramsal model oluşturmak için kullandı. Bu modeli C4G Projesi, oyun tasarımına dayalı öğrenmede öğrenciler tarafından tasarlanacak ve uygulanacak uygun oyun seçimi yoluyla kızların programlama motivasyonunu artırmak için kullanmaktadır.

Araştırma, çalışmanın belirli bir dijital oyun türünü oynama motivasyonunun cinsiyete bağlı olduğunu ortaya çıkardığını belirtti. Kadınların bilgisayar oyunu oynamadığı klişesi artık geçerli değildir. Bu araştırmada ortaya çıkan dijital oyunlar için cinsiyet tercihlerindeki farklılıklar, Vermeulen ve diğerleri tarafından tanımlanan halihazırda sunulan farklılıklara çok benzer. Kadınlar erkeklerden daha çok keşif ve yaratıcı oyunu tercih ediyor. Erkeklerden daha sık ama daha az süre oynuyorlar ve oyun türlerine yönelik tercihleri de farklı. Erkeklerin kadınlardan daha çok bilgisayar oyunu oynadıkları görülmüştür. Kadınlar ayrıca bulmaca oyunlarını, eğitici oyunlar, simülasyon oyun türlerinde sunulan ödüllü sosyal oyunları, işbirlikçi ve keşif oyunlarını, sanal hayatı, sanal dünyayı ve parti oyunlarını tercih ediyor. Dahası, kadınlar macera oyunları oynamayı sever, ancak kendileri oynamadan önce başkalarını gözlemlemeyi tercih ederler. Kadınlar için oyun oynarken motivasyon tercihi unsurları; meydan okuma, kaçış, eğlence, sosyal etkileşim, motivasyon, fantezi, rekabet ve uyarılmadır. Hem erkekler hem de kadınlar yarış, simülasyon ve sanal oyunları sevmektedirler.

Phan v.d. (2012), bilgisayar oyunu kullanımı, tercihi ve davranışları açısından erkek ve kadın oyuncular arasındaki cinsiyet farklılıkları konusunda yaptıkları çalışmada çok benzer sonuçlara varmıştır.

Tuparova, Tuparov, Veleva'nın (2019 ECGBL) çalışmasına göre, kızların ve erkeklerin oyunların özellikleri ve öğelerine olan tercihlerinde farklılıklar vardır. grafik tasarım, ses. oyunu farklı cihazlarda oynama imkânı, (bilgisayar, tablet, akıllı telefon vb.) birçok oyuncuyu dahil etme imkanı, çevrimiçi oynama imkanı, daha karmaşık oyun seviyelerine geçme imkanı bunlar arasındadır. Kızlar için oyunların en çok tercih edilen özelliği ise daha karmaşık oyun seviyelerine geçme imkânı, erkekler için en çok tercih edilen oyunun mantığı ve oyunun senaryosudur. Kızlar için en az ilgi gören özellik oyunlardaki ses, erkekler için ise oyundaki ödüllerdir. Son kullanılan cihazla ilgili olarak akıllı telefon hem erkek hem de kızlar tarafından en çok kullanılan oyun cihazıdır. Kızlar için ikinci kullanım cihazı dizüstü bilgisayar olup onu tablet takip ediyor. Erkek çocuklar için ikinci kullanım bir masaüstü bilgisayar ve dizüstü bilgisayardır. Yazarlar, erkeklerin kızlardan daha çok hikâye oyunlarını, kızların ise erkeklerden hafıza oyunlarını daha çok tercih ettiklerini gözlemlemişlerdir. Cinsiyetten bağımsız tercih edilen oyunlar sosyal unsurları olan oyunlar, dikkat/konsantrasyon, tepki hızı oyunları, rol oyunları, bulmacalar, macera oyunlarıdır. Hem kızlar hem de erkekler için en az ilgi gören oyun türü ise bulmaca oyunudur.

Oyun tasarımına dayalı öğrenmeye dayanan ve kızlar için ilginç ve motive edici görevler hazırlayan C4G Projesindeki özel durumu dikkate almalıyız. Programlamayı animasyon filmler (ör. hikâye anlatımı) veya basit oyunlar oluşturmak için bir araç olarak sunan bir kodlama ortamı, kızlar için uygun olabilir. Çünkü kızlar oluşturmak istedikleri bir hikaye veya basit bir oyun fikirini kodlama ile üretebilirler (Wolz, 2009). Kelleher, Pausch ve Kiesler (2007), her iki durumda da sırasıyla Storytelling Alice ve Generic Alice programlarını kullanarak çocuk programcılar üzerine yaptığı araştırmalarda, kızların ve erkeklerin bilgisayar programcılığı konusunda benzer deneyime sahip olduklarını, programlamayı öğrenmede eşit derecede ilgilendiklerini ve eşit derecede etkili olduklarını tespit ettiler. Dolaysıyla, programlama performansı, cinsiyetten bağımsız olarak kullanıcıların programlamaya harcadıkları süre ve önceki programlama deneyimleri ile ilişkilidir.

4. PROGRAMLAMAYI OYUNLAR ÜZERİNDEN ÖĞRETMEK

Bu bölüm, çocuklara programlama becerileri öğretmek için kullanılabilecek platformları/oyun ortamlarını ve farklı yaklaşımları sunmaya ve karşılaştırmaya odaklıdır.

4.1. Programlamayı Oyunlar Üzerinden Öğretmenin Yaklaşımları

Şu anda yeni kodlayıcılara programlamayı öğretmek için kullanılan yaklaşımlar geniş bir yelpazededir.

Yaklaşımlar şu şekilde sınıflandırılabilir:

- Oyun tabanlı tasarımla programlamayı öğrenmek – Oyun tasarımı tabanlı öğrenme olarak da bilinir ve oyun tasarlayarak ve oluşturarak nasıl kodlanacağını öğrenmek için programlama dillerinin kullanımı anlamına gelmektedir.
- Oyun tabanlı ortamda programlamayı öğrenmek.
- Oyun tabanlı ortamda oyun tasarımı ile programlamayı öğrenmek. C4G metodolojisi programlama öğrenme ortamlarının bu kategorisine aittir.

Bunların bir örneği, özellikle kullanıcı dostu olan blok tabanlı görsel programlama dillerinin kullanımıdır. Bunların kullanımı kolaydır (taşıma ve bırakmaya dayalı). Yazım ve mantıkla ilgili hataları önlerler (Ouhabi et al, 2015). İçerikleri kolay anlaşılır bir biçimde sunarlar. Scratch (Resnick v.d., 2009), (Scratch, n.d.), Snap! (Weintrop et al, 2015), (Snap!, n.d.), veya Alice 2/3 (Alice, n.d.), görsel programlama dillerinin iyi bilinen ve başarılı örnekleridir. Sırada, en çok bilinen görsel programlama dillerinin (ve ortamlarının) karakteristiklerini karşılaştıran bir tablomuz var.

	PLATFORM	HEDEF KİTLE	DİL(LER)	ÜCRET
Scratch	Web-tabanlı (Çevrimdışı versiyonlu)	8-16	Görsel bilgisayar programlama dili	Bedava
Snap!	Web-tabanlı (Çevrimdışı versiyonlu)	12-20	Görsel bilgisayar programlama dili	Bedava
Alice	Windows, Mac OS X, Linux	12-20	Görsel bilgisayar programlama dili	Bedava
Tynker	Web-tabanlı, iOS	7+	Görsel bilgisayar programlama dili, JavaScript, Python	Ücretli

Tablo 4.1. Bazı görsel blok tabanlı programlama ortamlarının örnekleri.

Bilgisayar oyunları da kodlamayı öğrenmek için kullanılabilir. Ya öğrencileri kendi oyunlarını geliştirmeye ve yapmaya teşvik ederek (oyun tasarımı ile öğrenme) ya da onlara öğrenme çıktıları, programlamayla ilgili öğrenme çıktılarını kapsayan ciddi oyunları oynama fırsatı sunarak (oyun tabanlı öğrenme) yapılabilir. Eğitim aktiviteleri için amaçlanan bilgisayar oyunlarının motive edici ve eğlenceli bir öğrenme ortamı yaratma potansiyelleri vardır. Çünkü eğitici standartları karşılayan, öğrenme görevleri barındıran, geri dönüş veren ve yüksek eğitici sonuçları olabilecek aktiviteleri kapsarlar.

Programlama öğretmek için oyunları kullanma fikrinin arkasında bu oyunların daha sürükleyici ve motivasyonel olmalarından dolayı öğrencilere sayısal düşünmeyi ve programlama becerilerini eğlenceli ve tanıdık ortamlarda öğrenmelerine, o becerileri bir programlama dilini öğrenmeye aktarmadan önce fırsat tanır (Kazımoğlu, 2012). Yine de Kazımoğlu ve diğerlerine göre programlama ve sayısal düşünmeyi öğrenmeye yönelik ciddi oyunlar sadece eğlence için yapılmamalı ve bir müfredatı ve becerileri göz önünde bulundurmalıdır ki farklı programlama yapıları arasında ayırım yapılsın ve iyi programlama uygulamaları teşvik edilsin (en yüksek puana ulaşmak gibi bir oyunlaştırmayla).

Ciddi oyunlar aynı zamanda sıkıcı uygulamaları ilginç tecrübelere dönüştürerek programlama için faydalı olabilir. Shabanah (2010) Algoritma Oyunu Tasarlayıcısı yapmıştır ve algoritma görselleştirme sistemlerinde etkinliği artırmak için algoritma oyunlarının yapılışını kolaylaştırmayı amaçlamıştır. Grivokostopoulou (2016) yapay zekâ algoritmalarını öğretmeye yönelik Pacman oyununu baz alan bir oyun geliştirmiştir. Bu oyunla algoritma görselleştirmeleriyle ve animasyonlarla öğrencilere bir algoritmanın işleyiş biçimi, işlevleri ve belirli parametrelere göre uygun kararların nasıl verildiği gösterilerek öğrenme süreçlerinde yardım edilmesi amaçlanmıştır.

Hızla yayılan daha hızlı İnternet bağlantısının edinilmesi ve gitgide daha fazla kişinin evinde bilgisayar olmasıyla CodeCombat ve LightBot gibi çevrim içi oyun tabanlı platformlar ortaya çıkmaya başladı. Combéfis v.d.(2016) programlama becerilerini öğretmek için kullanılan çevrim içi platformların ana türlerini inceledi ve şu faktörlerin ciddi bir oyunu daha motive edici ve başarılı kıldığı sonucuna vardı:

1) Bir geri dönüş ve değerlendirme süreci

2) Estetik
3) İşbirliği ve çok oyunculu yanlar
4) Oyun üzerinden yol gösterme
5) Olumsuz sonuçların olmaması
6) Müzik
7) Oyuncuların ilgisini sürdürmek için orta düzeyde bir zorluk

Programlamayı öğrenmeye yönelik ciddi oyunların öğrenme kazanımlarıyla ilgili olarak, Miljanovic'in (2018) gerçekleştirdiği 49 ciddi programlama oyununun geniş kapsamlı incelenmesine göre; programlamaya yönelik geliştirilen ciddi oyunların çoğu sorun çözmeye ve temel programlama kavramlarına odaklanmaktadır. Çok daha az sayıda oyun ise veri yapılarına, geliştirme metodlarına ve yazılım tasarımına yer vermektedir.

Programlama becerileri öğretmeseler de dolaylı fakat etkili bir biçimde soyutlama (örn. tetris), ayrıştırma (örn. sudoku), algoritmasal düşünme (örn. bulmaca oyunları), değerlendirme (örn. hafıza oyunları), ve genelleştirme (örn. örüntü oyunları) gibi önemli sayısal düşünme becerilerini öğretebilen oyunlar vardır (Franković et al, 2018).

Oyun tasarımı tabanlı öğrenmede, genellikle görsel programlama dillerinin kullanımı ve onların blok tabanlı ortamlarının bir kombinasyonu ile tasarım süreci ve oyunların kodlaması vardır. Yeni programcılarla Scratch ve Pascal (komut dizisi veya metin düzenleyici üzerinden prosedürel dil) kodlama ortamlarının kullanımlarını karşılaştıran bir araştırmada, öğrenenlerin Scratch kullanırken bilgisayar bilimi çalışmalarına devam etmek için çok daha fazla motive oldukları gözlemlenmiştir (Ouahbi et al, 2015). Araştırma aynı zamanda oyunların ve hikâyelerin yaratılmasının öğrenenleri programlama öğrenirken daha yaratıcı ve bağımsız kıldığını gösterir. Weintrop (2015) tarafından lise öğrencilerinin Snap! ve Java kullanımlarını karşılaştıran başka bir araştırma blok tabanlı yaklaşımın Java'dan daha kolay bir öğrenme eğrisinin olduğunu bulmuştur. Böylece, blok tabanlı öğrenmenin (Scratch ve Snap gibi) yeni programcılar için daha erişilebilir olduğu görüşü ortaya konulmuştur. Araştırmanın bulgularına göre bunun sebebi blokların, nasıl kullanılacaklarına dair ipuçları veren görsel yapılarından dolayı daha kolay okunmaları, daha rahat oluşturulmaları ve hafızaya yardımcı olmalarıdır.

Programlama öğretmeye yönelik entegre yaklaşım ve 14 yaşındaki öğrencilerin kazanımlarına etkileri burada tartışılmıştır (Tuparova, Nikolova, Tuparova). Bu yaklaşım iki adımı birleştirmektedir.

- Var olan eğitici bilgisayar oyunlarını kullanmak**:** bilgilerin ve becerilerin edinimi, öğrenme aktivitelerinin motivasyonu, algoritmik düşünmenin gelişimi için oyun kurallarını, arayüz öğelerini ve özellikleri ile etkinliklerini soruşturmak amacıyla var olan eğitici bilgisayar oyunlarını denemek.
- Oyunun matematiksel modeli, grafiksel kullanıcı arayüzü (GUI), kodlama, test etme ve doğrulama da dâhil olmak üzere eğitici bilgisayar oyunları tasarlamak ve geliştirmek.

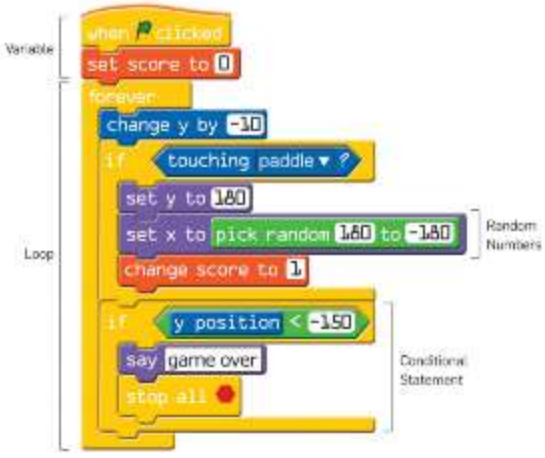
Yaklaşımın uygulanması için C# programlama ortamı kullanılır. Model, informatik okuyan 126 öğrenciyle test edilmiş ve öğrenciler deney ve kontrol grubu olarak iki gruba ayrılmıştır. Öğrencilerin kazanımları pratik ödevler ve kâğıtla yapılan sınavlarla ölçülmüştür. Sonuçlar, deney grubundaki erkek ve kızların kazanımları arasında bir fark olmadığını göstermiştir. Fakat araştırmacılar deney grubundaki kızların kontrol grubundaki kızlardan daha iyi sonuç aldığını bildirmiştir.

4.2. Programlama Öğretmeye Yönelik Oyun Tabanlı Ortamlar

Sırada kodlama ve programlama öğreten oyunları kullanan farklı ortamların kullanımına ait bazı örnek var.

4.2.1. Oyun Tasarımı ile Öğrenmek

Scratch



Resim 4. 1 Scratch Metni Örneği

Scratch (n.d.) bir blok tabanlı programlama ortamıdır. Yani öğrenenlere, kod bloklarını yapıştırarak ve görsel geri dönüş alarak programları birleştirme fırsatı sunar (Weintrop, 2015).

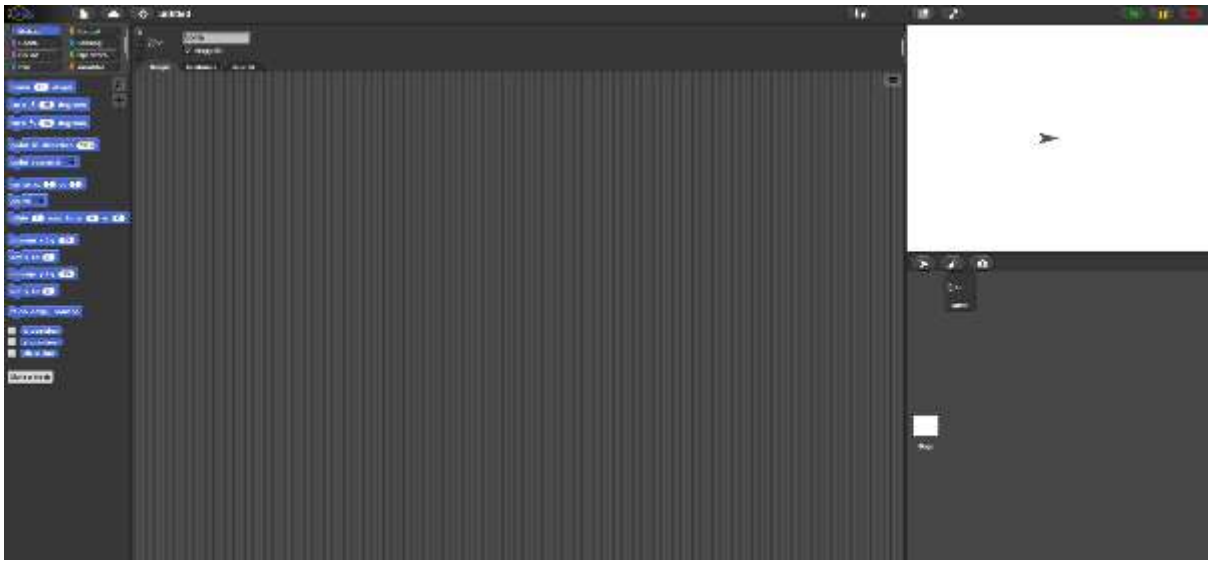
Böylelikle, öğrenenler söz dizimine dikkat etmeden programlamanın temel öğelerine aşina olabilirler (Ouahbi, 2015).

Scratch özellikle daha genç öğrenciler arasında programlamaya bir giriş olarak kullanılabilir. Fakat aynı zamanda diğer programlama ortamlarına geçişi kolaylaştırmak için de kullanılabileceğinden dolayı bilgisayar bilimini tanıtır ve ona ilgi duyulmasını sağlar (Wolz, 2008). Meerbaum-Salant (2013) tarafından yapılan bir araştırmaya göre çoğu öğrenci Scratch üzerinden kayda değer bir seviyede bilgisayar bilimi kavramlarını öğrenebilir. Ancak daha yüksek seviyede soyutlama gerektiren belirli konuları öğretirken zorluklar ortaya çıkar. Fakat bu, süreç içerisinde öğretmenlerin öğrencilere destek olamalarıyla aşılabilir.

Scratch sitesi dünya çapında bir kullanıcı ağını destekler ve kullanıcıların projeler paylaşmasına olanak sağlayan bir programcı topluluğuna ev sahipliği yapar (Meerbaum-Salant, v.d., 2013). Scratch 50'den fazla dilde mevcut olup bunlara proje ortağı kurumların dilleri olan Türkçe, Hırvatça, İngilizce, Yunanca, İtalyanca, Portekizce, Slovence ve Bulgarca dahildir. Aynı zamanda programın bir bilgisayara indirilmesini ve internet bağlantısı olmadan çalışabilmesini sağlayan bir çevrimdışı düzenleyicisi vardır.

Snap!

Snap! (n.d.) de tasarımı Scratch'ten baz alınan ve üzerine ek özellikler eklenen bir görsel blok tabanlı programlama dilidir. Snap!, Scratch gibi ağ tabanlıdır. Fakat bir tarayıcı üzerinden çevrimdışı olarak da çalışabilir.



Resim 4.2 Snap! Kullanıcı Arayüzü

Scratch'in özelliklerine ek olarak Snap! sınıf listeleri, sınıf prosedürleri, sınıf karatkerleri (kukla), sınıf kostümleri, sınıf sesleri ve sınıf devam çizelgeleri barındırır. Böylece Scratch'e kıyasla daha ileri düzeydeki kitleler için daha uygundur. Snap! 40'tan fazla dilde mevcut olup bunlara proje ortağı kurumların dilleri olan Türkçe, Hırvatça, İngilizce, Yunanca, İtalyanca, Portekizce, Slovence ve Bulgarca dâhildir.

Alice

Resim 4.3 Alice 3 Kod Düzenleyici

Alice (n.d), öğrenenlerin yaratıcılığını animasyonların, interaktif anlatımların veya basit 3D oyunların yapılmasıyla destekleyerek onları programlamaya motive etmeyi amaçlayan blok tabanlı bir programlama ortamıdır (Kelleher, 2007). Öğrenenlerin var olan üç boyutlu objelere yenilerini ekleyebilecekleri ve işlevleri ile metodlarını değiştirebilecekleri bir Sanal Dünya Düzenleyicisi barındıran sanal dünyaların oluşturulmasına olanak sağlar. Sanal dünya oluşturulduktan sonra öğrenciler, oyun/anlatım/animasyon mantığını geliştirmek için kodlar yazabilir (Sykes, 2007). Alice, tecrübesi olmayan öğrenenler için iyi bir programlama dilidir. Çünkü animasyonu yapılan programları, kodlarını yazarken görebilmelerini sağlar (Cooper, 2000).

Üstelik, Storytelling Alice'e (Alice 2'yi baz alan ve ek yaratıcı hikâye yazma araçları barındıran bir programlama ortamı) odaklanan başka bir araştırma, Generic Alice ile Storytelling Alice'in kızlara programlama kavramlarını öğretmede eşit derecede başarılı olduklarını; Storytelling Alice ile kızların daha çok motive olduklarını ve programlama sürelerini artırıp programlamaya olumlu bir etkisinin olduğunu bulgu olarak ortaya koymuştur.

Serinin en yeni çıkan ürünü Alice 3, 13 dilde mevcut olup bunlara AB dilleri olan İngilizce, Portekizce, Yunanca, Slovence, Bulgarca, İspanyolca ve Almanca dâhildir.

Tynker

Tynker (n.d.) bir taşı-ve-bırak görsel programlama dilini baz alan eğitici bir programlama platformudur. Sunulan diğer programlama ortamlarının aksine Tynker ticari bir üründür. Scratch'e göre artılarından biri, kodlamaya oyun olarak nasıl yaklaştığıyla ilgildir. Kullanıcıların karakterleri hareket ettirmek, etkileşime sokmak ve farklı görevleri yerine getirmek için bir program kurmaları gerekmektedir. Çocuklar şablonları anlamaya başlasın diye oyunculara çözmeleri gereken sorunlar sunar (Geist, 2016). Tynker aynı zamanda adım adım talimatlar vermesi için içerisinde kurulu bir öğretici barındırır ki öğrenci kodlama kavramlarının nasıl uygulanacağını öğrensin. Öğrencinin aynı zamanda JavaScript kodundaki blokları görselleştirme imkânı da vardır. Bu onlara blokları metin tabanlı bir programlama dilinde anlama fırsatı verir.

Tynker 23'ten fazla programlama kursu, 11 iPad kursu ve 2000'den fazla kodlama aktiviteleri sunmakta ve bireysel planlar ile aile planları (4 kişiye kadar) sağlamaktadır. Uygulama yalnızca İngilizce dilinde mevcuttur.

Resim 4.4 Tynker Kullanıcı Arayüzü

4.2.2. Oyun tabanlı bir ortamda programlama öğrenmek

Kodlama becerilerini geliştirmeye yönelik oyunları baz alan bazı platformlar hakkında bilgileri aşağıda bulabilirsiniz.

Code Combat

Resim 4.5 Code Combat Oyunu

Code Combat (n.d.) ağ tabanlı bir macera oyunudur. Hem bireysel öğrenciler için hem de sınıflar için planları vardır. Aynı zamanda öğretmenlerin derslerde kullanmaları için olan kaynakları da

sunar. 100'den fazla oynaması bedava ve aboneye özel seviyeleri vardır. 50'den fazla dilde mevcut olup bunlara proje ortağı kurumların dilleri olan Türkçe, Hırvatça, İngilizce, Yunanca, İtalyanca, Portekizce, Slovence ve Bulgarca dahildir.

Öğrencilerin karakterleri hareket ettirmek, etkileşime sokmak ve farklı görevleri yerine getirmelerini sağlamak için kendi kodlarını (JavaScript veya Python'da) yazmaları gerekir. Bunun sebebi oyunun blok tabanlı olmamasıdır. Oyun süreç içerisinde ipuçları verir ve kavramları kademeli olarak tanıtır. Oyun 5 farklı dünyaya ayrılmış olup farklı kavramları oyuncu oyunda ilerleme kaydettiktçe tanıtır. Bunlar;

1. Kithgard Zindanı – Söz dizimi, metodlar, parametreler, diziler, döngüler ve değişkenler

2. Backwoods Ormanı – Koşul, Boolean mantığı, ilişkisel operatörler, işlevler, obje özellikleri, olay düzenleme, girdi düzenleme

3. Sarven Çölü – Aritmetik, sayaçlar, while döngüleri, böl, devam et, dizi karşılaştırması, min/max bulma

4. Cloudrip Dağı – Obje değişmezleri, uzak metod, çağrı, for döngüleri, karmaşık işlevler, resmetme, modulo

5. Kelvintaph Buzulu – İleri düzey teknikler.

Miljanovic ve diğerlerine (2018) göre, oyunun eğitici içeriği sorun çözme ve algoritma tasarımının kavramsal yanlarını (sözdizimi & anlambilim, değişkenler & ilkel veri tipleri, ifadeler & görevler, girdi & çıktı, koşullular & yinelemeliler, işlevler & parametreler ve özyineleme gibi), temel progamlama kavramlarını (diziler, heterojenik kümeler ve soyut veri tipleri gibi) ve temel veri yapılarını kapsar.

Human Resource Machine

Human Resource Machine (n.d.) görsel programlama diline dayanan bir bulmaca oyunudur. Bu oyunda oyuncu, bir ofis çalışanını programlayarak ve gitgide zorlaşan bulmacalarla ilerleyerek bir görevi otomatikleştirmek durumundadır (Johnson, 2016). Oyuncu, talimatları taşıyıp bırakarak bir

Resim 4.6 Human Resource Machine Oynanışı

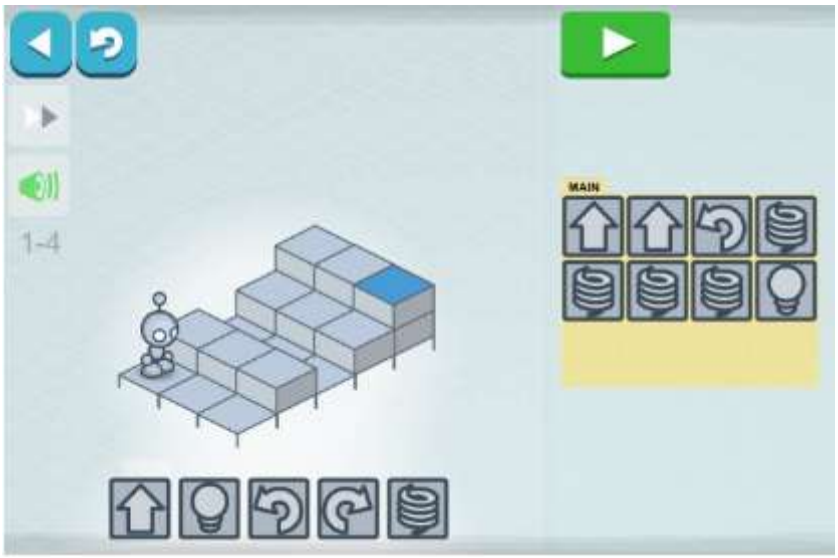
hareket sekansı oluşturmak zorundadır. Yeni komutlar, daha karmaşık kullanımların yapılabilmesi için kademeli olarak tanıtılırlar. Görsel programlama dili üzerinden temel programlama kavramlarını algoritma karşılaştırmasıyla öğretir.

LightBot

LightBot (n.d.) Human Resource Machine'e benzeyen mekanikleri olan ve seviyeleri geçmek için mantıksal düşünme gerektiren bir bulmaca oyunudur. Oyuncu, 40 seviyeyi geçmek için bir robotu tuşları yakması için yönlendirmek zorundadır. Lightbot'ta kullanılan komutlar birer ikon olarak görünür.

Bir programlama dilinin açık bir öğrenimi olmasa da oyuncular problem çözme becerilerine odaklanarak "...sekanslama ve algoritmaların uygulanması hakkında bir anlayış kazanırlar" (p.6) (Miljanovic, 2018). Yine de öğrenciler oyunlar üzerinden koşullular, yinelemeliler, özyineleme ve hata giderme stratejieri gibi farklı temel programlama kavramlarını öğrenirler. Tuşların alanı sınırlı olduğu için öğrencinin kod verimini de öğrenmesi gerekir.

Mathrani (2016) tarafından yapılan bir araştırma, ortaokul seviyesinde olan öğrencilerle Lightbot kullanmıştır ve öğrencilerin oyunları oynamayı sevdiklerini ve bunun işlevler, prosedürler koşullular ve özyineleme gibi programlama yapılarını öğrenmek için etkili bir yol olduğunu tespit etmiştir. Uygulamaya dâhil olan çoğu öğrenci aynı zamanda bu yaklaşımın onların normalde zor kavrayacakları kavramları daha rahat anlaşılır hale getirdiğini ifade ettiler.



Resim 4.4 Lightbot Arayüzü

May's Journey

May's Journey hedef kitlesi doğrudan ortaokul kız öğrencileri olan bu listedeki tek oyundur. Oyuncuların oyunun Java'dan ilham alınan özel programlama diliyle bir labirenti geçtikleri üç boyutlu bir bulmaca oyunudur. Oyun, programlamanın temellerini öğreterek kızların ilgisini bilgisayar bilimine çekmek için tasarlandı. Geliştirici, görsel programlama ve gerçek programlama dillerinin arasındaki geçişi kolaylaştırmak için sahte kodlar kullanmıştır. Eğitim içeriği temel talimatları ve sekansları, mantığı, döngüleri, değişkenleri, if söylemlerini, karşılaştırıcıları, Boolean mantığını, tamsayı ve diziler üstünde operasyonları kapsar (Jemmali, 2016).

Bu oyunda iki aşama vardır. Tipik oyun mekanikli bir keşif aşaması ve bir kodlama aşaması. Kodlama aşamasında, oyuncu çalışan programdan görsel geri dönüş alırken kodu yazabilsin diye ara yüz; kod için ipuçları da keşif aşamasında sağlanır. Hikâyede ana kahraman arkadaşından koparılmış ve yıkılan bir dünyada yaşamaktadır. Oyunun dünyasını düzeltmesi ve gizemleri çözmesi gereklidir. Gizemin her bir parçası kademeli olarak gösterilir ve bu oyuncuları daha fazla oynamaları için motive

eder. Ergenlik öncesi çağdakiler ana hedef kitle olduğu için oyuncu, karakterde kendinden bir şey görebilsin diye ana karakteri bir ortaokul kız öğrencisi gibi tasarlanmıştır. Tasarımda aynı zamanda kızların tercihlerini göz önünde bulundurmuştur (yüksek ışık, sıcak renk şeması ve daha az agresif bir illüstrasyon). Test grubu da sunulan tasarıma olan beğenisini ifade etmiştir.

Resim 4.8. May's Journey Oynanışı

No Bug's Snack Bar

No Bug's Snack Bar taşı-ve-bırak blok tabanlı bir yaklaşımı olan ve sorun çözmeye odaklı ciddi bir araştırma oyunudur. Öğrenme çıktıları değişken manipülasyonu, eylem sekansları, koşullular ve yinelemeliler ile hata giderme stratejieriyle ilgilidir (Vahldick, 2017).

Resim 4.9. No Bug's Snack Bar Oynanışı

Oyunda ana karakter atıştırmalık satılan bir dükkânda çalışıyor ve kod kullanarak farklı görevleri tamamlaması gerekiyor. Bu oyunda uygulanan bir yenilik, öğretmenlerin öğrencilerin nasıl ilerleme kaydettiğini gözlemelerine yarayan bir aracın olmasıdır. Aynı zamanda bir oyunlaştırma yaklaşımı da vardır. Öğrenciler oyunlarda ilerleme kaydederken ve sunulan sorunlara daha iyi çözüm yollarını bulurken puan kazanırlar. Oyunda ipuçları veren bir asistan ve öğretmenin hangi öğrencilerin yardıma ihtiyacı olduğunu görmesini sağlayan idari bir sistem mevcut olup öğretmen onlara kişiselleştirilmiş ipuçlarını bu sistem aracılığıyla verebilmektedir. Böylece bir öğretmenin varlığı temel bir gereksinim olarak görülmüştür.

Robot ON!

Robot ON! C++ öğrenen lisans öğrencilerini hedef alan başka bir araştırma bulmaca türü bir oyundur. Bu oyunda oyuncu birkaç dizi görevle bir "robot giysisi" etkinleştirmek zorunda olan bir bilim insanıdır. Oyuncu bir seviyeyi tamamladıktan sonra yeni bir robot sistemini etkinleştirir. Öğretmenler başka programlama dillerini kullanarak kendi seviyelerini oluşturabilirler. Oyuncunun renk kodlu farklı araçları vardır. Kodlardaki renkler farklı araçlara karşılık gelir. Oyuncu farklı araçları teker teker ve bir öğretici eşliğinde öğrenir (Miljanovic, 2016).

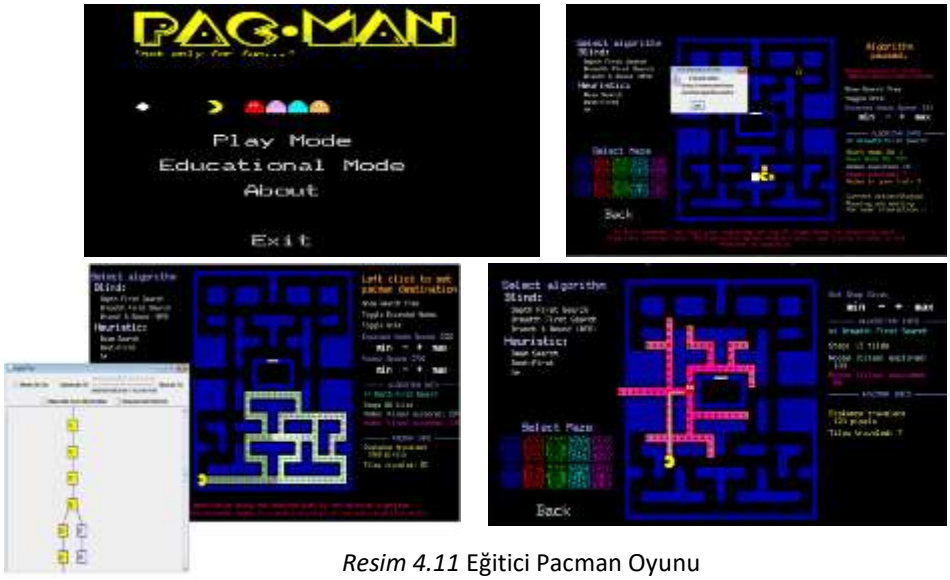
Resim 5 Robot ON! Oynanışı

Kod yazmaya odaklanmaktansa bu oyun hata giderme becerileri ile başkalarının yazdığı kodu anlamayı öğretmek için programlama anlayışına odaklanır. Oyun ilerledikçe oyuncuya şu araçlar verilir:

1. Aktivatör aracı (kontrol akışını öğrenmek için)
2. Yargılama ve yargı kaldırma araçları (kod davranışını öğrenmek için)
3. Adlandırıcı araç (değişken amacını öğrenmek için)
4. Doğrulama aracı (veri akışını öğrenmek için)

Eğitici Pacman Oyunu

Eğitici Pacman Oyunu, arama algoritmalarını öğretmek için tasarlanmış ciddi bir araştırma oyunudur. Öğrencilerin farklı arama algoritmalarının nasıl davrandıklarını, onların açıklamalı grafiksel tasvirleri üzerinden görmelerine olanak sağlar (Grivokostopoulou 2016).



Resim 4.11 Eğitici Pacman Oyunu

Oyunun iki modu vardır:

1) Eğitici Mod: Öğrenci, dilediği bir metinsel açıklamayı ve bir algoritmanın sahte kodu ile grafiksel akış şemasını okuyabilir. Oyun aynı zamanda öğrencinin algoritmayı farklı Pacman Labirentindeki görselleştirmelerle öğrenmesine fırsat tanır.

2) Oynama Modu: Öğrencinin labirent seviyelerini farklı koşullarla ve bir zaman sınırı içerisinde çözmesi gerekir. Bu seviyeler öğrencinin labirentte Pacman'i hareket ettirmek için belirli bir arama algoritmasını uygulamasını gerektirirecek şekilde tasarlanmıştır. Öğrenciden karakteri belirli algoritmada hareket ettirerek rastgele bir pozisyondan bir kiraza (ödül) veya karakter güçlendiriciye ulaştırması istenir.

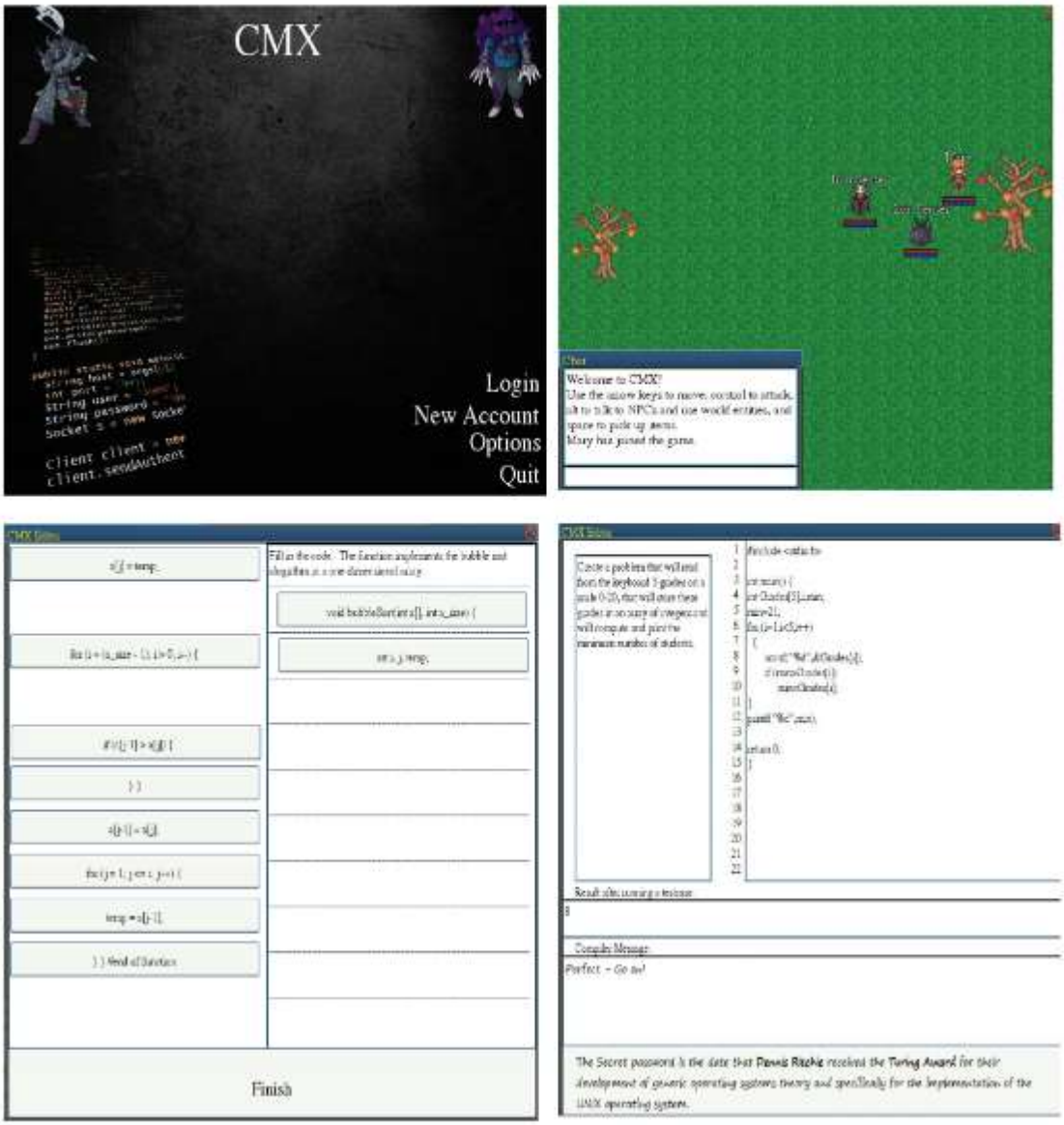
CMX

CMX programlama öğrenmeye yönelik bir devasa çok oyunculu çevrim içi rol yapma oyunudur (MMORPG). Oyunda iki takım vardır: Crackerlar ile hackerlar ve bunlar küresel bir toksik atık

fabrikasında şifreleri bulmak için birbirileriyle yarışırlar. Senseiler onlara C programlama dilini öğrenmede yardımcı olurlar. Oyunda üç seviyede Senseiye önceki seviyedeki şifreyi bularak

ulaşılabilir. Öğrenciler, C dilini anlamada daha yüksek bir seviyeye ulaşmış görünüyorlar çünkü yalnızca teorik sorulara cevap verebilmekle kalmıyor, aynı zamanda çalıştırılabilir programları taşıma

ve bırakma araçlarını kullanarak kurabiliyor ve C dilinde programlar yazabiliyorlardı (Malliarakis,



Resim 4.12 CMX Oyun Ortamı

2014).

Tablo 4.2. Programlamayı öğretmede kullanılan oyun ortamlarının karşılaştırılması

OYUN	PLATFORM	HEDEF KİTLE	DİL(LER)	ÖĞRENME ÜNİTELERİ	OYUN TÜRÜ
Code Combat [17]	Ağ tabanlı	9+	JavaScript, Python	Sözdizimi, metodlar, parametreler, diziler, döngüler ve değişkenler, koşul, Boolean mantığı, ilişkisel operatörler, işlevler, obje özellikleri, eylem düzenleme, girdi düzenleme, aritmetik, sayaçlar, while döngüleri, böl, devam et, diziler, dizi karşılaştırması, min/max bulma, obje değişmezleri, uzak metod, çağrı, for döngüleri, karmaşık işlevler, resim, modulo	Ticari Oyun (*Freemium*)
Human Resource Machine [29, 30]	Windows, Mac OS X, Linux, iOS, Android	9+	Görsel Programlama Dili tabanlı	Girdi & çıktı ve koşullular & yinelemeliler algoritma karşılaştırması	Ticari Oyun (*Ücretli*) Bulmaca Oyunu
Lightbot [29]	iOS, Android, Windows, Mac OS X	9+	Görsel Programlama Dili tabanlı	Koşullular ve yinelemeliler ve özyineleme hata giderme stratejileri	Ticari Oyun *Ücretli* Bulmaca Oyunu
May's Journey [31]	Windows	12-18	Özel Programlama Dili (Java gibi obje odaklı bir dilden ilham alınmıştır)	Temel talimatlar ve sekanslar mantık, döngüler, değişkenler, if söylemleri, karşılaştırıcılar ve Boolean mantığı, tam sayılar üzerinde operasyonlar, diziler üzerinde operasyonlar	Araştırma Oyunu Bulmaca Oyunu
No Bug's Snack Bar [32]	Ağ tabanlı	18+	Görsel Programlama Dili	Değişken manipülasyon, eylem sekansları, koşullular & yinelemeliler, hata giderme stratejileri	Araştırma Oyunu
Robot ON! [33]	Windows	18+	C++	Söz dizimi & anlam bilim, değişken & ilkel veri tipleri, ifadeler & görevler, koşullular &	Ücretsiz/Araş-tırma Oyunu

				yinelemeliler, program anlama	
Educational 'Pacman' Game [16]	Windows	18+	YZ Öğrenmeye Yönelik Oyun Araştırma Algoritması	Algoritmalar (algoritmaların temel metinsel anlatımı, ona karşılık gelen grafiksel akış şeması ve sahte kod)	Araştırma Oyunu
CMX [34]	Windows	18+	C	Diziler üzerine teori, bir dizinin programının çalışması sonrası değişkenlerin çıktıları, dizi programlarının söz dizimi, bir dizideki değişken değerlerin girişi, dizinin toplamının hesaplanması, dizinin azami değerinin hesaplanması	Araştırma Oyunu MMORPG

5. ÖĞRETMENLER VE ÖĞRENCİLER İÇİN OYUN PLATFORMU

5.1. C4G Platformu ve Tasarım Odaklı Düşünme Yaklaşımı

Bir yazılım kullanarak temel ve orta öğretimde başta kızlar olmak üzere gençler arasında programlama becerilerini geliştirmeyi amaçlayan projenin ikinci fikri çıktısı "O2 - Kızların Programlama Becerilerinin Geliştirilmesini Ciddi Oyunlar Yoluyla Destekleme" sonucunda ICT ile ciddi oyunun mükemmel bir birleşimi gösterilmiştir.

Proje çerçevisnde geliştirilen oyun yazılımı, bilgisayar bilimi eğitimine ve kariyerlerine erkek ve kadın katılımı arasındaki mevcut boşluğu bilgisayar bilimini herkes için çekici kılan metodolojik öğrenme müdahalelerini tanıtarak aşmak için tasarlanmıştır. Oyun ve gerçekleştirilecek öğrenme etkinlikleri, kızların bilgisayar bilimine katılımını teşvik etmek amacıyla geliştirilmiştir. Proje ürünleri öğrencilerin akranlarıyla birlikte anlamlı çözümler tasarlamalarına yardımcı olan yaratıcı bir süreç anlamına gelen tasarım odaklı düşünme yaklaşımına göre tasarlanmışlardır.

Bu tasarım süreci zorlukları belirlemek, bilgi toplamak, potansiyel çözümler üretmek, fikirleri iyileştirmek ve çözümleri test etmek için yapılandırılmış çerçevesi sayesinde çok etkilidir.

Süreç, doğası gereği yinelemelidir ve etkileşim gerektirir. Süreçteki her aşama, deneyi, çözüm fizibilitesini ve düşünmeyi teşvik etmek için bir öğrenme deneyimi boyunca yeniden gözden geçirmeye başvurmaktadır (Luka, 2014).

Tasarlanmış düşünme etkinlikleri, sınıf içinde ve dışında işbirliğine dayalı bir deneyim sağlar. Öğrenciler doğrudan bilgi toplama, bilgi üretme, iletişim ve sunumla meşgul olurlar.

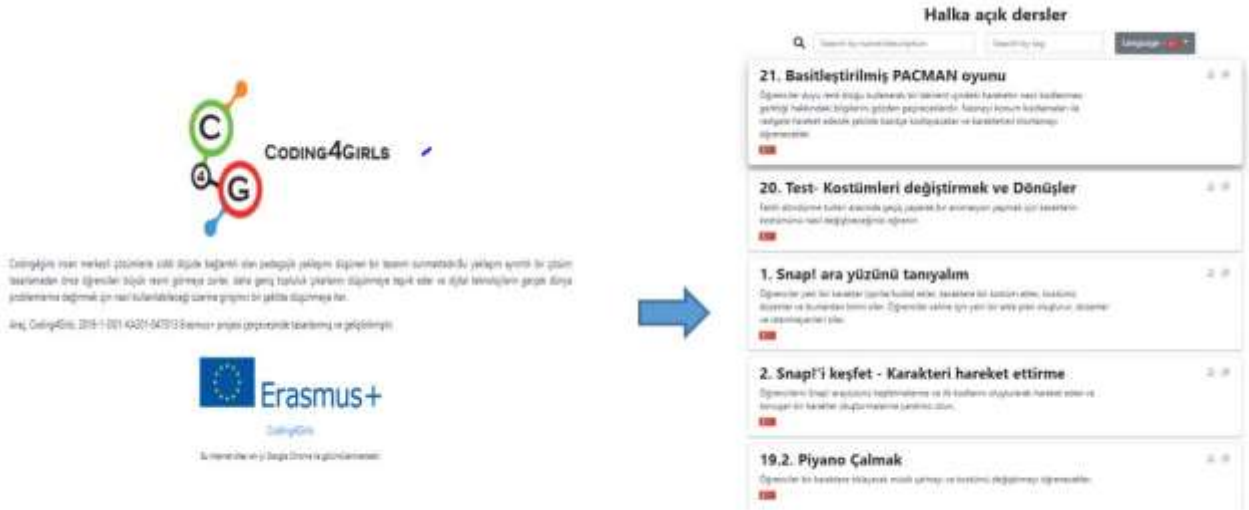
Coding4Girls projesinde kullanılan düşünme yaklaşımı, öğrencileri öğrenme süreçlerinde teşvik etmek ve motive etmek için oyun temelli öğrenmenin temel avantajlarından yararlanmaktadır. Katılım, motivasyon ve elde edilen sonuçları artırarak belirli bir etkinliği daha eğlenceli hale getiren bazı önemli oyun öğelerini (örneğin puanlar ve zorluklar) ve senaryoları kullanır.

Özellikle, öğrencilerin ayrıntılı bir çözüm tasarlamadan önce büyük resmi görmelerine yardımcı olurken karşılaşılan zorlukların çözülmesi; öğrencileri dijital teknolojilerde ve gerçek dünyadaki sorunları ele almak için nasıl kullanılabilecekleri hakkında birer girişimci olarak düşünmeye teşvik eder. Proje kapsamında geliştirilen yazılım, birbiriyle bağlantılı "Öğretmen Platformu" ve "Öğrenci Oyun Ortamı" olarak iki farklı bölümden oluşmaktadır.

5.2 Öğretmen Platformu

Öğretmen Platformu, öğretmenlerin Snap! kullanarak öğrencilerinin kodlama becerilerini geliştirmek için belirli kurslar tasarlayıp hazırlayabilecekleri web tabanlı bir platformdur.

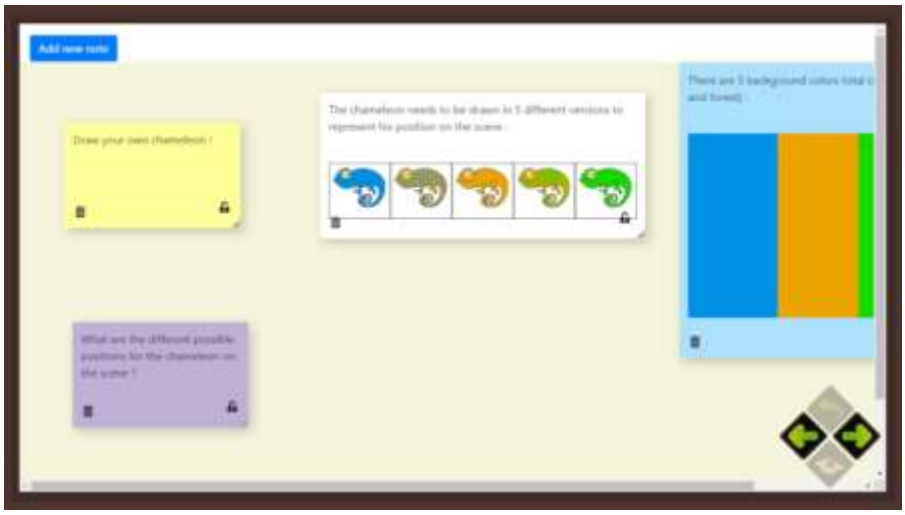
Platformun içinde, her öğretmenin emrinde hem özel hem de genel bir alan vardır. Öğretmenler, özel alanda kurslarını öğrencilerinin ihtiyaçlarına göre hazırlayabilirler. Genel alan ise diğer öğretmenler tarafından oluşturulan tüm derslerin bir deposudur. Amaç, diğer meslektaşlar tarafından hâlihazırda hazırlanmış olan öğrenme aktivitelerini paylaşmak, kullanmak veya bunlardan ilham almaktır (Şekil 5. 1.). Her öğretmen tüm halka açık kursları sınıflarının özelliklerine göre kişiselleştirebilir.



Şekil. 5.1. Öğretmen Platformu ve kodlama kurslarından bazıları.

Bu kurslar, tümü kapsamlı bir konuya bağlı olan, tematik olarak ilgili faaliyetler için bir gruplama alanı olarak oluşturulur. Sorun, kursun başında, işbirliği içinde geçici çözümler geliştirmek için hep birlikte beyin fırtınası yapan öğrencilere sunulur. Bu, öğrencilerin yeni fikir üretebilecekleri ve yaratıcı bir şekilde problem çözme sürecine dahil olabilecekleri öğrenme sürecinin çok önemli bir aşamasıdır.

Beyin fırtınası bölümü, öğretmenler tarafından Öğretmen Platformunda bazı temel sorular veya öğrencilerin tartışmaları sırasında kullanılacak bazı düşünceler ve yansımalar kullanılarak hazırlanabilir. Yazılım, aşağıdaki Şekil 5.2.'deki gibi metin, resim veya video kullanarak sanal bir pano üzerinde post-it hazırlama imkanı sunar:



Şekil. 5.2. Öğrencilerin katkısı ve tartışması için Öğrenci Oyun Ortamı'nda beyin fırtınası alanı.

Tüm post-it'lerin düzenlenebilir veya düzenlenemez olacak şekilde ayarlanabilmeleri için kilitlenebilen veya açılabilen bir asma kilit simgesi vardır. Yalnızca öğretmenler post-it'leri kilitleme ve kilidini açma olanağına sahiptir. Öğrenciler yalnızca üzerinde işlem yapmadan bir gönderinin kilitli olup olmadığını bilebilir.

Beyin fırtınası aşamasından sonra öğrencilere kapsamlı problemleri çözmek için gerekli araçları sunmak amacıyla tasarlanmış özel aktiviteler Snap! kullanılarak sunulur.

Şekil 5.3.'te, Öğretmen Platformu'nda yayınlanan her bir C4G kursunun yapısı gösterilmektedir.

Şekil. 5.3. Kurs Yapısı

Derslerdeki bütün grup aktivitelerine görev denir. Bunlar öğrencilere önceden öğretmenler tarafından belli bir sıraya göre sunulur (Şekil 5.4). Örneğin, bir öğretmen temel programlama bilgileri üzerine bir ders oluşturmak istiyorsa ilk aktivite “boolean” kavramıyla ilgili, ikincisi “koşullu yapılarla” ilgili ve sonuncusu “döngülerle” ilgili olabilir. Bu öğretmenlere, Öğretmen Platformunda öğrencilerine özelleştirilmiş öğrenme adımları hazırlama ve öğrencilere Öğrenci Oyun Ortamında son öğrenme görevine yavaş yavaş yöneltilme fırsatı sunar.

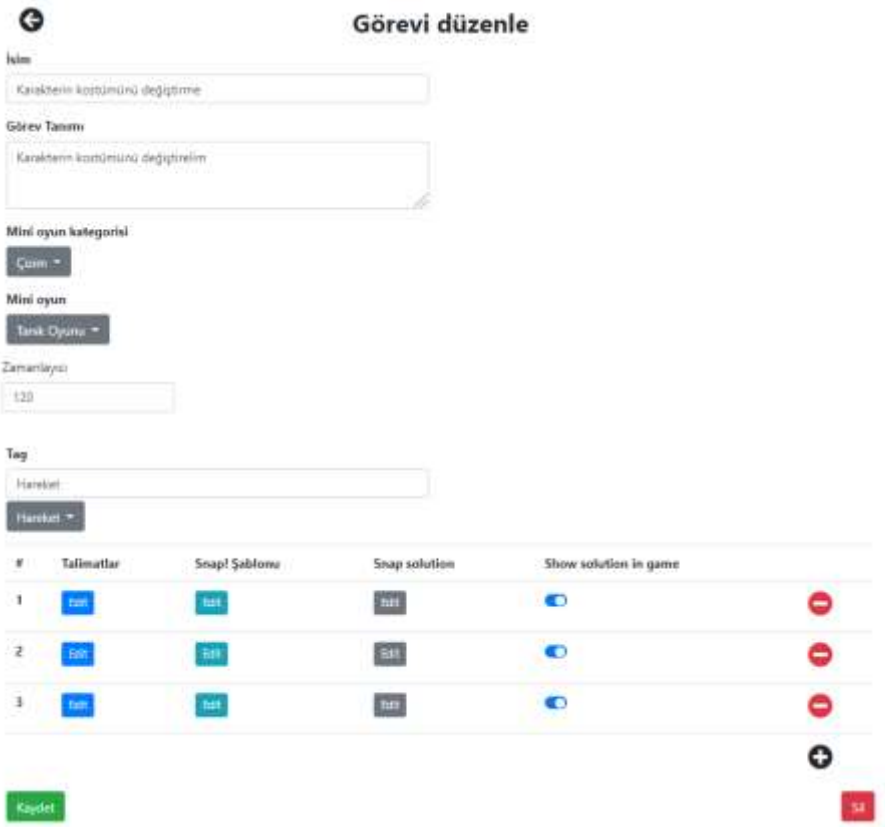
Elbette öğrencilerin son çözüme kavuşmadan önce bütün görevleri öğretmenlerin oluşturduğu sıraya göre oynamaları ve çözmeleri gerekir.

Şekil 5.4 Öğretmen Platformundaki kodlama dersindeki görevların bir örneği.

Her görev proje grubunun geliştirdiği bir mini oyunla bağlantılı olabilir. Amaç öğrencilere araştırılan belli bir kodlama niteliğinin son sonuçlarının ne olabileceğini anlama konusunda yardım etmektir.

Örneğin, bir görev “döngü” niteliği araştırmasıyla ilgiliyse “Match-3” onunla bağlantılı olabilecek bir mini oyundur.

Bu yüzden her görev, görevi çözmek için Snap! platformu hariç, öğrencinin Öğretmen Platformundaki hazırlıklarında öğretmenleri tarafından tanımlanan talimatlarla dolu bir sayfaya uyarak oynaması gereken bir mini oyun barındırabilir (Şekil 5).



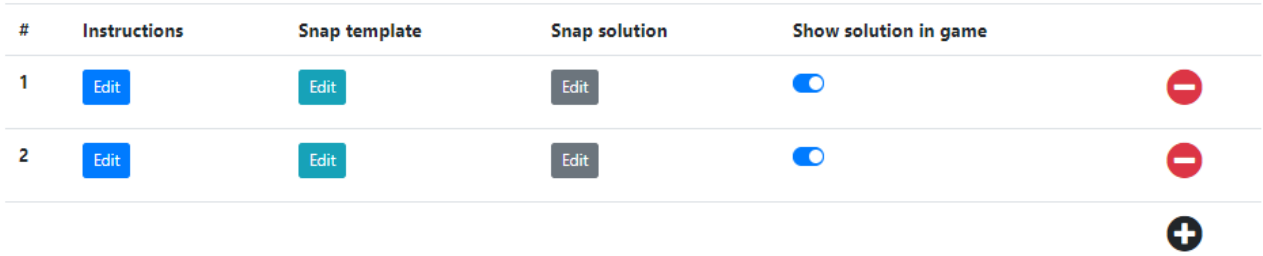
Şekil 5.5: Öğretmen Platformundaki bir görevin görünümü

Öğrencilere sunulan tüm oyunlar (şu anda 11 oyun) C4G dersleri ve senaryoları üzerine tasarlanan esas programlama kavramlarıyla ilgilidir. Ancak göreve mini oyun dahil etmek zorunlu değildir. Bu öğretmenin tercihine bağlıdır.

Her görev üç parçadan oluşur:

1. **Talimatlar:** Burada öğretmenler öğrencilerinin yerine getirmeleri için görev hakkında ipuçları verebilir ve açıklamalarda bulunabilirler.
2. **Snap Taslağı:** Burada öğrenciler Öğrenci Oyun Ortamında açık kaynağın blok kodlamasını kullanarak mesul oldukları görevi yerine getirmeye çalışırlar.
3. **Snap Çözümü:** Burada öğretmenler görevin son çözümünü gösterip göstermeme kararı alabilirler.

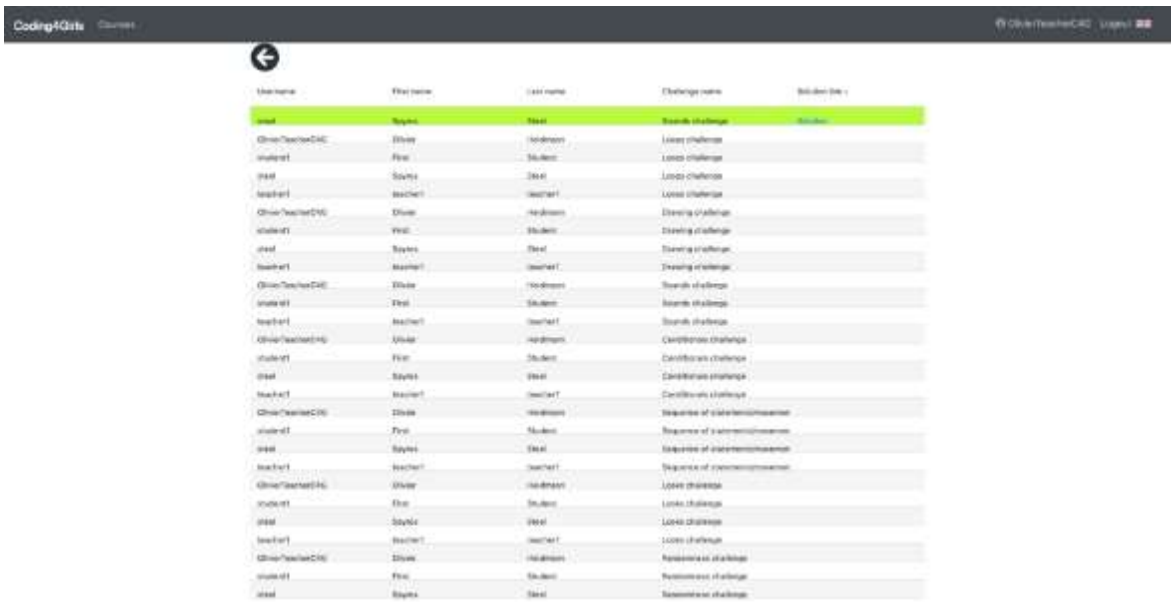
Görev programlama bilgisi bakımından karmaşıksa birden fazla görev şeklinde oluşturulabilir (Şekil 3.7). Bu yolla öğretmenler öğrencilerine karmaşık bir aktiviteyi daha küçük parçalara ayırarak çözmede adım adım ve farklı aşamalarda yol gösterebilirler.



Şekil 5.6: Karmaşık bir görevin daha küçük alt görevlere ayrılması

Öğretmenler öğrencilerinin bütün Snap! cevaplarını Öğretmen Platformundaki tabloda gösterilen her görevde teslim etmesinden sonra toplayabilirler. Tablo, derste var olan tüm görevlerin detaylarını ve kullanıcıların listesini barındırır. Her dizide kullanıcı adı, ad, soyad, görevin adı veya çözülmüş görevler yer alır.

Eğer "Çözüm bağlantısı" sütununda teslim edilen bir çözüm yoksa sütun boş kalır. Aksi takdirde dizi vurgulanır ve "çözüm" sözcüğü en son sütunda gözükür. Ayrıca üstüne tıklandığında kullanıcının teslimi gösterilir (Şekil 3.8.).



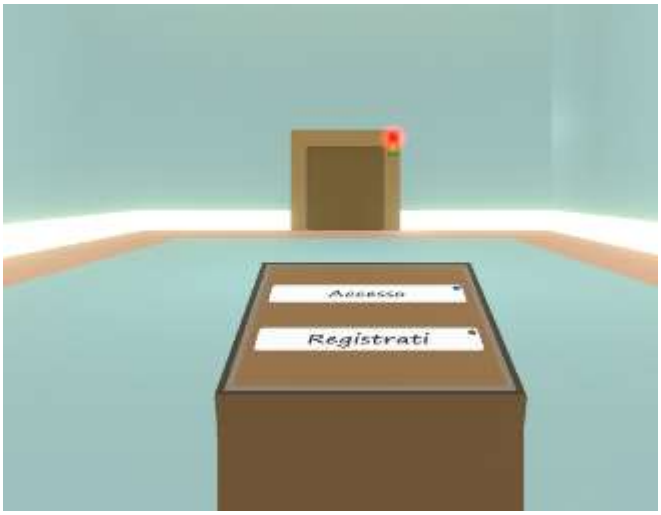
Şekil 5.7: Göreve kullanıcılar tarafından bir çözümle verilen cevaplar

Böylece, öğretmenler öğrencilerinin ilerlemelerini ve sonuçlarını her zaman gözleyebilir ve takip edebilirler.

5.3. Öğrenci Oyun Ortamı

Öğrenciler, bu görevlere Unity 3D oyun olarak indirilebilir yazılım olan Öğrenci Oyun Ortamı üzerinden erişebilirler. Burada öğrenciler öğretmenlerinin hazırladıkları dersleri eğlenceli, sürükleyici ve interaktif bir şekilde keşfedebilir ve tamamlayabilirler.

Dersler, tasarımsal düşünme yaklaşımının öğelerini kullanarak öğrencilere kapsayıcı bir sorun sunarak onu adım adım çözmek için kullanılan araçları çözmelerine ve sunmalarına olanak tanımaktadır. Bu dersler, yukarıda da belirtildiği gibi öğretmenler tarafından hazırlanır ve Öğretmen Platformunda yayınlanırlar. Öğrenciler ise derslere Öğrenci Oyun Ortamından erişebilirler.



Şekil 5.8: Öğrenci Oyun Ortamına erişen ilk oda

Öğrenciler bir derse portalın terminaline doğru kodu girerek katılabilirler. Kod özeldir ve Öğretmen Platformundaki dersi halihazırda hazırlayan öğretmenler tarafından sunulmalıdır. Ders teslim edildikten sonra kullanıcının ismi portalın üzerinde görünür hale gelir ve aynı zamanda seçili dersi de gösterir (Şekil 5.8).

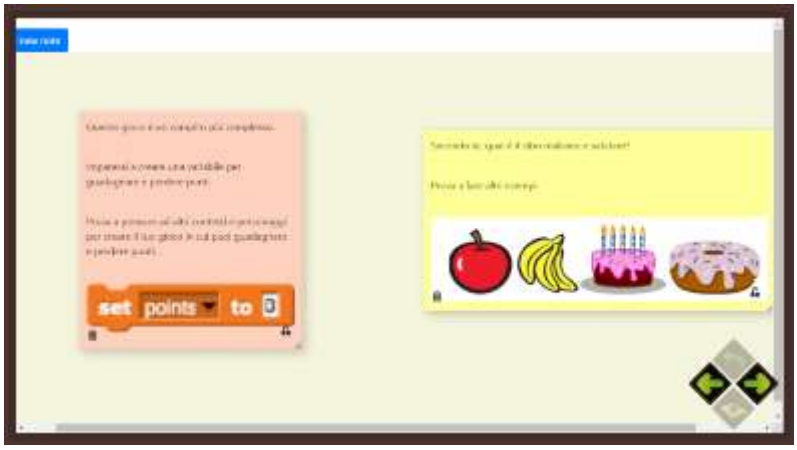
Şekil 5.9: İki terminalli ikinci oda. Biri ilk kez derse katılmak için, diğeri de halihazırda kaydolunmuş dersi seçmek için.

Öğrenciler (öğretmenlerin kararına göre) belirli ihtiyaçları ve sorunları dile getiren oyunları kodlamaya ve tasarlamaya teşvik edilirler. Bu bir "alçak giriş yüksek tavan yaklaşımı"dır. Öğrencilere kolay sorunlardan başlayıp daha zor görevlere doğru ilerlemelerine olanak verir. Öğretmenler öğrencilerine, onları görevi tamamlamaya davet ederek "Yarı-Geliştirilmiş" senaryolar tasarlayıp sunarlar ve bunlarda bir çözüm kısmen hazırdır. Bu sebeple öğretmenler öğrencilerinin ihtiyaçlarına ve bilgilerine uygun C4G Proje yazılımlarını kullanarak küçük ve baş edilebilir modüller hazırlayabilirler.

Tasarımsal düşünme yaklaşımının izinden giderek proje grubu, öğrencileri belirli bir kodlama sorunu çözmede sınamak için tasarlanmış birkaç öğrenme senaryosu ve ders hazırlamışlardır. Şu anda 21 öğrenme senaryosu,"öğretmenler için toplanmış oyun tasarımı tabanlı öğrenme sayfaları" raporunda toplu halde ve C4G yazılımının yapısına uygun tasarımsal düşünme yaklaşımına tekrardan uyarlanmıştır.

Senaryoların basitten daha becerikli öğrenciler için ileri düzeye farklı seviyeleri mevcut olup İngilizce, Slovence, İtalyanca, Hırvatça, Bulgarca ve Yunanca dillerinde hazırlanmıştır.

Görevler bireysel ve sınıf içerisinde grup tartışmaları yapılarak veya sanal olarak Öğrenci Oyun Ortamında işbirliği içinde çözülebilir. Yazılım, öğrencilerin Şekil 3.11'de gösterildiği gibi multimedya notlarını yerleştirerek ve ayarlayarak tartışıp fikirlerini paylaşabilecekleri bir beyin fırtınası ortamı sunar. Derse dahil olan her öğrenci katkılarını yazabilir ve bunlar o derse kaydolan herkes tarafından görülebilir.



Şekil 5.10: Öğrenci Oyun Ortamında *"Bir piknik için yemek almak"* öğrenme senaryosundaki beyin fırtınası alanı

Aşağıdaki şekil öğretmenin Öğretmen Platformunda oluşturduğu dersin Öğrenci Oyun Ortamındaki görevlerinin panosunu göstermektedir. 0'dan 13'e kadar olan sayılar görevleri temsil eder. Bu panoda 0'ıncı görev oyunun başında verilen genel talimatları temsil eder. 0 görevini açmak öğrencileri beyin fırtınası platformundan önce gelen ilgili Snap! Taslağı olan global sorun talimatlarına yönlendirir. 1'den 13'e kadar olan diğer sayılar ise öğretmenler tarafından hazırlanan görevleri temsil eder.

Şekil 5.11: Öğrenci Oyun Ortamı'ndaki görevlar panosu

İçlerinden birini seçince görev detayları, görevin ismi solda, görev ile bağlantılı mini oyun ortada olacak şekilde gözükür. Görevi başlatmak için de ortada bir düğme gözükür. Şekil 5.11 işlenilecek

konuya göre görevinin ismini gösterir. Şu an bu “Koşullu görev”dir. Görev konusunun yanı sıra, sistem onunla bağlantılı 3D mini oyunu (örn. “Yolunu bul”) gösterir.

Öğretmenler öğrencilerin isteğe bağlı hangi mini oyunları (3’ünü denkleştir oyunu, Zar oyunu, Envanter oyunu, Yolunu bul oyunu, Ses oyunu, Yılan oyunu, Bulmaca oyunu, Desen eşleştirme oyunu ve şıklı sorulu bir kısa sınav gibi) oynayacaklarına mini oyunlardan bir tanesini seçerek karar verebilirler.

Şekil 5.12: Var olan bir mini oyunun örneği – Zar Oyunu

Öğrenci Oyun Ortamındaki her görev şu şekilde hazırlanmıştır: Kodlama kavramını gösteren bir mini giriş oyunu, görevin Snap'te yerine getirilmesi için eğitici bir sayfa, iki Snap! platformu (biri görevin çözümü için kullanılmak amacıyla, diğeri de aktivitenin çözümünü göstermek için).

Eğitici sayfa, Snap!’te yerine getirilmesi için öğrenme görevinin hedefini ve bağlamını sunmak amacıyla HTML'dir (Şekil 14). Genellikle görseller ve videolarla zenginleştirilmiştir.

Şekil 5.13: Öğrenci Oyun Ortamındaki görevin eğitici sayfasının bir örneği

Bu sayfanın ardından öğrencilerin çözmesi veya yerine getirmesi için bir sorunu veya kodlama aktivitesini içeren ve öğretmen tarafından sağlanan bir taslağa dayalı bir Snap! platformu (Şekil 5.15) gelir.

Şekil 5.14: Öğrenci Oyun Ortamında kodlama görevini çözmesi için öğretmen tarafından sağlanan Snap! taslağının bir örneği.

Sunulan görevin çözümünü göstermek için başka bir Snap! platformu daha sağlanacaktır. Ancak çözüm plaatformunun gösterilip gösterilmeyeceğine öğretmen karar verebilir. Bu, eğitmen tarafından seçilen öğretme süreci yönetimine bağlıdır.

Şekil 5.15. Öğrenci Oyun Ortamında, öğretmenlerin öğrencileri için sağladığı görevlerin çözümüyle birlikte sunduğu Snap! platformunun bir örneği.

Her ders daha fazla görev barındırdığı için, bu adımlar öğretmenlerin belirlediği toplam görev sayısı kadar tekrar edilir. Bu, karmaşık bir kodlama aktivitesinin basit ilk düzey adımlara bölünmesine olanak tanır. Bu adımlarda, öncesinde gelen cevap ve/veya çözüm, sonraki aktivitenin gerçekleştirileceği taslak olur.

Sonunda ders, artımlı bir öğretim çatısı süreci üzerinden öğretilerek programlamayı temsil eden görevlerin belirli bir sayısından oluşmuş olur.

Öğrenciler dersteki tüm görevleri (Şekil 7) tamamlandıktan sonra tekrar ilk kodlama sorununa yönlendirilirler. Edindikleri yeni bilgileri sentezlemeleri istenir. Dersin en sonunda oyuncular, yüzleşme anları canlandırarak soruna ait diğer öğrencilerin sunduğu tüm çözümlerini görebilirler.

Bu C4G yaklaşımı ve araçları, eğlenceyi ve öğrenme görevlerini birleştiren kişiselleştirilmiş eğitim aktiviteleriyle öğrencilere kodlama becerilerini geliştirme, yükseltme ve pekiştirme fırsatı sunar.

6. DERS ÖRNEKLERİ (ÖĞRENME SAYFALARI)

6.1. Öğrenme Senaryoları Sayfaları

6.1.1. Öğrenme sayfalarının genel tanımı

C4G Projesi kapsamında ciddi oyunu ve tasarımsal düşünme yaklaşımlarını uygulayan uçtan uca öğrenme aktivitelerini dahil eden 22 öğrenme senaryosu sayfası geliştirildi. Öğrenme sayfaları İngilizce dilinde ve proje ortaklarının ulusal dillerinde (Türkçe, Bulgarca, Hırvatça, Yunanca, İtalyanca, Portekizce ve Slovence) sunulmakta ve projenin web sayfasında (https://www.coding4girls.eu/results_03.php) yer almaktadır.

Hazırlanan öğrenme sayfaları, özlü bir biçimde eğitmenlere sunulan ciddi oyunları ve tasarımsal düşünmeyi öğrenme metodolojilerini öğretimlerinde uygulamak konusunda yardımcı olan bilgiler verir. C4G Projesi aktif, oyun tabanlı öğrenme tasarımının izinden gider ve her öğrenme aktivitesinin kızlarla erkeklerin programlama becerilerini inşa etmeye yönelik geliştirilmesi için bilgileri içerir.

Öğrenme sayfaları aşağıdaki bilgileri içerir:

- Söz konusu öğrenme aktivitesinin genel eğitimsel hedefi
- Öğrenme aktivitesi tarafından kapsanan kavramlar
- Spesifik öğrenme hedefleri
- Beklenen öğrenme çıktıları
- C4G'ün oyun tasarımı tabanlı öğrenme yaklaşımının adım adım kullanımı
- Geliştirilen bilgileri ölçmeye yönelik değerlendirme metodları
- Sınıf işbirliği bağlamında öğrenciler arasında tartışma başlatacak sorular

Öğretmenler senaryoları ve oyunları önerilen sırada veya tercih ve ihtiyaçlarına göre serbestçe seçebilirler. Aynı zamanda öğretmenlerin, öğrencilerin matematikte öğrendikleri bazı kavramlar hakkındaki bilgilerine göre bazı senaryoların uyarlanmasını halletmeleri (örn. koordinat sistemleri, koordinatlar, negatif sayılar, kesirler vs.) gerekir.

Öğrenme sayfaları, kullanıcı etkileşimi süreçleri ile geri dönüş verme dahil, hem sunulan ciddi oyunun genel işlevselliğini, hem de sunulan ciddi oyunda uygulanacak olan tüm öğrenme aktivitelerinin tanımlarını kapsar.

Hazırlanan öğrenme sayfaları temel bir programlama kavramından daha ileri düzey programlama kavramlarını işlerler. 22 öğrenme sayfasının tamamı Ek 1'de mevcuttur. Ek 2'de C4G Proje ortamındaki oyunların halka açık bölümünde sunulan tüm oyunların kodları verilmiştir.

6.2. C4G Projesi Çerçevesinde Geliştirilen Oyun Ortamının Kullanımı ile Örnekleri

Öğrenci Oyun Ortamı, Windows ve Mac OS X'te mevcut olan Unity tabanlı bir ciddi oyundur. Kullanıcı oyunu oynarken ilk önce Proje ve Erasmus+ logolarını 4 saniye boyunca göstererek proje feragatnamesini sunan bir giriş sahnesiyle karşılaşır.

Oyuncu ondan sonra lobi olarak adlandırılan büyük boş bir odaya girer ve görünmez bir avataı yönlendirerek oyunun dünyasını birinci şahıs bakış açısından görür. Odanın ortasına bir konsol yerleştirilmiştir. Bu, kullanıcının C4G Proje sunucusuna kimlik bilgilerini girerek veya yeni bir hesap açarak giriş yapabilmesini sağlar. Giriş yaptıktan sonra bir kapı açılır ve oyuncu başka bir odaya ilerleyebilir, yeni bir derse kaydolabilir ya da halihazırda aktif olan bir derse katılabilir. İstenilen ders seçildikten sonra kullanıcı oyunda ilerlemek ve seçili dersin kesin içeriğini keşfetmek için havada duran bir portaldan geçer.

Bir ders, Snap! Platformu kullanılarak ve her biri bir mini oyunla örneklenen bağlantılı kodlama görevlerinin bir derslemesinden oluşur. İdeal olarak her görev, derste öğrenilecek konunun bir yönünü veya adımını örnekler.

Varsayılı olarak, C4G Projesi her biri bir mini oyun tarafından örneklenen birkaç temel kodlama kavramını tanımlamıştır. döngüler, koşullular, değişkenler, söylemler, paralellik, operatörler ve eylemler. Bunun yanı sıra öğretmenler dilerlerse mini oyunun yerine seçenekli sorulardan oluşan bir quiz sınavı yapabilir veya hiçbir şey yapmayabilirler. Böylece sadece görevlerdeki Snap! egzersizleri kalmış olur.

Şekil 6.1. Öğretmen Platformunda mevcut olan mini oyunların kategorisi

Her görev Snap! platformunda çözülecek kodlama görevlerini içerir ve görevin panosunda gösterilen mini oyunla ilişkilendirilebilir (Şekil 6.2).

Şekil 6.2. Öğrenci Oyun Ortamındaki görev panosu

Mini oyunun belirli bir görevle bütünleştirilmesi, öğretmenler tarafından tasarlanan görevin arkasındaki programlama kavramını örneklendirmek için olup zorunlu değildir. Bu, öğretmenin öğrencileri için göreve mini oyun dahil edip etmeyeceği ve var olan oyunların listesinden hangi mini oyunun oynanacağına karar verebileceği anlamına gelir.

Görevi bir mini oyunla bağdaştırma, eğer öğrenciye anlama, kavram kanıtı, mekaniklerin görselleştirilmesi ve tabi ki öğrenciyi eğitim süreçlerinde motive edip sürükleme konularında artı bir değer katacaksa göz önünde bulundurulmalıdır.

Şekil 6.3: Öğretmen Platformunda mevcut olan mini oyunlar

Kullanıcı bir mini oyuna bağlı göreve giriş yaptığında mini oyunun yer aldığı lokasyona ışınlanır. Farklı oyunlar için kumlu plajlardan karlı dağlara kadar uzanan farklı birçok yer vardır.

Şekil 6.4: Bir mini oyunla bağdaştırılan bir yerin örneği

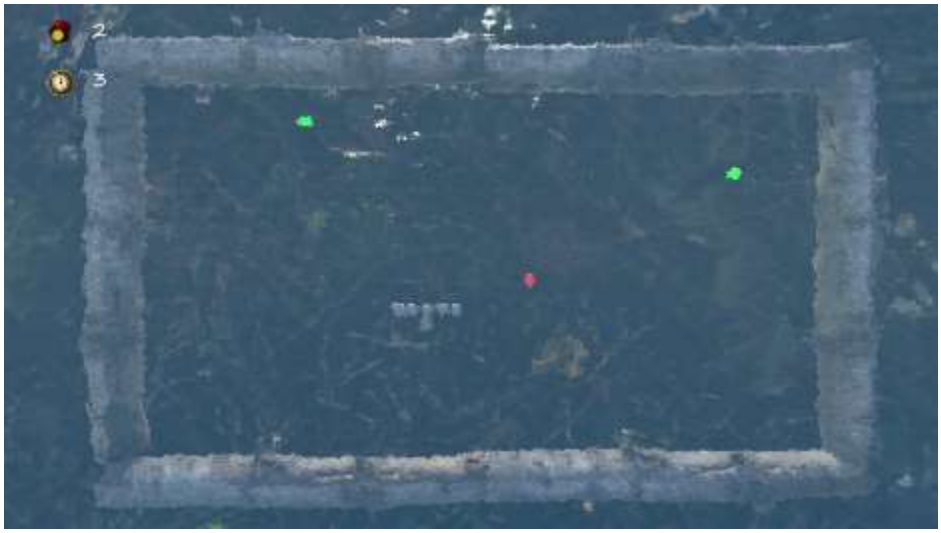
Mevcut mini oyunlar arasında yılan oyunu, 3'ünü denkleştir oyunu, zar oyunu, envanter oyunu, yolunu bul oyunu, adım atma oyunu, ses oyunu, bulmaca oyunu, desen eşleştirme oyunu ve bir çoktan seçmeli sınav vardır.

Her mini oyunun bir zorluğu vardır. Mevcut mini oyunların çeşidi, öğrencilerin olası farklı yaş gruplarını ve/veya derslerde öğretilen programlama konularını kapsar.

En basit mini oyunlardan biri yılan oyunudur. Bir nehir kenarında yer alır ve oyuncunun aktiviteyi başlatmak için ahşap bir ok işaretini kırmızı bir noktaya kadar takip etmesi gerekir (Şekil 6.5).

Şekil 6.5: Yılan oyunuyla bağdaştırılan yerin örneği

Yılan oyunu (Şekil 6.6) suda yer alır. 4 klavye ok tuşuyla yılana yön verilir ve mümkün olduğunca hayatta kalmaya çalışılır. Yılan parlak yeşil noktaları yiyerek uzunluğunu artırır. Ancak kendi kuyruğuna çarptığında, ekranın kenarına çarptığında veya kırmızı bir nokta yediğinde ölür. Yenilen yer yeşil nokta bir puan olarak yansır.



Şekil 6.6: Yılan oyunu

Bu basit mini oyun döngüler, hareketler, grafik değişimi vs. gibi birkaç programlama kavramını güzel bir şekilde örneklendirir.

Diğer bir Örnek “3’ünü denkleştir” oyunudur. Oyuncuların serbestçe dolaşabileceği karlı, yüksek bir bölgede yer alır.

Şekil 6.7: “3’ünü denkleştir” oyunuyla bağdaştırılan yer

"3'ünü denkleştir" mini oyunu tipik bir üçünü denkleştirme aktivitesine dayanır. Yan yana olan blokları yok etmek için aynı türden üç veya daha fazla bir grup oluşturacak şekilde taşıyıp bırakmanız gerekir.

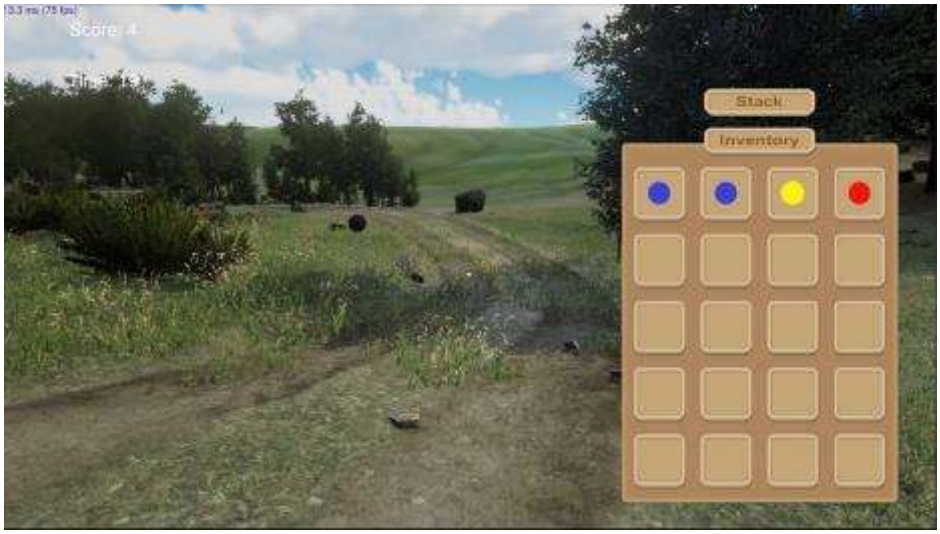
Şekil 6.8: "3'ünü denkleştir" mini oyunu

Zar oyunu karanlık bir ormanda yer alır. Mini oyunun interaktif kısmı terk edilmiş bir kulübede, kapının karşısındaki sandalyenin üstünde iki zar olan bir tahtadır.

Hem oyuncu hem de yapay zeka sırayla zar atarlar. İki zarın toplamı en büyük olan raundu kazanır.

Şekil 6.9: Zar oyunu

Envanter oyunu küçük bir ormanın yanındaki bir alanda yer alır. Alanın etrafında uçuşan renkli küreler vardır. Oyuncunun onları tahta levhada öğretmen tarafından verilen talimatlara göre toplaması gerekir.



Şekil 6.10: Envanter oyunu

Basma mini oyunu, bir adadaki plajda günbatımı döneminde yer alır. Oyuncunun, plajın merkezinde olan büyük kayanın üstüne yazılı soruları cevaplaması gerekmektedir. Doğru cevabı kodlamak için harflerin üstüne tek tek basarak cevaplaması gerekir.

Oyuncu, cevabı oluşturan bir harfe bastığı zaman harf yeşil yanar. Cevap sorunun yazılı olduğu kayanın yanındaki ikinci büyük kayada gözükür. Yanlış bir harfe bastığında ise harf kırmızı yanar ve kodlamaya tekrar başlanılması gerekir.

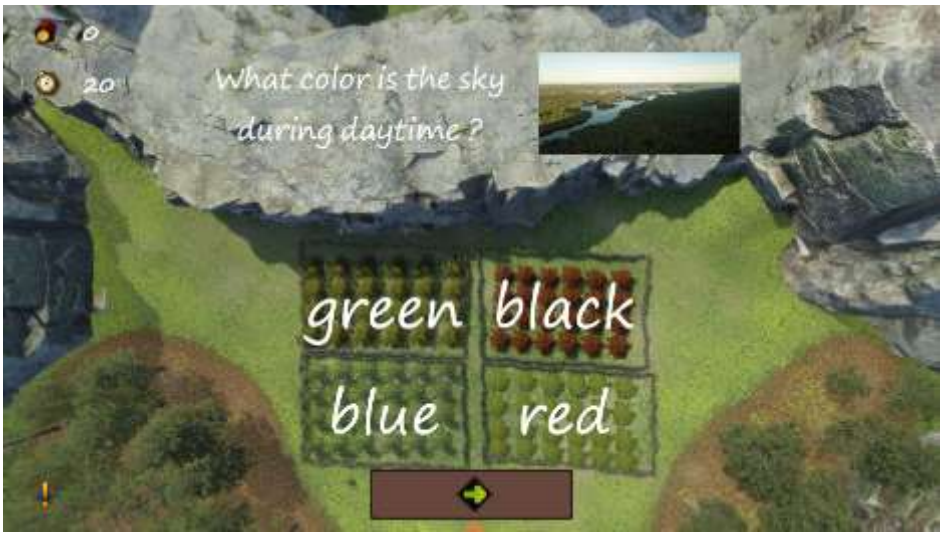
Şekil 6.11: Adım atma oyunu

Seçenekli sorulu kısa sınav, uçurumun kenarındaki bir alanda yer alır. Oyuncular başlangıçta mini oyunu başlatmak için dolaşmada serbesttirler.

Sorular uçurumun tepesinde gözükür. Her tarla doğru olabilecek dört sorudan birini içerir. Varsa onlara görseller eşlik eder.

Madalya ikonu öğrencinin anlık puanını temsil eder. Puan, doğru cevaplarla görevin bitimine kalan sürenin saniyelerine dayalıdır.

Alttaki ok tuşu sıradaki soruya geçmenizi sağlar.



Şekil 6.12: Seçenekli kısa sınav

Bulmaca oyunu, haritanın kayalık bir yerinde yer alır. Amaç alanda etrafa dağıtılmış panelleri bulmak ve panelin beyaz dairesinden başlayıp kırmızı karesine doğru giden bir çizgi oluşturarak bulmacaları çözmektir.

Şekil 6.13: Bulmaca oyununda bir panel

Desen eşleştirme oyunu bir nehrin yatağında yer alır. Kenarlarında sayılar yazan bazı kutular nehirden denize doğru yüzmeye başlarlar. Oyunun amacı, oyuncunun onları matematiksel işlemler gerçekleştirmek için toplamasıdır.

Başka bir deyişle, oyuncuların yüzen sayıları ve mevcut işlemleri mavi tabloda yazan sayıya ulaşmak için kullanmaları gerekir.

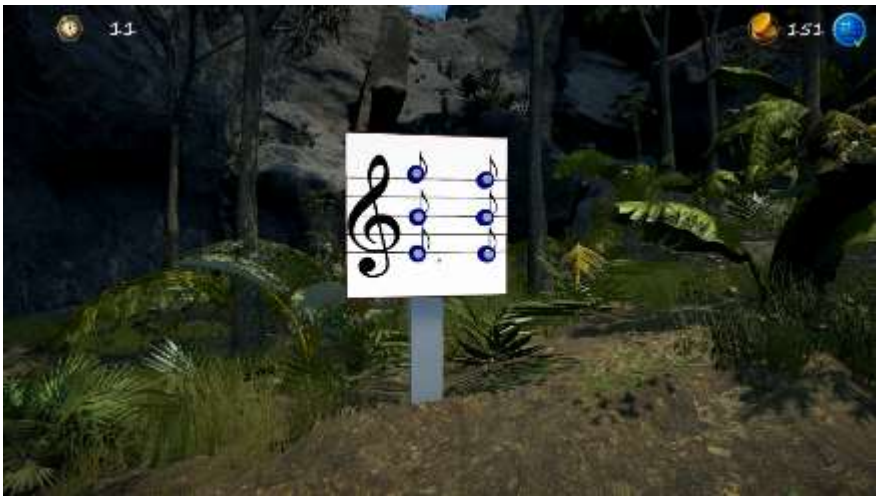
Hedef sayılar ve yüzen sayılar rastgele, fakat her zaman hedef sayıya ulaşabilmeyi sağlayan bir kombinasyon olacak şekilde oluşturulur.

Mevcut işlemler hocaya bağlıdır ve şunlardan biri olabilir:

- Basit işlemler (+, -, *, /)
- İleri düzey işlemler (üslü sayılar, karekök, modulo)
- Trigonometri (Cos, Sin, Tan)
- Booleans (VE, VEYA, YA DA).

Şekil 6.14: Desen eşleştirme oyunu

Başka bir mini oyun örneği de oyuncuların görevleri yerine getirmesi için yoğun dikkat gerektiren ses oyunudur. Tropik bir ormanda yer alır. Oyuncunun etrafa saçılı beş paneli bulması ve ormanda duyabildiği seslere göre bulmacaları çözmesi gerekir.

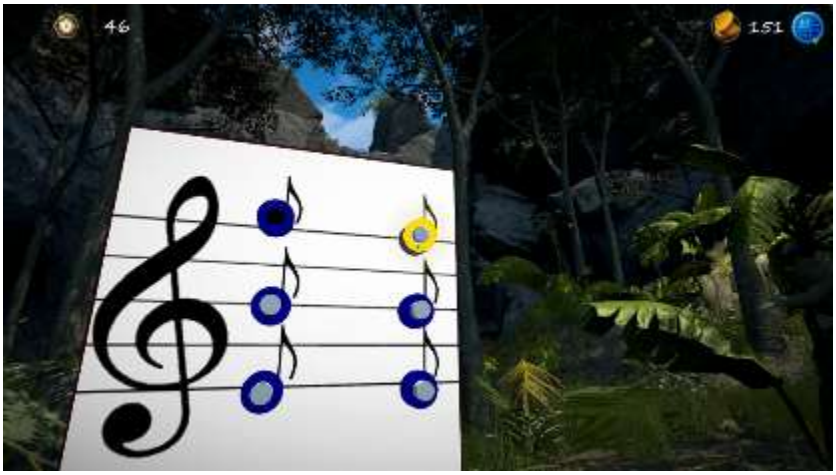


Şekil 6.15: Ses oyunundaki panellerden biri

Panelin her sütunu bir sesi ve her sıra sesin tonunu temsil eder. Üstteki tuşlar yüksek tonlar için ve alttakiler düşük tonlar içindir. Kullanıcı bir panele yakınsa 3 olası sesin bir kombinasyonu çalar (bir derin kuş ötüşü, bir orta sesli kuş ötüşü ve bir tiz kuş ötüşü). Kuş ötüşlerinin kombinasyonu bir döngüde tekrarlanır ve her tekrar diğerlerinden 3 saniye ile ayrılır.

Oyuncunun kuş ötüşlerinin çalma sırasını dikkatle dinlemesi ve panelde onu doğru şekilde tekrar etmesi gerekir.

Örneğin, ilk panlede (Şekil 6.16) kullanıcı ilk önce derin bir ötme, sonra tiz bir ötme duyar. Dolayısıyla doğru cevap panelin sol alttaki tuşuna ve sağ üstteki tuşuna basmaktır.



Şekil 6.16: Ses oyunundaki panelde gösterilen öğrenci cevapları

Belli bir tuşu seçmek için kullanıcının sadece fare imlecini tutması yeterlidir. Tuş sarı renkle vurgulanır. Basılan tuş doğruysa yeşil, değilse siyah görünür.

Ses genelde her oyunda önemlidir. Fakat bu oyunda azami derecede önemli olup oynanış döngüsünün merkezindedir. Bunun sebebi oyuncuların tiz, orta ve derin sesleri tanımalarının ve onlara odaklanmalarının gerekmesidir.

Öğrenci Oyun Ortamı genelde 10-15 yaşlarındaki her öğrenci tarafından kullanılabilir. Ortam, sınıf içindeki ve dışındaki kodlama aktivitelerine eşitlikçi katılımı teşvik etmeyi amaçlar. Ayrıca, programlama kavramlarını, öğrencilerin keşfedebileceği üç boyutlu sanal bir ortamı, kavramları gösteren mini oyunları, olası bir çözüm için sınıf arkadaşları arasında fikir alışverişi sağlayan işbirlikçi bir ortamı, bir sorunu halletmeye yönelik bakış açıları geliştirmek için diğerlerinin çözümlerinin incelemelerini ve öğretmenin sunduğu incelemelerin çözümlerini ve karşılaştırmalarını kullanarak gösterir.

Programlama kavramlarını daha da örneklendirmeye yarayacak çok daha fazla mini oyunla C4G Proje yazılımını zenginleştirecek gelişmeler ve geliştirmeler öngörülebilir. C4G ciddi oyununun modüler yapısı ve Unity'de tasarlanma biçimi bu tür gelecek eklemeleri mümkün kılmaktadır.

EK- 1 ÖĞRENME SAYFALARI

TEMEL ÖĞRENME SENARYOLARI		GELİŞTİREN ORTAK
1	**Snap! ile tanışın – Ara yüz** Snap!'i yakından tanıyın - Görsel programlama ortamı	UL
2	**Karakterinizi canlandırmanın vakti geldi** Programlama bloklarını bulmak, bağlamak, hareket ettirmek, karakter oluşturmak ve konuşturmak	UL
3	**Sahnede hareket etmek** Blokları anlamlı bir sıraya sokmak	UL
4	**Kostüm değiştirme ve döndürmeler**	UL
5	**Çiftliğin sesi** Ses ekleme, içeri aktarmak, kayıt etmek ve ses çıkarmak	UL
6	**Bukalemunun yaz tatili, basit versiyon** Olaylara aşina olma, renk algılama, Boolean değerleri, iki farklı oyun durumunu kontrol etme ve bunlara yanıt verme	UL
7	**Prense ve prensese evcil hayvanlarını bulmada yardım etme** Koşulları kullanma ve çizme	UL
8	**Tebeşir ile çizim yapmak** Döngü kullanmak, çevirmek, arka planı değiştirmek	UL
9	**Çöpleri toplamak ve parkı temizlemek** Değişkenlere aşina olmak, karakterleri çoğaltmak, kod blokları	UL
10	**Kedileri beslemek** Değişkenler (döngünün içinde/dışında), döngüler, rastgele sayılar, dizi birleştirme, 4 işlem, girdi kullanma	UL
11	**Bir barınaktaki kedi sayısını tahmin etmek** Rastgele değerler, değişken girişi, koşul ifadeleri, karşılaştırma işlemleri, sayaç kullanma	UL
İLERİ SEVİYE ÖĞRENME SENARYOLARI		**GELİŞTİREN ORTAK**
12	**Sağlıklı yiyecekler yakalamak** Değişkenler, koşullar, döngü, nokta yönü, rastgele kullanma	UL
13	**Hikâye anlatma**	S WU
14	**Çizim yapma**	U NIRI
15	**Fareyi yakalamak** Döngüler, koşullu ifadeler, değişkenler kullanma	UL
16	**Piknik için yiyecek satın almak** Değişkenler, koşul ifadeleri, 4 işlemi kullanma	UL
17	**Operasyonlar**	S WU
18	**Geri dönüşüm**	S

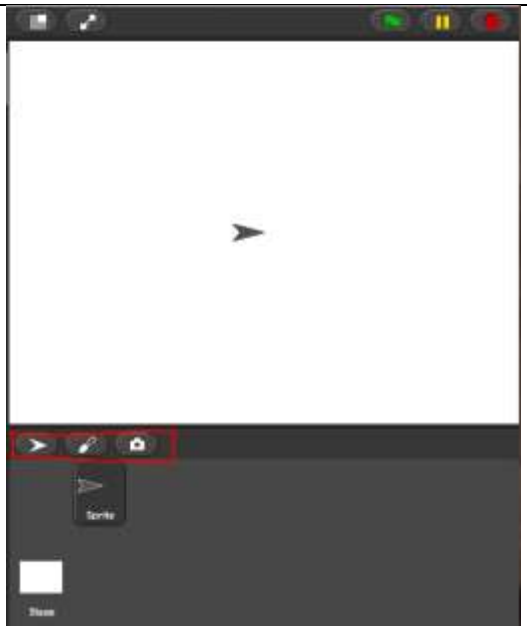
		WU
19.1	**Piyano çalma 1**	S WU
19.2	**Piyano çalma 2**	U NIRI
20	**Test**	S WU
21	**Basitleştirilmiş PACMAN oyunu** Olay tabanlı nesne hareketi, renk algılama, Boole değerlerini kullanma, iki farklı oyun durumunu kontrol etme ve bunlara yanıt verme	UL

TEMEL ÖĞRENME SENARYOLARI

Öğrenme Senaryosu 1 - Snap! ile tanışın - Arayüz

Öğrenme Senaryosu Adı	**Snap! İle tanışın – Ara yüz**
Geçmiş Programlama Deneyimi	/
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Snap! İle tanışın. Görsel öğrenme ortamı. Özel Öğrenme Çıktıları: ● Öğrenci yeni karakter (Sprite – Kukla) oluşturabilir ● Öğrenci karakterine kostüm ekleyip değişiklikler yapabilir. ● Öğrenci karakteri merkeze koyarak düzgün bir şekilde döndürebilir. ● Öğrenci sahneye yeni bir arka plan ekleyebilir ve düzenleyebilir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	Öğrenci yeni bir karakter (Sprite – Kukla) ekler. Karaktere bir kostüm ekler. Kostümü düzenler ve bunlardan birini siler. Öğrenci sahne için yeni bir arka plan oluşturur, düzenler ve istenmeyenleri siler. ***Amaç:*** Bir saatin sonunda öğrenciler, bir oyunda kullanmak için en sevdikleri karakteri ve gerçek ya da hayali yaşam ortamını çizeceklerdir. Etkinliği tüm öğrenciler için daha motive edici kılmak amacıyla, karakter çizimlerinin bu hedef kitleye uygun olduğu bilimsel çalışmalarda tespit edilmiştir.
Faaliyetin Süresi	45 dk.
Öğrenme ve Öğretme Strateji ve	Öğretmen Gösterimi Bireysel Çalışma

Metotları	
Öğretme Formları	Ön Çalışma Bireysel Çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Bir saatin sonunda öğrenciler, bir oyunda kullanmak için en sevdikleri karakteri ve gerçek ya da hayali yaşam ortamını çizecekler. [Adım 1] Öğrencilere Snap!'i bulabilecekleri web sayfasını gösterin! (https://snap.berkeley.edu/). Onlara ara yüzün farklı bölümlerini gösterin. Bloklu bölüm, senaryoları bir araya getirebilecekleri / kostümleri değiştirebilecekleri / sesleri ekleyebilecekleri bölüm, üzerinde karakter (Sprite – Kukla) bulunan sahne, karakter listesi.  [Adım 2] Aşağıdaki resimde işaretlenen düğmelerden birini tıklayarak yeni bir karakter oluşturabilirsiniz.



Yeni bir karakter çizmeyi deneyeceksiniz. Boya fırçasına tıkladığınızda, karakterinizi Microsoft Paint Uygulamasındakine benzer şekilde çizebileceğiniz bir açılır pencere açılır.

Öğrenciler için görev: İlk karakterinizi çiziniz. 10 dakikanız var.

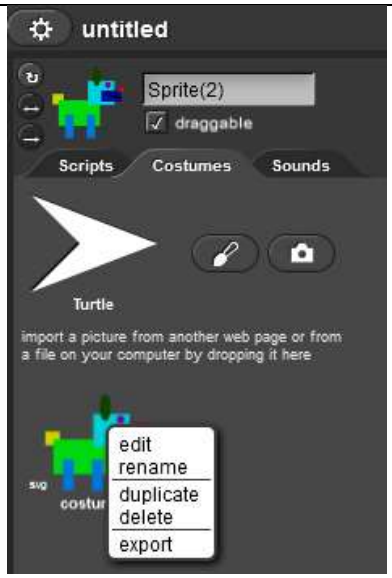
Hareketli grafik çizildikten sonra, hareketli grafiğin dönüş merkezinin olmasını istediğiniz yerde olduğundan emin olmalısınız.

Bunun için  simgesini kullanabilirsiniz.

Öğrenciler için görev: Karakterinizi merkeze koyun.

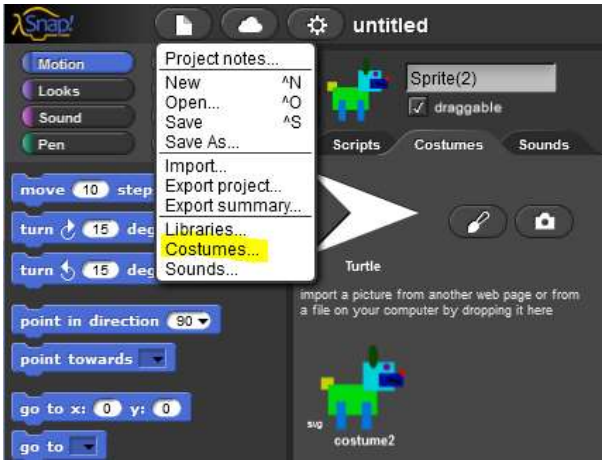
[Adım 3]

Karakterinizi düzenlemek için yalnızca hareketli grafiğiniz tıklandığında görünen *Kostümler* sekmesini seçin. Düzenlemek istediğiniz bir kostüme sağ tıklayarak düzenlemeyi seçin. Ayrıca kostümünüzü aynı menüden kopyalayabilir veya silebilirsiniz.



[Adım 4]

Halihazırda var olan bir kostümü içe aktarmak için, üzerine bir parça kağıt çizilmiş simgeye tıklayın ve *Kostümler'i* seçin….



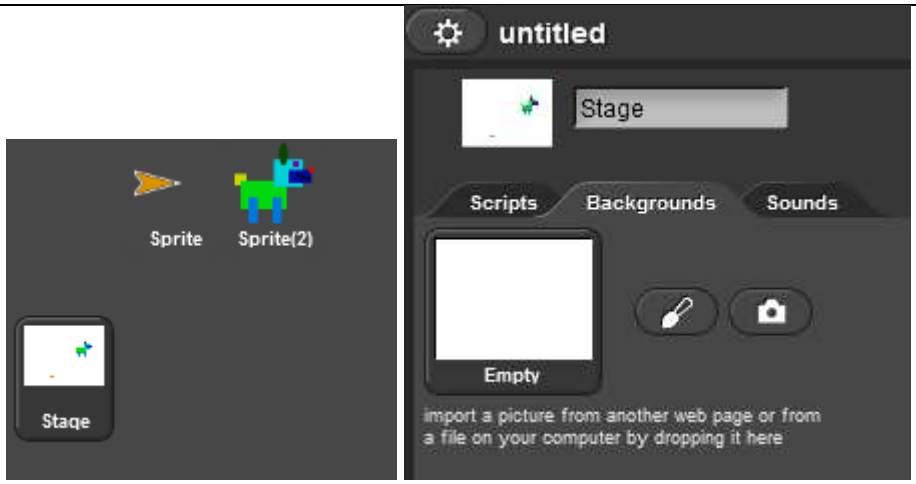
Yine, bu seçenek yalnızca karakteriniz sahnede tıklandığında gösterilecektir.

Öğrenciler için görev: Bir kostüm seçerek karaktere ekleyin.

[Adım 5]

Artık karakteriniz var ve sahneye biraz arka plan eklemelisiniz. Bunu yapmak için önce sahnedeki karakterin yerine *Sahne Alanı (stage*)'na tıklayın.

Arka plan eklemek için yeni arka plan tabını tıklayın.

	 ***Öğrenciler için görev:*** Kendi arka planınızı oluşturun. ***Öğrenciler için görev:*** Mevcut arka planlar arasında arama yapın ve bunlardan birini içe aktarmaya ekleyin. Böylece iki tane elde edersiniz. Öğrenciler için görev: Arka planınızı düzenlemenin bir yolunu bulun. Arka planlarınızdan birini silmenin bir yolunu bulun. Böylece geriye yalnızca biri kalır. ***Düşünme ve değerlendirme:*** Öğrenciler karakterlerini ve çevrelerini yaşadığı yerde çizmeyi başardılar mı? Herhangi bir sorunları oldu mu? Onları nasıl çözdüler?
Öğretmenler İçin Araçlar ve Kaynaklar	https://snap.berkeley.edu/
Öğrenciler İçin Kaynaklar ve Materyaller	Öğrenci için talimatlar (C4G1_InstructionsFor Student.docx)

Öğrenme Senaryosu 2 - Karakterinizi hayata döndürme zamanı

Öğrenme Senaryo Başlığı	Karakterinizi hayata döndürme zamanı

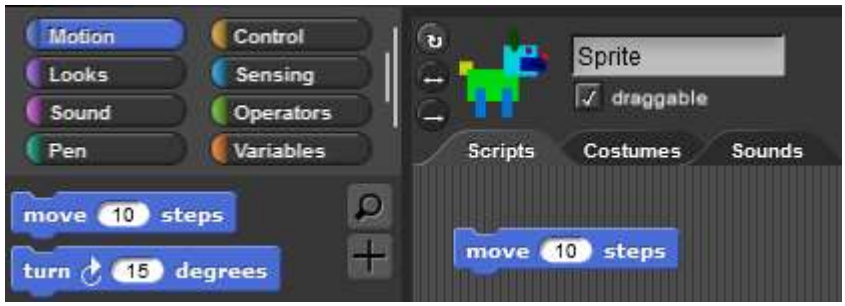
Önceki programlama deneyimi	/
Öğrenme Çıktıları	Genel öğrenme sonuçları: • Öğrenciler programlama bloklarını nerede bulacağını ve bunları bir sıraya nasıl bağlayacaklarını bilir. • Öğrenciler bir karakterin (Sprite – Kukla) nasıl taşınacağını bilir. • Öğrenciler karakterine bir şey söyletmeyi bilir. Algoritmik düşünme odaklı özel öğrenme çıktıları: • Anlamlı bir blok dizisi oluşturma
Amaç, Görev ve Etkinliklerin Kısa Tanımı	Öğrenciler programlama bloklarının nerede depolandığını ve uygun olanları nasıl bulacağını, hangi blok kategorilerinde olduğunu ve blokların bir sıraya nasıl bağlanacağını öğrenir.
Faaliyet Süresi	45 Dakika
Öğrenme ve Öğretme Stratejisi ve Yöntemleri	Öğretmen sunumu Bireysel çalışma
Öğretim Formları	Ön çalışma Bireysel çalışma
Öğretim özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Bu derste karakterinizi hareket ettirecek ve bir şeyler söyleteceksiniz. Onlara bu derste programlayacakları bir programın örneğini gösterebilirsiniz. [Adım 1] Önce kullanabileceğiniz programlama bloklarının nerede olduğuna bakalım. Neredeler? Sol tarafta, blokların farklı kategorilerini bulabilirsiniz. Hareket, Görünüm, Sesler, Kalem, Kontrol, Algılama, İşlemler ve Değişkenler. İlk kullanacağız blok. 

Öğrenciler için görev: Önce bloğu bulun ve sonra üzerine çift tıklayın. Ne oldu?

[Adım 2]

Bloğu bir programa bağlamaya başlamak için " 

" bloğunu kod alanı kısmına taşıyın.



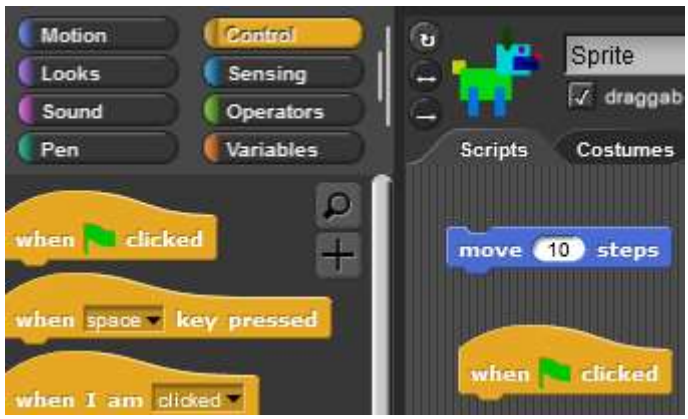
Kodu çalıştırmak için *kod alanı/scripts* sekmesinin içindeki bloğu çift tıklatabilirsiniz.

[Adım 3]

Snap'teki programlar genellikle yeşil bayrağa tıklayarak başlar.

Öğrenciler için görev: Farklı kategori türlerine tıklayın ve yeşil bayrak tıklanırsa programı başlatan bir blok bulmaya çalışın.

Çözüm:



Programın doğru adım sırasına göre çalışmasını istiyorsanız blokların bulmacalarda olduğu gibi bağlanması gerekir. Bunun gibi;

Şimdi yeşil bayrak tıkladığınızda, karakteriniz resim üzerinde farklı bir konumdan 10 adım hareket edecek.

[Adım 4]

Bir bloğun üzerinde beyaz boşluk varsa bu orada yazılan sayıları veya harfleri değiştirebileceğiniz anlamına gelir.

Öğrenciler için görev: Karakterinizi bir seferde 10 yerine 30 adım boyunca hareket ettirin.

[Adım 5]

Karakterine bir şey söylet. İlgili bloğu nereden bulacaksın? 

ve 

arasındaki fark nedir?

[Adım 6]

Her iki *söyleme (say)* komutunu *görünümler (Look)* kategorisinde bulabilirsiniz. Temel fark, 

programı kod devam etmeden önce, __ saniye beklemesini söylememesi veya herhangi bir zamanda söylemeyi bırakmasıdır.

[Adım 7]

Bir önceki derste kullandığınız karakterinizi (Sprite – Kukla) alın. Sahnede sürükleyerek sahnenin sol tarafına taşıyın ve bir program yazmak için bu bloğu 

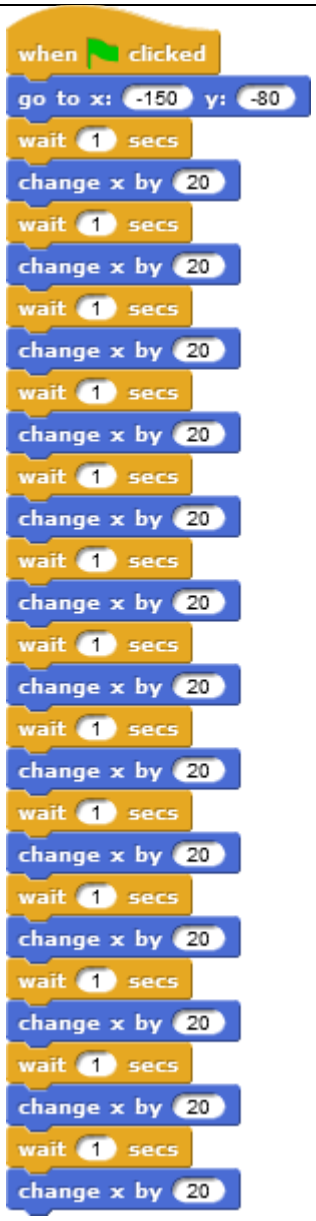
kulanın. Resimdeki blok sahnenin sağ tarafından sol tarafına hareket yapar. Unutmayın! Her hareketten sonra karakter bir şey söylemeli ve tek hamleden fazlasını yapmalı.

	Deneyin: Programınız her çalıştırıldığında karakter tam olarak aynı pozisyonda mı durdu? Karakterinizin her zaman aynı konumdan başlayıp sahnede durmasını sağlayacak bir blok bulabilir misiniz? Öğretmen İçin İpucu: Eğer karakter sahneden kayboluyorsa, sağ fare tuşu ile üzerine tıklayarak *göster* seçip sahneye geri çağırabilirsiniz. Bu blok  hangi x ve y konumunda olduğunu belirlemek için kullanılır. Karakterinizi sahne üzerinde olmasını istediğiniz noktaya taşımanızı ve karakterin konumunu x ile y konumuna tıklayarak (blokların *Hareket* kategorisinin alt kısmında) bu blok sayesinde gerçekleştirebilirsiniz. Düşünme ve Değerlendirme: Görevi tamamlamak için karakterinizin kaç kez hareketi tekrarlayıp diziyi tekrarlaması gerekti? Sayı sınıftaki herkes için aynı mı? Neden?
Öğretmen için Araçlar ve Kaynaklar	**Örnek program:** https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dog_goes_home
Öğrenciler için kaynaklar/materyaller	- Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx) - Öğrenci kendi karakterini ve arka planını oluşturmak isterse aşağıdaki linki kullanabilir: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dog_goes_home_tmp

Öğrenme Senaryosu 3 - Sahnede hareket etmek

Öğrenme Senaryosu Adı	**Sahnede hareket etmek**
Geçmiş Programlama Deneyimi	● Öğrenci, programlama bloklarını nerede bulacağını ve bunları bir diziye nasıl bağlayacağını bilir.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Anlamlı bir blok dizisi oluşturma Algoritmik düşünmeye yönelik Özgül Öğrenme Çıktıları: ● Öğrenci, karakteri sahnede konumlandırır. ● Öğrenci, karakterin x ve y konumunu değiştirir. ● Öğrenci x döngüsünü tekrar kullanır. ● Öğrenci, karakteri hareket halindeki hareketinin yönünün ve adımlarının karakterin döndüğü yöne bağlı olduğunu öğrenir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Öğrenci sahnede karakteri x ve y yönünde hareket ettirmeyi öğrenir. Verilen görevleri çözmek için kolay bir program programlar. Karakteri farklı bir yöne nasıl çevireceğini ve bunun *move_steps (__adım hareket et)* bloğunu nasıl etkilediğin öğrenir. **Görevler:** Hareketli grafiği x yönünde hareket ettiren bir program oluşturun. Hareketli grafiği y yönünde hareket ettiren bir program oluşturun. X ve y yönlerinde hareketi birleştiren bir program oluşturun. **Amaçlar:** sahnede x ve y yönündeki hareketleri ayırt etmek ve tekrar döngüsünü kullanmak.
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Öğretmen gösterimi (demonstrasyon) Bireysel çalışma
Öğretme Formları	Ön çalışma Bireysel Çalışma

Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Farklı hayvanların hedeflerine ulaşmalarına yardım edeceksiniz. Bunu yapmak için onlara sahnede nasıl hareket edecekleri konusunda talimat vermeniz gerekecek. [Görev 1] "Topu yakalayın"ı açın ve köpeğin topu yakalaması için kod ekleyin.  ve  blokları kullanarak köpeğin topa doğru koştuğu bir animasyon yapın. Görevin muhtemel çözümü:



Gördüğünüz gibi, sola veya sağa hareket ettiğinizde x değişir. X = 0 ise, karakteriniz sahnenin ortasındadır. Ortanın sağında, x değerleri 0'dan büyük sayılardır.

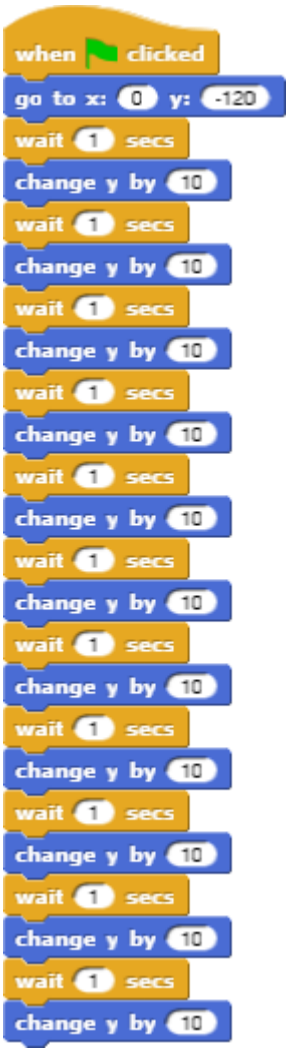
İpucu: Bu aktivite ondalık sayılar konusunu bilen daha büyük yaştaki öğrencilerle yapılırsa bekleme süresi daha kısa olabilir. Örneğin 0.1. koordinat sisteminin ne olduğunu bilirlerse bazı açıklamaların yapılmasına gerek olmayabilir.

[Görev 2]

Maymunun ağaca tırmanmasına yardım et ve muzları getirmesi için maymuna kod ekle. 

ve 

blokları kullanarak maymunun palm ağacına tırmanmasını sağla. Bu görevin muhtemel çözümü:



Gördüğünüz gibi yukarı veya aşağı hareket ettiğinizde y değişiyor. Y 0 ise karakteriniz sahnenin ortasındadır. Ortadan yüksek olanların tümünde y değerinden büyüktür. Karakterinizin sahnede orta çizginin altında olmasını istiyorsanız sanki dalışa gidiyorsunuz gibi düşünebilirsin. Sayının önüne – (eksi işareti) koyun ve suyun kaç metre altında olduğunuzu yani sahnede çizgiden kaç adım geride olduğunuz belirtin.

Eğer ağaçtan inmek istiyorsanız 

bloğunu kullanabilirsiniz.

İpucu: Bu aktivite ondalık sayılar konusunu bilen daha büyük yaştaki öğrencilerle yapılırsa bekleme süresi daha kısa olabilir. örneğin 0.1. koordinat sisteminin ne olduğunu bilirlerse bazı açıklamaların yapılmasına gerek olmayabilir.

[Adım 3]

Her iki görevde de birbirinin yerine iki blok kullanmak zorundaydınız. Kodu kaç defa tekrarlamak zorunda kaldınız?

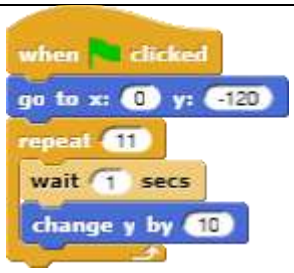
Bilgisayara kodunuzu belirli sayıda tekrar etmesini söyleyerek bu kodu yazmayabilirsiniz. Bu *repeat (tekrar)* döngüdür. Aynı eylem veya bir eylem dizisi kendini birden fazla kez tekrarladığında bunu kullanabilirsiniz.

İki görev içinde kodunuzu değiştirin. Yani 

(tekrar) döngüsünü kullanın. Tekrarlamak istediğiniz kod bu bloğun içine konulmalı ve boş alana kaç defa tekrarlanması gerektiğini yazmalısınız.

Köpek için kod

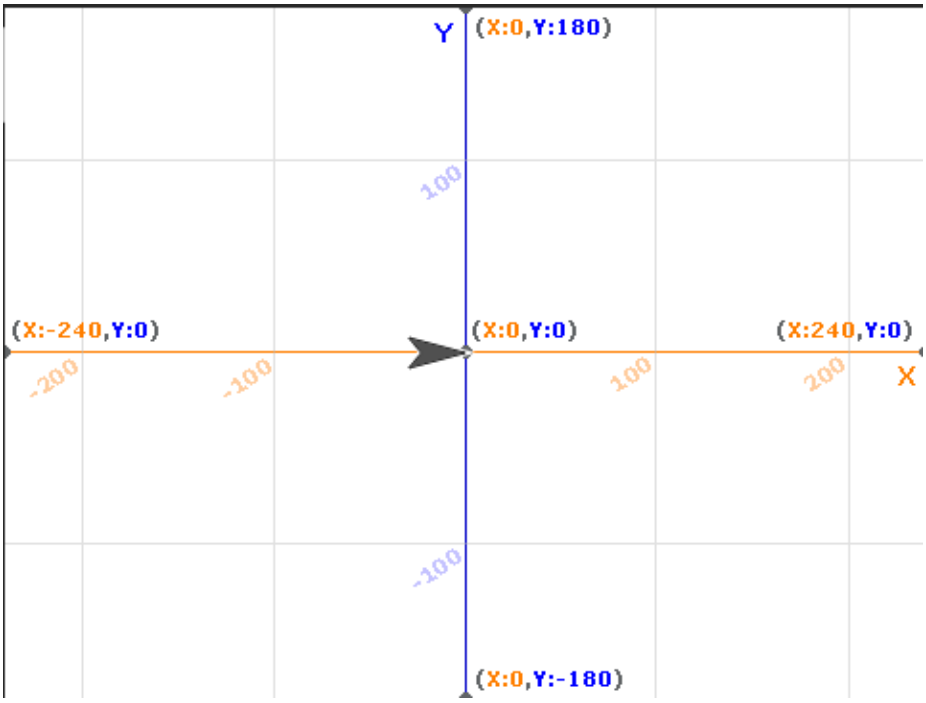
Maymun için kod



Görev: Köpeğin topa doğru koşmasını ve geri dönmesini sağlamaya çalışın.

Görev: Maymunun ağaca tırmanmasını ve aşağı inmesini sağlamaya çalışın.

En çok neyi sevdin? Pozisyonlama için X-Y grid arka planını kullanarak hareketli grafiğin x ve y konumu konusunda yardım alabilirsiniz.



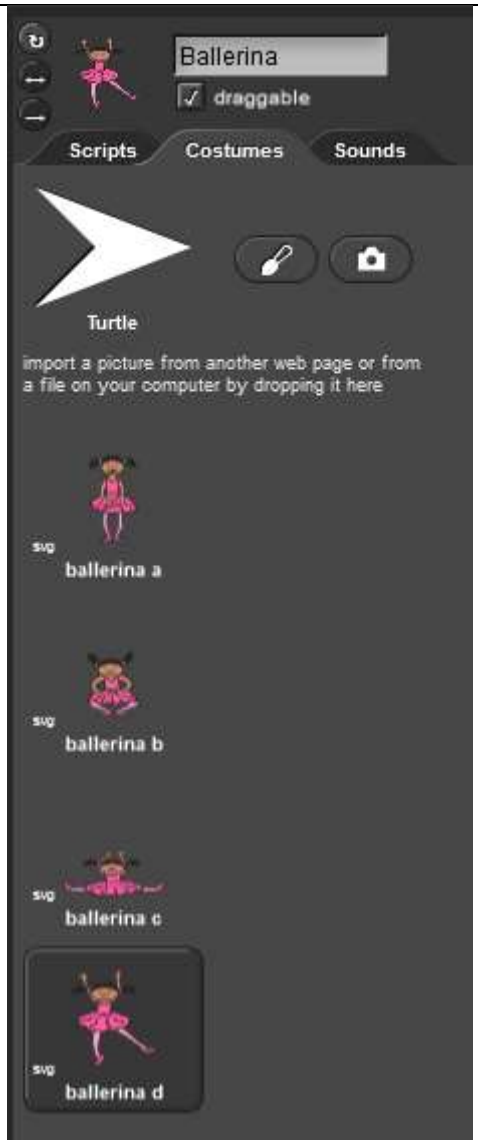
Öğretmenler İçin Araçlar ve Kaynaklar	• Topu yakalamanın çözümü https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_moving_x • Maymunu ağaca tırmandırmanın çözümü

	https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_moving_y
Öğrenciler İçin Kaynaklar ve Materyaller	• Topu yakala https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Catch_the_ball • Maymunu ağaca tırmandır. https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Help_monkey_climb_the_tree • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx

Öğrenme Senaryosu 4 - Kostüm değiştirme ve döndürmeler

Öğrenme Senaryosu Adı	Kostüm değiştirme ve döndürmeler
Geçmiş Programlama Deneyimi	Hareket
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: • Anlamlı bir blok dizisi oluşturma Algoritmik düşünceye odaklı özel öğrenme çıktıları: • Öğrenci, animasyon yapmak için karakterin kostümünü değiştirir. • Öğrenciler karakterlerin dönüşünü değiştirir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Öğrenci, bir animasyon yapmak için karakterin kostümünü nasıl değiştireceğini öğrenir. Ayrıca, karakterin farklı dönüş türleri arasında nasıl değişiklik yapılacağını da öğrenir. **Görevler:** Karakterin kostümünü değiştiren bir program oluşturun. Her programda karakter için uygun dönüş türünü ayarlayın

	Amaçlar: Karakterin kostümünü değiştirmeyi ve uygun dönüş türünün nasıl ayarlanacağını öğrenin.
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Demonstration Bireysel çalışma
Öğretme Formları	Ön çalışma Bireysel çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Yürüyor, dans ediyormuş gibi görünmesi için bir karakter animasyonunu nasıl yapacağınızı öğreneceksiniz. [Adım 1] Yeni bir boş proje açın. Beyaz bir kâğıt parçasına benzeyen simgeye tıklayın ve *kostümler…(costumes)* öğesini seçin. Önce *Balerin a*'ya ardından *İçe Aktar*'a tıklayın. *Balerin b, balerin c ve balerin d* için de aynısını yapın. Karakterinizin *kostümler* sekmesinde artık 4 balerin kostümünüz var. *Kostümler* sekmesinin üzerindeki metni değiştirerek karakteri *Balerin* olarak yeniden adlandırabilirsiniz: Kostüm Sekmesi:

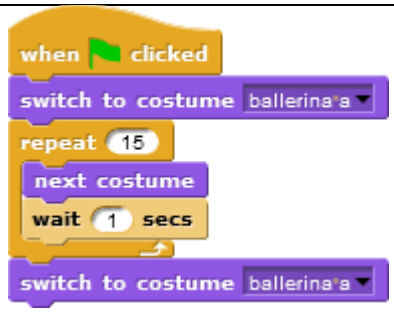


Şimdi komut dosyası (Scripts) sekmesine geri dönün. Yeşil bayrak tıklandığında başlayan ve her saniyede 15 kez değişen *balerin* görünümünü değiştirecek bir kod oluşturmaya çalışın.

next costume Bloğunu kullanmanız gerekecek. Balerinin iki ayağının da dansın başında ve sonunda yerde olmasına özen gösterin.

Başlangıç ve bitiş pozisyonları dansın bir parçası değil.

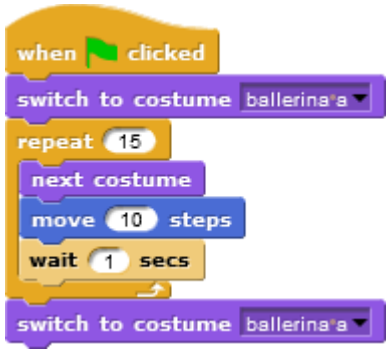
Çözüm:



[Adım 2]

Balerinimiz her zaman aynı pozisyonda olmak istemediği için her kostümü değiştirdiğinde küçük hareketler yapıyor. Bu hareketi onun dansına ekleyin.

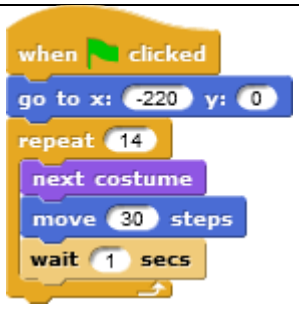
Olası çözüm:



[Adım 3]

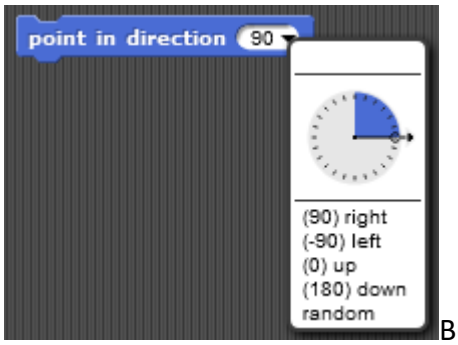
Yeni bir boş proje açın ve *Avery* (Snap!'te özel bir karaker/kukla/sprite) yürüyüş kostümlerini içe aktarın. *Avery*'nin yürümesi için uygun bir arka plan ekleyin. *Avery*'nin sahnenin sol tarafından sahnenin sağ tarafına yürüdüğü bir animasyon oluşturun. *Avery*'nin bir şekilde nasıl canlandırılacağını, *Adım*'larının gerçek hayatta olduğu gibi bağlantılı görüneceğini anlamaya çalışın

Olası çözüm:



[Adım 4]

Şimdiye kadar bir karakterin yalnızca bir yönde hareket ettiği bir program yazdınız. Bu görevde peynire ulaşmak için fareyi döndürmeniz gerekecek. Karakteri döndürmek için aşağıdakilerden birini seçebilirsiniz.

a) 

Buradan hangi yöne doğru dönmesi gerektiğini söyleyebilirsiniz.

b) Belli bir açıyla saat yönüne 

ve saat yönünün tersine 

dönmesini söyleyebilirsiniz.

Tam bir daire 360 dereceye sahiptir. Bu nedenle şu an bulunduğunuz yerden ters yönde dönmek isterseniz 180 derece dönersiniz. Sola dönmek isterseniz saat yönünün tersine 90 derece dönersiniz. Sağınıza dönmek isterseniz saat yönünde 90 derece dönüyorsunuz.

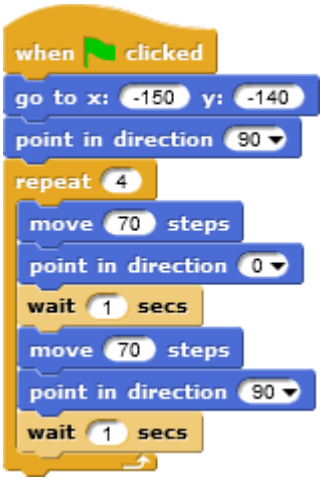
Bakınız:

https://snap.berkeley.edu/snap/snap.html#present:Username=spe

lac&ProjectName=C4G_Find_cheese.

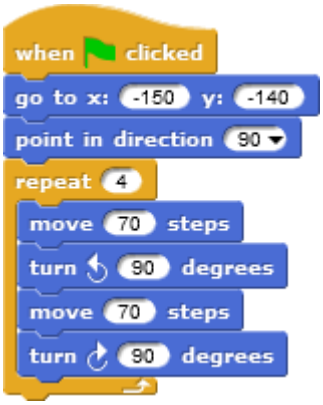
Farenin sadece yeşil alanda yürümesi gerekiyorsa peynire ulaşmak için takip etmesi gereken bir program yazın. Fareyi gideceği yöne çevirin ve __ Adım bloğunu hareket ettirin. Farenin nasıl hareket ettiğini görmek için satırlar arasında 1 saniye bekleyin.

Çözüm:



Şimdi 90 derece çevirerek bir program yazmaya çalışın.

Çözüm:



[Adım5]

Gördüğünüz gibi fare peynire ulaşmak için farklı yönlere dönmüştür. Bazen karakterinizin baş aşağı dönmesini istemezsiniz. Ya da sadece sola veya sağa dönerek başının üstünde yürümesini istemezsiniz. Karakterinizin istediğiniz gibi döndüğünden emin olmak için solundaki uygun simgeye tıklamanız gerekir.

	 Dairesel ok karakterinizi istediğiniz her yöne (farede olduğu gibi) döndürmenize yarar. <-> şeklindeki ok ise karakterinizin yalnızca sağa ya da yalnızca sola dönmesini sağlar (Köpeğin kafasının üzerinde yürümemesi için kullanacağın şey). Son ok (->) hareketli grafiğin her zaman olduğu gibi görüneceği anlamına gelir (bunu maymun için kullanabilirsiniz). Köpek ve maymun için programlarınızı yeniden yazmaya çalışın. Bu sefer önce nesneye gidip dönerek geri dönsünler. Döndürme stilini doğru şekilde değiştirdiğinizden emin olun.
Öğretmenler İçin Araçlar ve Kaynaklar	• Balerin programlama çözümü: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dancing • Avery: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Avery_walking • Peyniri bulmanın çözümü: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Find_cheese_solution
Öğrenciler İçin Kaynaklar ve Materyaller	• Peyniri bul https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Find_cheese • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 5 - Çiftliğin sesi

Öğrenme Senaryosu Adı	Çiftliğin sesi
Geçmiş Programlama Deneyimi	• Öğrenci, bir arka plan ekleyebilir. • Öğrenci, yeni bir karakter ekleyebilir. • Öğrenci, karaktere nasıl bir şey söyletebileceğini bilir.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: • Snap'ın medya kitaplığından ses ekleyin. • Diğer ortamlardan ses aktarın. • Yeni bir ses kaydedin. • Bir tuşa basarak ses çalın. Algoritmik düşünceye odaklı özel öğrenme çıktıları: • Öğrenci Snap'ın medya kitaplığından ses ekler ve belirli bir tuşa basıldığında sesi çalar, • Öğrenci bilgisayardan sesi alır ve belirli bir tuşa basıldığında çalar, • Öğrenci yeni bir ses kaydeder ve belirli bir tuşa basıldığında çalar.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Oyuncunun belirli tuşlara basarak hayvanların seslerini öğrendiği basit oyun programı. **Görevler:** İlk Adım'da öğrenci sahne arka planını seçmelidir. Daha sonra, öğrenci kadın çiftçiyi talimatları söylemesi için programlamalıdır. 1) Köpeği duymak istiyorsanız, "D"! tuşuna tıklayın. 2) İneği duymak istiyorsanız, "C"! tuşuna tıklayın. 3) Koyunu duymak istiyorsanız "S"! tuşuna tıklayın. 4) Domuzu duymak istiyorsanız "P"! tuşuna tıklayın. 5) Atı duymak istiyorsanız "H"! tuşuna tıklayın. Bundan sonra öğrenci, görevi kadın çiftçinin yönlendirdiği şekilde programlamalıdır. **Amaç:** Öğrencilere yeni bir sesin nasıl ekleneceği ve nasıl kullanılacağı anlatılacaktır. Ayrıca ses bloğunu block ("*play sound*

	[name_of_sound]”) ve kontrol bloğunu (“*when [the_key] key pressed”*) nasıl kullanacaklarını da öğreneceklerdir.
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme Oyun temelli öğrenme.
Öğretme Formları	Ön öğrenme Bireysel çalışma
Öğretme Özeti	Motivasyon-Giriş (Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Oyunu oynayarak öğrencileri motive ederiz (kodu görmezler). Dersin amacı oyunu böyle yapmaktır.  [Adım 1] İlk adım oyunun arka planını belirlemektir. Arka plan farklı hayvanlar içermelidir. Üç seçeneğimiz var. 1. Öğrenciler arka planı kendileri çizebilirler. 2. Öğrenciler çevrimiçi ücretsiz resim arayabilirler. 3. Öğrenciler için arka plan sağlarız (zamandan tasarruf etmek istiyorsak). Öğrenciler nasıl arka plan ekleyeceklerini önceden öğrendikleri için bu aktiviteyi bireysel yapabilirler.

[Adım 2]

İkinci adım kadın çiftçiyi eklemektir. İlk Adım'daki gibi seçeneklerimiz aynı.

1. Öğrenciler kadın çiftçiyi kendileri çizebilirler.

2. Öğrenciler internette kadın çiftçinin ücretsiz imajını arayabilirler.

3.Öğrenciler için kadın çiftçi imajını sağlayabiliriz (eğer zamandan kazanmak istiyorsak).

Öğrenciler yeni bir karakteri nasıl ekleyeceklerini önceden öğrendikleri için bu aktiviteyi bireysel yapabilirler.

[Adım 3]

Daha sonra öğrenciler oyuncu için talimatları programlamalıdır. Talimatlar kadın çiftçi tarafından verilmektedir. Öğrenciler bunu *Looks / say [string]* ya da *wait[n] (Görünümler Söyle [string] veya [n] kadar bekle)* bloklarını kullanarak yapabilirler. Öğrenciler bunu nasıl yapacaklarını önceden öğrendikleri için bu aktiviteyi bireysel yapabilirler.

Uygulama

Şimdi öğrencilere oyuna nasıl ses ekleneceğini göstereceğiz. Üç seçeneğimiz var.

1. Snap'in medya kitaplığından bir sesi içe aktarma.

2. Bilgisayarımızdan bir sesi Snap! içine sürükleyerek içe aktarma.

3. Snap!'te yeni bir ses kaydetme.

Öğrencilere her üç seçeneği de frontal öğretim şeklinde gösteriyoruz. Hepsini tanıttığımızda aşağıdaki görevleri ayrı ayrı programlamaya başlarlar (öğretmen desteği ile).

[Adım 4]

Öğrenciler köpeğin sesini programlamalıdır. Oyuncu "D" tuşuna bastığında köpeğin havlaması gerekir. İlk olarak öğrenciler sesi Snap'in medya kitaplığından arka planın ses sekmesine aktarır.

Akabinde çocuklar köpeğin sesini seçerler (Dog 1 veya Dog 2).

Öğrenciler “D” tuşuna basıldığında çalınacak olan köpeğin sesini programlamak zorundadır. Bunu *Control/when [the_key] key pressed* veya *Sound/play sound [name_of_sound] (Kontrol/ / …düğmesi basıldığında veya Ses/ sesi çal [sesin adı])*bloğunu kullanarak yaparlar.

[Adım 5]

Öğrenciler hayvan seslerini programlamalıdırlar. İlk olarak bilgisayarlarından ses eklemeleri gerekir. Bunu arka plan sesleri sekmesindeki sesleri sürükleyerek yaparlar.

Sesleri içe aktardıktan sonra yeniden adlandırmak için sesleri sağ tıklayabiliriz. Bizim durumumuzda onlara inek, domuz, at ve koyun

şeklinde ad verebiliriz.

Daha sonra öğrencilerin sesi arka planın komut dosyalarına eklemesi gerekir. Bunu *Control/when[the_key] key pressed ve Sound/play sound[name_of_sound] (Kontrol/ / …düğmesi basıldığında veya Ses/ sesi çal [sesin adı])* bloklarını kullanarak yapabilirler.

[Adım 6]

Sonraki adım kadın çiftçinin karşılama selamlamasını programlamaktır. Oyuncu oyuna başladığında kadın çiftçi "çiftliğime hoş geldiniz" demelidir. İlk olarak öğrenciler kadın çiftçinin karşılama selamlamasını kaydetmelidir. Bunu, (kadın çiftçinin) Sesler sekmesinde bulunan ses kaydedici (kırmızı düğme) ile yapabilirler. Sesi kaydettiklerinde (record), bilgisayara kaydetmeleri (save) gerekir.

Ses kaydı tamamlanınca sağ tıklayıp ismi değiştirebilirler. Bizim

örneğimizde isim çiftlik olacak.

Şimdi öğrenciler sesi kadın çiftçinin sesinin dosyasına (script) kaydetmeliler. Bunu *sound/play sound[name_of_sound] (ses/ sesi çal [sesin adı])* bloğu 

ile yapabilirler.

Ek Görev:

Öğrenci, yeni karakter (çiftçi, tavuk, traktör, ...) ve sesler ekleyerek çiftliği istediği gibi geliştirebilir.

Düşünme ve değerlendirme

Öğrenciler aşağıdakileri özetlerler:

- Kodlarına sesleri nasıl eklediklerini,
- Koda ses eklemek için hangi blokları kullandıklarını,
- Kodlarında hangi denetim bloklarını kullandıklarını,
- Neden ve nasıl ses bloklarını ve kontrol bloklarını kullandıklarını.

	[Nihai kod] *Kadın çiftçi*  *Arka plan* 
Öğretmenler İçin Araçlar ve Kaynaklar	• Snap! Sitesindeki örnek: https://snap.berkeley.edu/project?user=tadeja&project=Farm • İmaj için: https://pixabay.com/ • Ses için: https://www.zapsplat.com/ • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. • Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	• Snap!'ten örnek https://snap.berkeley.edu/project?user=tadeja&project=Sounds%20of%20the%20farm_0 • İmaj: https://pixabay.com/ • Ses: https://www.zapsplat.com/ • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 6 - Bukalemunun yaz tatili, basit sürüm

Öğrenme Senaryosu Adı	Bukalemunun yaz tatili, basit sürüm
Geçmiş Programlama Deneyimi	Ön programlama becerileri gerekmemektedir.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Olaya dayalı nesne hareketi ● Tekli veya çoklu renk algılama ● Mantıksal ifadelerde Boolean değer okumaları ● Farklı oyun durumlarını tanımlama, ayırt etme, dinamik olarak kontrol etme ve bunlara yanıt verme Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, olayları kullanarak ok tuşlarıyla nesne hareketini gerçekleştirir ve kısıtlamaları dikkate alır. ● Öğrenci, tekli veya çoklu renk algılama okuması için boole değerini elde etmek için bir algılama renk bloğu kullanır. ● Öğrenci, nesne durumunun nesnenin dokunduğu renklerle ifade edilebileceğini fark eder. ● Öğrenci iki temel, beş (tam) farklı durumu ayırt eder ve bunları mantıksal ifadelerle nasıl ifade edeceğini bilir. ● Öğrenci nesnenin konumunun dinamik olarak değiştiğini fark eder. Mevcut durumu tekrar tekrar kontrol etmek için sonsuza kadar döngü (forever loop) kullanır. ● Öğrenci, nesnenin mevcut konumuna bağlı olarak farklı yanıtlar vermek için if/koşul cümlesini kullanır.

Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Nesnenin arka plan rengine göre kostümünü değiştireceği basit bir oyun programlayın. Görevler: Öğrenciler bukalemunu programlayarak görünüşünü (kostümü) değiştirmeli ve beş farklı durumda nerede olduğunu söylemelidir: 1) Denizde yüzerken rengini maviye çevirmeli ve "denizde yüzüyorum" 2) Deniz ve kumsal arasındayken cildi yarı mavi-yarı kum rengine döner ve "deniz ile sahil arasındayım" der. 3) Kumsalda kum rengi alır ve "kumsalda rahatlıyorum" der. 4) Kumsal ile orman arasında yarı yeşil-yarı kum rengine döner ve "kumsal ile orman arasındayım" der. 5) Ormanda cildi yeşile döner ve "ağacın gölgesinde serinliyorum" der. **Öğrencilere, dinamik olarak değişen oyun durumları arasında ayrım yapmak ve doğru yanıtları vermek için renk bloğunu algılama ve mantıksal ifadelerde bloğun nasıl kullanılacağı anlatılacaktır.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirliğine dayalı öğrenme, problem çözme
Öğretme Formları	Ön öğrenme Bireysel Çalışma/İkili Çalışma/Grup Çalışması
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Bukalemun bir yaz tatiline çıktı. Denizde yıkanmayı, plajda dinlenmeyi, hava çok sıcak olduğunda serinlemek için yakındaki ağaçların barınağına gitmeyi seviyor. Bukalemun olduğu için rengini şimdiki arka plana göre değiştirir. **[Temel Versiyon]** Bu temel versiyonda iki farklı hâl arası geçiş yapacağız.

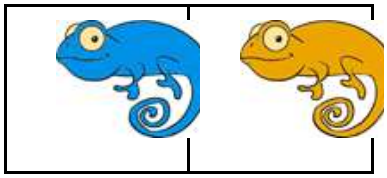
[Adım 1]

Öğrencilerden sahne arka planını, her biri farklı bir yeri temsil eden mavi ve kumlu aynı renkte iki parçaya bölünecek şekilde düzenlemelerini istiyoruz. Mavi renk deniz için, kum rengi (sandy) ise kumsal içindir. Öğrencilere arka planı daha gerçekçi hale getirmek amacıyla diğer öğeleri dâhil etmeleri için talimat verebiliriz. Örneğin: dalgalar, deniz kabukları, kumdan kaleler, güneş şemsiyeleri, vb.

Öğrencilerin daha büyük ve arka plandan tamamen farklı renklerle renklendirilmiş öğeleri seçmemeye dikkat etmeleri gerekir. Bu durumda renk algılama bloğu, karakterin sahnenin hangi bölümünde olduğunu algılayamayacaktır.

[Adım 2]

Öğrencilerin bukalemunu çizip, cildini iki farklı renkle boyamaları gerekmektedir.



[Adım 3]

Öncelikle bukalemunlarını tuşları kullanarak dört yönde hareket ettirmeleri gerekir. Kendi tuş kombinasyonlarını seçebilirler (Ör: Ok tuşları veya WASD-klavye harfleri). Bu noktada, önceki faaliyetlerden bunu nasıl yapacaklarını bildiklerini varsayıyoruz. Öğrencilere hareketi programlarken uygun blok kullanmazsak karakterin sahnenin dışına çıkabileceğini hatırlatmalıyız (sekme kenar bloğundaysa).

Bukalemun hareketini biraz daha gerçekçi kılmak için baktığımız yatay yöne bakacak şekilde sola veya sağa dönmesini istiyoruz (*point in direction / yönüne dön* bloğu kullanarak).

[Adım 4]

Öğrencilere, dokunduğu rengi (renkleri) algılayan karakter kavramını tanıtıyoruz. "touching color?" bloğu ile Boolean değerleri (-belirli bir renge dokunuyorsa Doğru veya Yanlış) biçiminde bilgi alabiliriz. Bu bloktan Boolean değeri aldığımız için, onu gövdesinde listelenen komutları çalıştırıp çalıştırmayacağımıza karar verildiği If/Koşul cümlesinin başında kullanabiliriz.

Daha sonra öğrencilerle bukalemunun sahnedeki farklı pozisyonlarının neler olabileceğini ve bunları “touching color? (renge dokunur)” bloğu kullanarak nasıl ifade edebileceğimizi öğretebiliriz.

İşte iki örnek;

1. Mavi renge dokunuyor -> Touching color [blue]?

2. Kumlu renge dokunuyor -> Touching color [sandy]?

Belli bir renge dokunduğunda görünüşünü değiştirmeli ve ona nerede olduğunu söylemesini sağlamalıyız. Kostümleri arasında geçiş yaparak bir karakterin görünümünü değiştirebiliriz. Bu, olası kostümlerden hangisini görüntülemek istediğimizi seçtiğimiz *Looks/switch to costume[option]* (Görünümler / kostüme geçiş

[seçenek]) bloğuyla yapılır. Bukalemun konuşmasını sağlamak için *Looks/say[text] (*Görünüm / söyle [metin]) bloğunu kullanıyoruz.

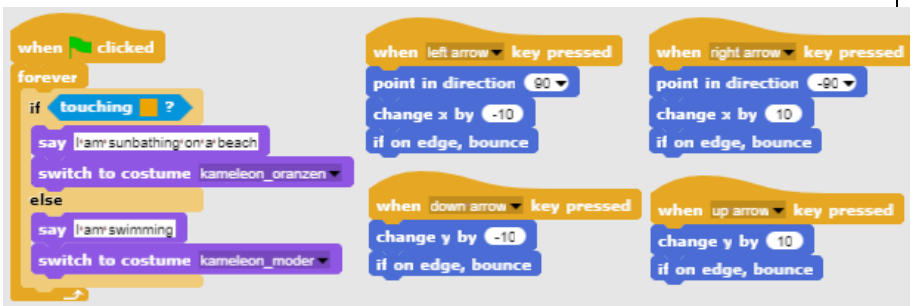
Çünkü "if - else" koşullu bloğunu kullanabileceğimiz sadece iki olasılık vardır.

Hangi rengi kontrol edeceğimizi seçebiliriz. Ardından diğer renkler "else" durumuna girecektir. Örnek kodda kumlu (sandy) rengi seçtik.

[Adım 5]

Kullandığımız programın tüm süresi boyunca belirli komutları yürütmemiz gereken durumlar için – forever loop (- sonsuza kadar döngü) butonunu kullanırız. Sonsuz döngünün gövdesi altında yazılan her şey defalarca yürütülecektir. Öğrencilerimizle bu oyunu oluşturmak için bizim durumumuzda tam olarak istediğimizin/ihtiyacımızın bu komut olduğunu tartışıyoruz.

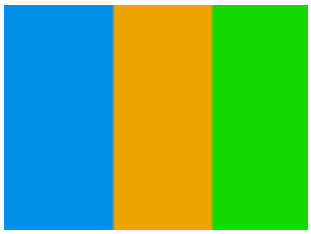
[Nihai Kod]



[Tam Versiyon]

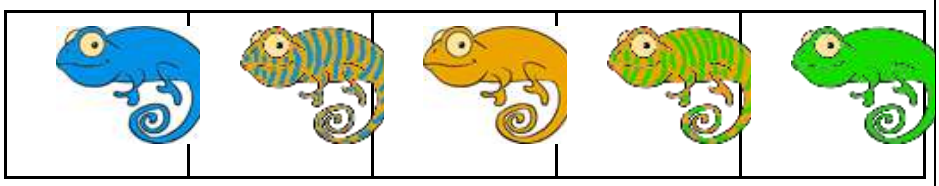
[Adım 1]

Öğrencilerden sahne arka planını, her biri farklı bir yeri temsil eden aynı renkte üç parçaya bölünecek şekilde düzenlemelerini istiyoruz. Deniz için mavi renk, kumsal için kum rengi ve orman için yeşil. Arka planı daha gerçekçi hale getirmek için dalgalar, deniz kabukları, kumdan kaleler, güneş şemsiyeleri, ağaçlar vb. gibi başka öğeler ekleyebilirler. Ancak eklenen öğelerin ana karakterin kendisinden daha büyük olmamasına dikkat etmeleri gerekir. Çünkü bu durum karakteri üç renkten hiçbirine dokunmayacak ve Snap'in algılama özelliği karakterin sahnenin hangi bölümünde olduğunu algılayamayacaktır.



[Adım 2]

Öğrencilerin bir bukalemun çizmeleri ve derisini sahnedeki konumunu temsil eden beş farklı kombinasyonda boyamaları gerekiyor.



[Adım 3]

Öncelikle bukalemunlarını tuşları kullanarak dört yönde hareket ettirmeleri gerekir. Kendi tuş kombinasyonlarını seçebilirler (Örnek:

Ok tuşları veya WASD). Bu noktada, önceki faaliyetlerden bunu nasıl yapacaklarını bildiklerini varsayıyoruz. Öğrencilere, hareketi programlarken uygun blok kullanmazsak karakterin sahnenin dışına çıkabileceğini hatırlatmalıyız (sekme kenar bloğundaysa).

Bukalemun hareketini biraz daha gerçekçi kılmak için, baktığımız yatay yöne bakacak şekilde sola veya sağa dönmesini istiyoruz (point in direction bloğu kullanarak).

[Adım 4]

Öğrencilere, dokunduğu rengi (renkleri) algılayan karakter kavramını tanıtıyoruz. "touching color?" bloğu ile Boolean değerleri (-belirli bir renge dokunuyorsa Doğru veya Yanlış) biçiminde bilgi alabiliriz. Bu bloktan Boolean değeri aldığımız için onu gövdesinde listelenen komutları çalıştırıp çalıştırmayacağımıza karar verildiği If/Koşul cümlesinin başında kullanabiliriz.

Daha sonra öğrencilerle bukalemunun sahnedeki farklı pozisyonlarının neler olabileceğini ve bunları “touching color?” bloğu kullanarak nasıl ifade edebileceğimizi öğretebiliriz.

Hızlıca beş tane olduğunu anlıyoruz.

1. Tamamen mavi kısımda -> Touching color [blue]? ([mavi] renge dokunuyor mu?)

2. Mavi ve kumlu kısım arasında -> Touching color[blue]? VE Touching color [sand]? ([mavi] renge dokunuyor mu? VE Renkli [kum]

dokunmak?)

3. Tamamen kumlu kısımda -> touching color [sand]? (renkli [kuma] dokunuyor mu?)

4. Kumlu ve yeşil kısım arasında -> Touching color[sand]? VE Touching color [green]? (Rengine [kum] dokunuyor mu? VE Renkli [yeşil] dokunmak?)

5. Tamamen yeşil kısımda -> Touching color [green]? ([yeşil] renge dokunuyor mu?)

Belli bir renge/renklere dokunduğunda bukalemun görünümünü değiştirmeli ve ayrıca ona nerede olduğunu söyletmeliyiz. Kostümleri arasında geçiş yaparak bir karakterin görünümünü değiştirebiliriz. Bu, olası kostümlerden hangisini görüntülemek istediğimizi seçtiğimiz *Looks/switch to costume[option]* (Görünümler/kostüme geçiş [seçenek]) bloğuyla yapılır. Bir bukalemun konuşması için Looks/say [text] bloğunu kullanıyoruz.

Öncelikle, bukalemunun sahnenin tamamen aynı renk kısmında olduğu daha basit durumlarla ilgileniyoruz:

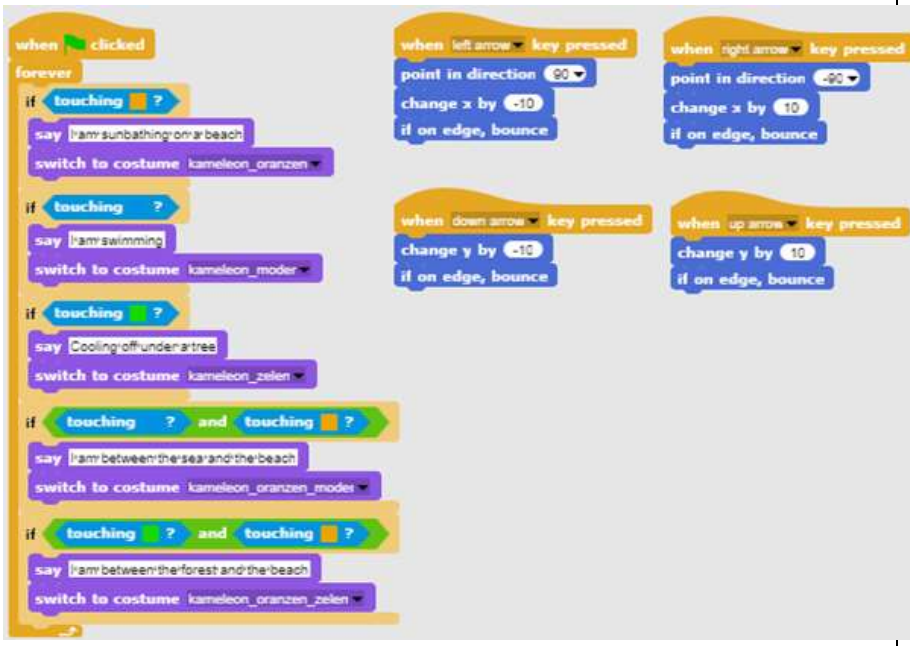
Daha sonra VE mantıksal işlemci kullanarak mantıksal bir ifade oluşturuyoruz. Çünkü bukalemunun aynı anda iki renge dokunup dokunmadığını doğrulamak istiyoruz.

Yukarıdaki koşullu cümleleri birleştirip *When Green Flag Clicked* (Yeşil Bayrak Tıklandığında) bloğunun altına koyarsak bu koşulların tam olarak bir kez kontrol edileceğini görürüz. Ana karakterin hareketini kontrol ettiğimiz için bukalemun pozisyonunun oyun sırasında sürekli değişeceğini fark etmelerine yardımcı oluyoruz. Bu nedenle, bu koşulları sürekli olarak yalnızca bir kez değil, kelimenin tam anlamıyla her zaman kontrol etmemiz gerekiyor!

[Adım 5]

Programın tümünün çalışması için belirli komutları yürütmemiz gereken durumlar için – forever loop (- sonsuza kadar döngü)'u kullanırız. Sonsuz döngünün gövdesi altında yazılan her şey defalarca yürütülecektir. Öğrencilerimizle bu oyunu oluşturmak için bizim istediğimiz / ihtiyacımızın bu komut olduğunu tartışabiliriz.

[Nihai kod]



[Öğrenciler Kodu Ayarlarlar]

Bu aktiviteyi basitleştirmek için kodun bir kısmını önceden bir şablon dosyasında hazırlayabilir ve öğrencilere bunu tamamlamaları için talimat verebiliriz.

Önerilen öğrenme yolunu izleyen öğrenciler nesneyi anahtarlarla hareket ettirmeyi zaten öğrendi. Böylece hareket kodunu bir şablon dosyasına ekleyebiliriz. Tuş ayarlarını ok tuşlarından özel düzenlemeye (örneğin WASD) değiştirebilirler.

Sonsuz döngü kavramını (forever loop) ve arka plan rengini algılamak için sonsuz döngüyü nasıl kullanılacağını anlamalarına yardımcı olmak ve iki durumu algılamak için kod ekleyebiliriz:

1) Nesne tamamen tek renktedir.

2) Nesne aynı anda iki renge dokunmaktadır.

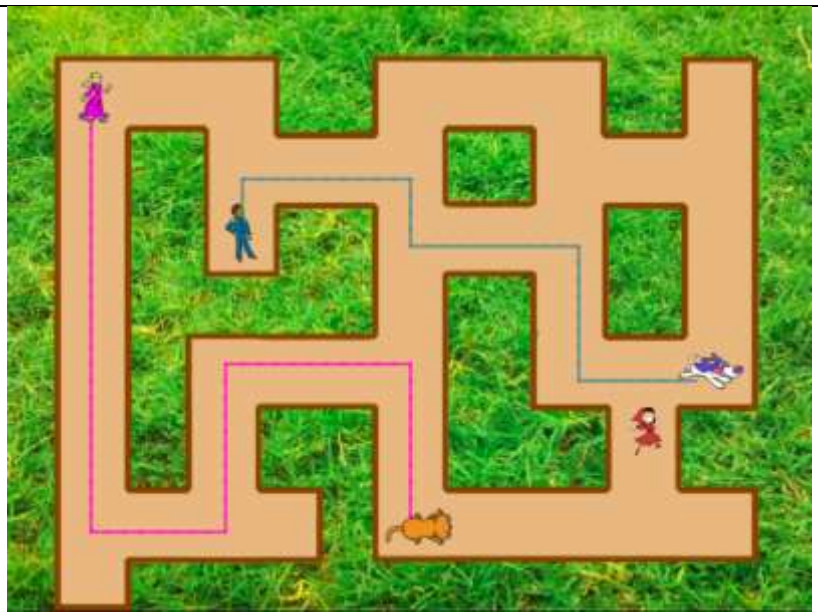
Her vaka için kodu tamamlamaları talimatını verdik.

Önerilen kod şablonu:

	
Öğretmenler İçin Araçlar ve Kaynaklar	• Snap!: Temel: https://snap.berkeley.edu/project?user=zapusek&project=chameleon_simple Tam: https://snap.berkeley.edu/project?user=zapusek&project=chameleon • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. • Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	• Snap! Şablonu: https://snap.berkeley.edu/project?user=zapusek&project=chameleon_template • Snap!'teki yarı tamamlanmış (Half-Baked) aktivite https://snap.berkeley.edu/project?user=zapusek&project=chameleon_half_baked Öğrenci için talimatlar (C4G2_InstructionsForStudent.do

Öğrenme Senaryosu Adı	Prense ve prensese evcil hayvanlarını bulmada yardım etme
Geçmiş Programlama Deneyimi	Karaktere metin ekleme Events (Olaylar) kullanarak ok tuşlarıyla nesne hareketi Nesne için *object is touching (nesne değiyor)* koşulunu kullanmak Event (Olaylar) kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● *Object is touching* için koşul ifadeleri ● Koordinatlara geçme ● Kalem yukarı, kalem aşağı (pen up/pen down) ● Kalem rengi Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, belirli bir renge dokunursa, nesne durumu için if cümlesini kullanır ve nesneyi geri koyar. ● Karakter için x ve y koordinatlarını başlatır. ● Öğrenci, çizgi/yol çizmek için kalem yukarı ve aşağı kalem özelliklerini kullanır. ● Öğrenci bağlandığı çifte göre kalem rengini değiştirir. ● Öğrenci, başlangıçta önceki tüm yolları temizlemesi gerektiğini fark eder.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Açıklama:** Kızlar, Prenses'e kedisini; Prens'e köpeğini bulması için yardım etmelidir. Bunu prensese gidip ona bir çizgi çizerek kedisinin yolunu göstererek yapacağız. Benzer şekilde kızlar Prens'e köpeğine giden yolu gösterecekler. Ancak kızlar, yollarının kesişmemesi için hayvanlar arasındaki çakışmalardan kaçınmalıdır. **Görevler:** İlk Adım'da öğrenciler uygun arka planı (labirent) seçmelidir. Labirente beş karakter eklerler. Karakterleri bir kız, bir prenses, bir prens, bir kedi ve bir köpek. Daha sonra kız için düğmelerle hareket etmeyi (events kullanarak) programlayacaklar. Ancak burada karakterin çimlerin üzerinde yürümemesine özen gösterilmelidir. Akabinde kalemle çizim ve events ile kalem rengini değiştirmeyi programlarlar. Ayrıca, yolu temizleyen ve kıza talimatları veren başlangıç olayını programlamaları

	gerekir. **Amaç: Öğrencilere anahtar hareketlerle çizim yaptırılacaktır. Bunun yanında, karakterin tüm ekranda hareket etmesini önlemek için koşul ifadelerini nasıl kullanacaklarını öğrenecekler.**
Faaliyetin Süresi	30 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, oyun tasarım tabanlı öğrenme, problem çözme
Öğretme Formları	Ön öğrenme Bireysel çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Aşağıdakiler öğrencilere başta verilir: ● Arka plan ● Kız karakter ● Tek yönde hareket kodu Kız, Prensese kedisini bulması ve Prensin köpeğini bulması için hayvanlarına giden yolu göstererek (çizerek) yardım etmeye karar verir. Karışıklığı önlemek için yollar farklı renklerde olmalı ve kesişmemelidir.



[Adım 1]

Öğrencilerden sahne arka planını bir labirent olarak düzenlemelerini istiyoruz. “if touching color” (renge dokunuluyorsa) uygulaması için ya arka planın (çimen) tek renkli olması ya da bizim durumumuzda olduğu gibi yolun tek renkli bir çerçeveye sahip olması gerekir. Uygun arka planı bulmada bu “sorunları” önlemek için onlara bu arka planı veriyoruz.

[Adım 2]

Öğrenciler aktivitenin başında zaten bir kız karakter oluşturmuşlardı. Başka dört tane karakter daha bulup labirente koymaları gerekiyor. Tüm hareketli resimler için boyutu uygun (labirentteki yolların genişliğinden daha küçük olan) şekilde ayarlamaları gerekir. Her hareketli grafik için aşağıdaki kodu kullanabilirler.

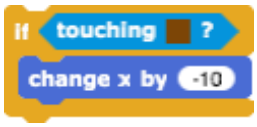
Kız karakter için önerilen büyüklük 8%’dir. Diğer karakterler daha büyük olabilir.

[Adım 3]

Bundan sonra, kızın hareketini tuşları kullanarak dört yönde yapmaları gerekmektedir. Önceki faaliyetlerden bunu nasıl yapacaklarını zaten bildiklerini varsayıyoruz. Onlara bir yön için kod veriyoruz. Bu da onların başka bir üç yön yapmasına yardımcı oluyor.

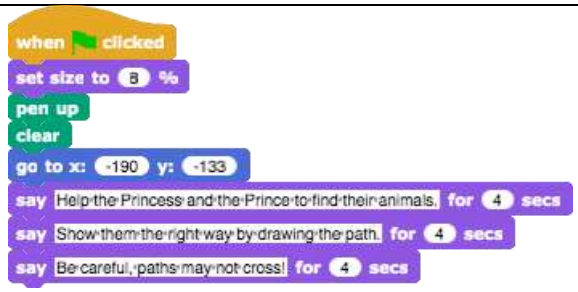
[Adım 4]

Bir sonraki adımda, kızın çayır boyunca hareket etmesini önlemek zorundalar. Bunu, kahverengi renge dokunuyorlarsa koşullu bir blok ekleyerek yapabilirler. Kız kahverengine (yolun sonuna) dokunuyorsa 10 adım geri gider. O iki adımı görmüyoruz ve sanki kız aynı pozisyonda kalıyor. Bu, sağa hareket etmek için bir koddur. Yani x'i -10 ile 10 adım geri, değiştirmek anlamına gelir.



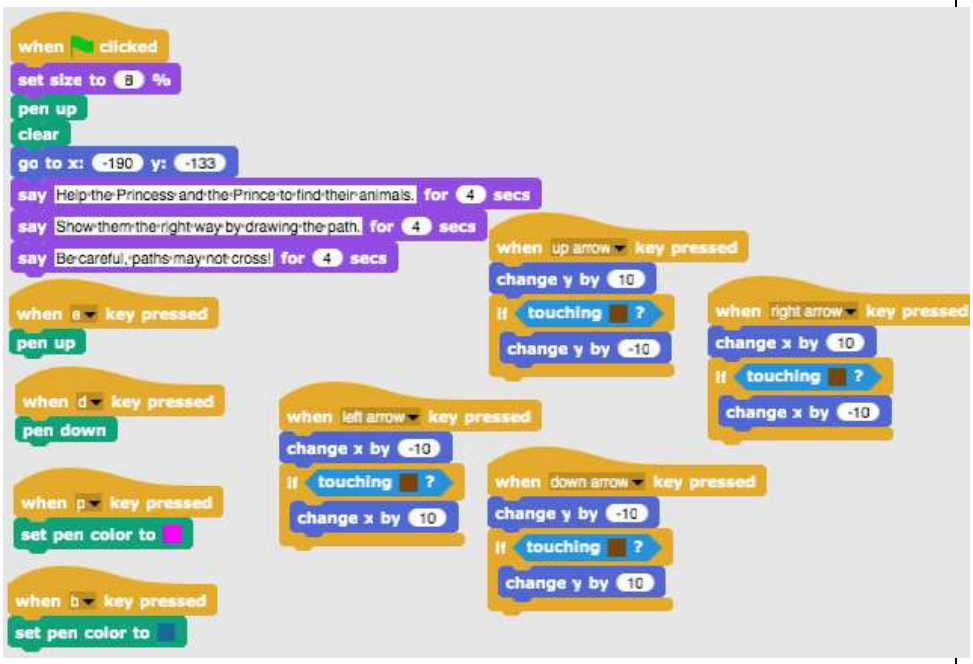
Bu kodu önceki kodun altına eklerler. Ör: Sağ ok için.

	Diğer üç yön için de benzer şeylerin yapılması gerekiyor. [Adım 5] Daha sonra çizimi programlarlar. Bunu, *when key pressed* (tuşa basıldığında) olayını (events) kullanarak kalem yukarı ve aşağı kalem blokları yaparak gerçekleştirirler.  “D” tuşuna basıldığında ve kız hareket ettiğinde bir çizgi çizer. "E" tuşuna basıldığında çizim durur. Benzer şekilde, tuşa basarak kalem rengini ayarlarlar.  [Adım 6] Son olarak, öğrenciler kızın başta söylediği bazı talimatları ekledikleri “*when clicked green flag”* (yeşil bayrak tıklandığında)’ı programlıyorlar. Oyunu oynarken durdurun ve tekrar oynayın. Öğrenciler aşağıdaki blokları buna eklemenin iyi olduğunu göreceklerdir. Kalem yukarı (*pen up*) (önceki oyundan kalması durumunda), temizle (*clear*) (önceki oyundan yolu temizler) ve x ve y'ye git (*go to x, y*) (kız her zaman çimenlerin üzerinde değil, yolun içinde olan bu koordinatlarda başlar). Kızın başlangıç koordinatlarını belirlemek için fareyle bir kızı alıp başlamasını istediğimiz yere bırakıyoruz. Ardından x konumunu ve y konumunu bulabileceğimiz hareket bloklarına (*motion* blocks) tıklıyoruz. X konumuna tıklayarak kızın x konumunu ve aynı şekilde y'ye tıklayarak y konumunu bulabiliriz.



[Nihai Kod]

Kız



Örneğin Prenses

[Ek Görevler]

Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler.

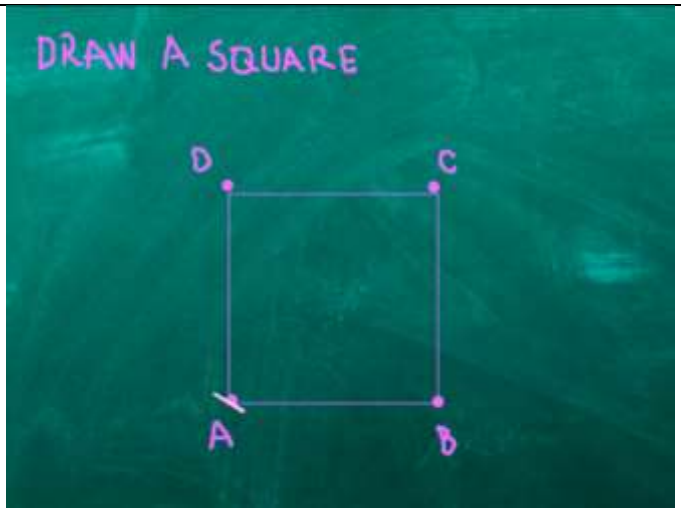
● Prens ve prenses için başlangıç koordinatlarını ayarlayın ve hareketleri için bir kod yazın. Onlar için uygun boyutu ayarlayın. Hayvanlarına bir yol çizmeliler.

	● Kız için başka bir karakter (hayvan) ekleyin. ● Her hareketli grafik farklı bir renkle çizilmelidir. ● Başlangıç talimatlarını ayarlayın. ● Bir hareketli grafiği taşımak ve bir hareketli grafiği tıklatarak çizim yapmak için talimatlar ekleyin. Örneğin, prenses diyor ki: "W, S, A ve D tuşlarına basarak beni hareket ettiriyorsun. Tuşa basarak yolu çiziyorum 3. Tuşa basarak çizmeyi bırakıyorum 4. Kedimi bulmama yardım et!"
Öğretmenler İçin Araçlar ve Kaynaklar	● Snap! te tüm aktivite mevcut: https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals ● Snap! te çözümleri ile ek görevler https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals%20%2B%20Add.%20Task ● Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. ● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'te yarı tamamlanmış (Half-Baked) aktivite. https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals%20-%20Part ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 8 - Tebeşir ile çizim yapmak

Öğrenme Senaryosu Adı	Tebeşir ile çizim yapmak
Geçmiş Programlama Deneyimi	Karaktere metin ekleme Kalem ile çizim (kalem yukarı, kalem aşağı, renk belirleme) Adımlarla Hareket Etmek Döngüler(Loops) kullanma Olayları (Events) kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: • Tekrarlama döngüsü • 90 derece döndürme • Bir yöne dönme • Arka planı değiştirme Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, aynı bloklar 2/4 kez tekrarladığında döngü tekrarını (loop repeat) kullanır. ● Öğrenci, farklı şekiller (kare, dikdörtgen, "T" harfi) çizerken 90 derece döndürmeyi kullanır. ● Öğrenci, 90 yönündeki anlam noktasını anlar. ● Öğrenci, bir tuşa basıldığında bir olay kombinasyonuyla arka planı nasıl değiştireceğini bilir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Oyuncu üç farklı arka plan elde etmek ve noktaları üç farklı şekle bağlamak zorundadır. Şekilleri oluşturmak için - bir kare, bir dikdörtgen ve bir "T" harfidir. **Görevler:** Öğrenciler "tahtaS" (boardS) arka planını seçerler ve bir kare çizmeye başlarlar. Başlangıç konumları "A" noktasıdır. Kare çizerken belirli adımları 4 kez tekrar ederler. Yani aynı kodu 4 kez yazmak yerine 4 kez döngü tekrarı kullanabilirler. Sonra yine bir döngü tekrarı kullanarak bir dikdörtgen çizerler. Bu sefer 2 kez tekrarlanır. Son görevlerinde adımların sayısını bulmaları gereken noktaları "T" harfi şeklinde

	birleştirmeleri gerekiyor. Mümkün olduğunda döngü tekrarını kullanabilirler. **Amaç: Bu etkinlikte öğrenciler bir kodla farklı şekillerin nasıl çizileceğini öğreneceklerdir. Ayrıca kodu kısaltmak ve arka planı değiştirmek için döngü tekrarını kullanmayı öğreneceklerdir.**
Faaliyetin Süresi	60 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, oyun temelli öğrenme, problem çözme
Öğretme Formları	Ön öğrenme Bireysel Çalışma/ Grup Çalışması
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Aşağıdakiler öğrencilere başlangıçta verilir. ● Birleştirmeleri gereken tüm noktaların bulunduğu üç arka plan ● Tebeşir karakteri (Sprite) Tebeşir bir kare, bir dikdörtgen çizmek ve "T" harfi şeklindeki noktaları birleştirmek istemektedir. Ancak nasıl hareket edeceğini ve nasıl döneceğini bilemez. Bir kod yazın ve tebeşire nasıl yapılacağını gösterin! [Adım 1]



Öğrenciler yukarıdaki arka planla başlar. Kare çizmek için bir kod yazarlar. "A" noktasından başlayarak X Adım kadar "B" noktasına hareket ettirirler. Sola 90 derece dönerler. X Adım kadar "C" noktasına hareket ettirirler. 90 derece sola dönerler. X Adım kadar noktaya "D", sola 90 derece dönün. "A" noktasına X adım getirin (ve sola 90 derece dönün).

90 derece döndürme'yi (*turn 90 degrees*) kullanmak en kolay yoldur. Çünkü her zaman 90 derece döndürmeyi kullanabiliriz (sadece sola veya sağa dönüp dönmemize bağlıdır). 0, 90, 180, -90 yönünde nokta kullanmak ise bir diğer seçenektir. Ancak bu biraz daha karmaşıktır. Çünkü 4 olasılığı ayırmak zorundayız ve bir döngü tekrarı kullanamıyoruz.

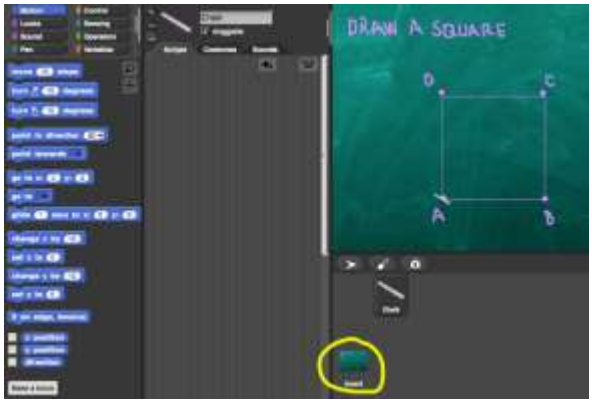
	Sadece çizimi/tüm adımları görmek için *1 sn. Bekle* (*Wait 1 secs*) bloğu eklenmiştir. Bu blok olmadan tüm kod bir saniyede gerçekleşir. Öğrencilerin bunu anlayabilmeleri için bu blok olmadan denemelidir. Öğrenciye, mümkünse kodu nasıl kısaltacaklarını soruyoruz. Tekrarlardan başka bir kısım var mı? Cevap Evet. Aynı kodu 4 kez yazmak yerine, programlamada döngü tekrarı kullanıyoruz.  Gerçekten ne çizdiğimizi görmek istiyorsak *tekrar (repeat)* döngüsünden önce bir *kalem aşağı (pen down*) bloğu koymalıyız. Tebeşir dönüş gerçekleştirirken etrafında dönmemesini istiyorsak yön bloğunda *döndürme* (*don't rotate*) seçeneğine tıklarız.  [Adım 2] Kodu etkinleştirmek için öğrenciler *olay (event)* bloğunu kullanır. Örneğin *S tuşuna basıldığında (when S key is pressed)*. Ayrıca kalem rengini de ayarlayabilirler ve önceki aktivitelerden bildikleri gibi aşağıdaki blokları takip edebilirler. *Kalem yukarı (pen up)* (önceki oyundan kalması durumunda), *temizle (clear)* (önceki oyundan çizimi temizler) ve *x, y'ye git (go to x, y)* (tebeşir her zaman bu koordinatlarda başlar). Bazen oyun sırasında programı durdurduğumuz ve ardından bir karakteri "garip bir yöne" döndürüldüğümüz olur. Bu bir oyuna yeniden başlarken sorun olabilir. Eğer bir karakter yanlış döndürülürse örneğin ilk adımda sağa değil aşağıya gidecektir. Bu sorunu önlemek için *90 yönünde*

dön (point in direction 90) bloğu ekliyoruz.

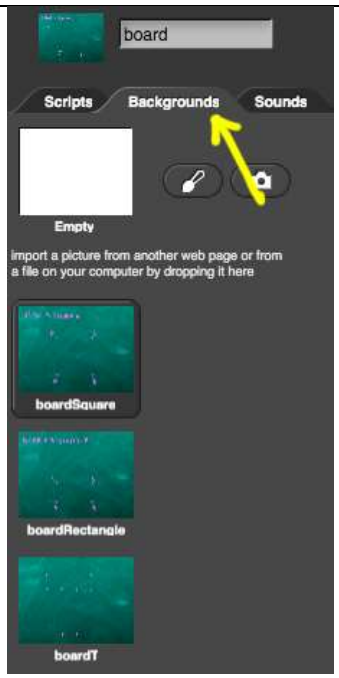
[Adım 3]

Bir kare çizdikten sonra bir dikdörtgen çizmek istiyoruz. Bu, arka planı değiştirmemiz gerektiği anlamına gelir. Bunu iki Adımda yapacağız:

a) Arka plana tıklıyoruz (ekranın sağ tarafında adlı pano).

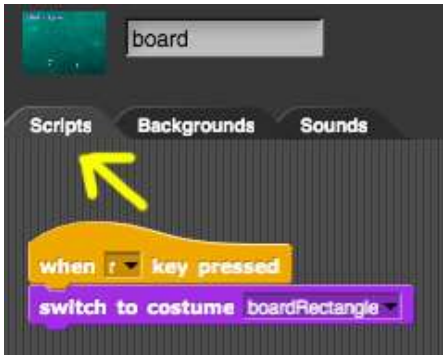


Arka planlara tıkladığınızda, bu etkinlik için önceden hazırlanmış olan üç gerekli arka planı (tahta Kare, tahta Dikdörtgen, tahtaT (*boardSquare, boardRectangle, boardT*)) görebiliriz.

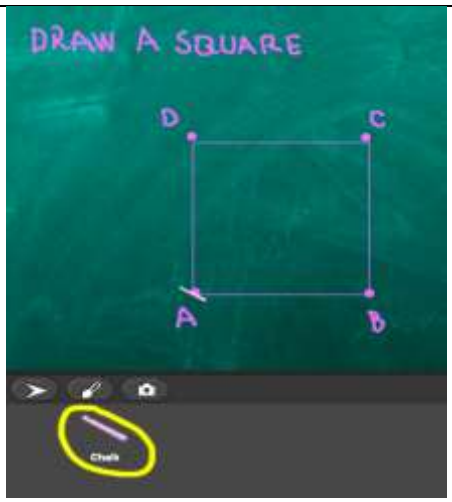


Bir kod yazmak için öğrencilerin Komut Dosyalarına tıklamaları gerekir.

Değişen arka planı programlamak için R tuşuna basıldığında bir event (olay) bloğu seçerler ve ardından *BoardRectangle (dikdörtgen pano) kostümüne geçin* tuşuna basarlar (*switch to costume boardRectangle*).

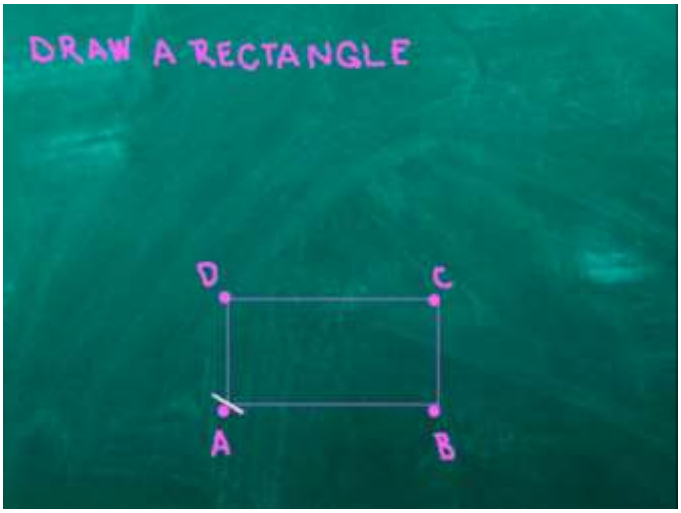


b) Tebeşir üzerine tekrar tıklıyoruz.



[Adım 2] kodunun altına, öğrenciler bir oyuncuya arka planı değiştirmek için ne yapmaları gerektiğini söyleyecekleri bir blok eklerler. Yani "R" tuşuna basın.

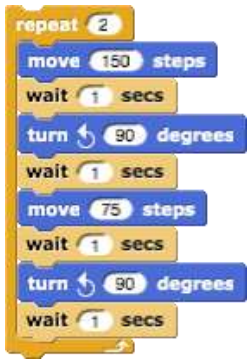
[Adım 4]



"R" tuşuna bastıktan sonra arka plan buna dönüşür. Öncekine benzer şekilde noktaları birleştirmeleri ve bir dikdörtgen çizmeleri gerekir. Öğrenciler önceki kod bloklarını kopyalayabilir ve programın bir dikdörtgen çizmesi için bunları düzeltebilirler.

Döngü tekrarını (*repeat*) değiştirirler. Şimdi bu döngü 2 kez

tekrarlanacak.



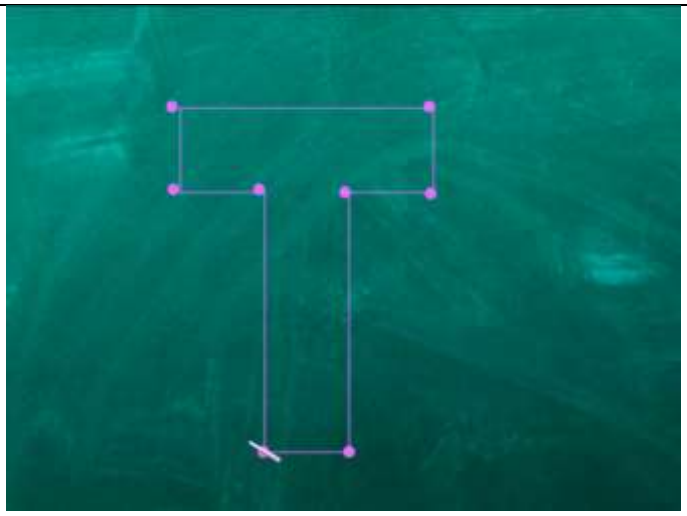
[Adım 5]

Bir dikdörtgen çizdikten sonra öğrenciler noktaları "T" harfi şeklinde birleştirecekler. Bu, arka planı değiştirmeleri gerektiği anlamına geliyor. bu yüzden bu adımda aslında [Adım 3]'ü tekrar ediyorlar. Sadece ("T") harfi ve kostümü (boardT) değiştiriyorlar.

a) Arka planı değiştirmek için bir kod yazdıkları arka plana (ekranın sağ tarafındaki *board* adlı panoya) tıklarlar. Bunu *T tuşuna basıldığında (when T key pressed)* ve ardından *kostüm T panosuna geç (switch to costume boardT)* kullanarak yapacaklar.

b) Tebeşire ve [Adım 4] ün altındaki koda blok eklemek için tekrar tıklarlar. Burada oyuncuya arka planı değiştirmek için ne yapması gerektiğini söylerler. Yani “T” tuşuna basın.

[Adım 6]



"T" tuşuna bastıktan sonra arka plan buna dönüşür. Öncekine benzer şekilde noktaları birleştirmeleri ve bir "T" harfi çizmeleri gerekir. Öğrenciler önceki kod bloklarını kopyalayabilir ve düzeltebilirler.

Öğrenciler, öncekiyle aynı olmayan başlangıç koordinatlarını değiştirmek zorunda kalacaklar. Önceki faaliyetten doğru koordinatları nasıl belirleyeceklerini zaten biliyorlar.

Daha sonra "T" harfini çizmek için bir kod yazarlar. Adımların sayısını bulmaları gerekir. Olası bir çözüm şudur:

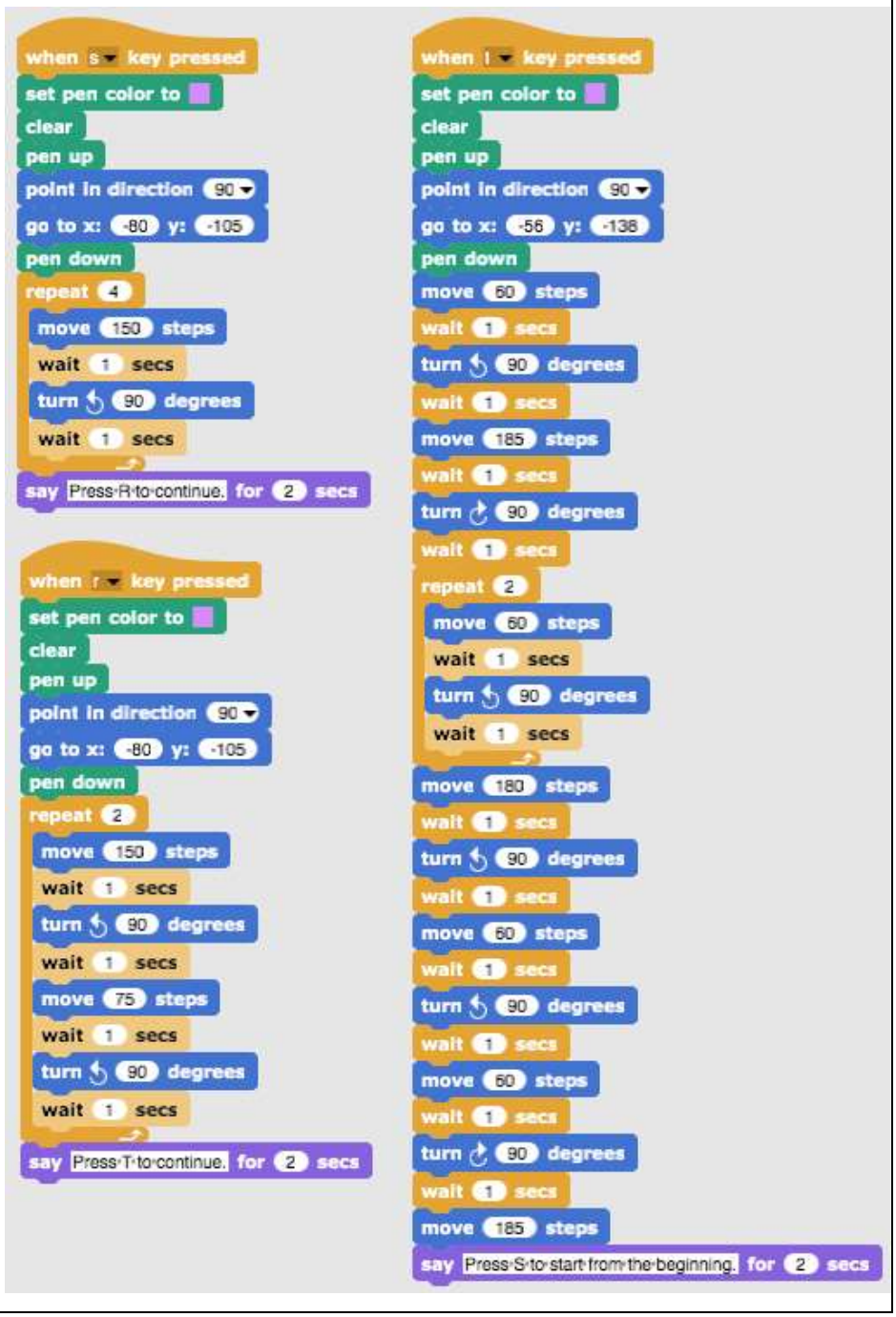
[Adım 7]

Arka planı değiştirdiğimize göre, ilk arka plana dönüp kare çizemiyoruz. O yüzden öğrenciler son bir kod ekleyecekler. [Adım 3/5] tekrarlayacaklar.

a) Arka planı değiştirmek için bir kod yazdıkları arka plana (ekranın sağ tarafındaki *board* adlı panoya) tıklarlar. Bunu S tuşuna basıldığında (*when S key pressed*) ve ardından *kostüm panosu Kare'ye geç (switch to costume boardSquare)* kullanarak yapacaklar.

b) Tebeşire ve [Adım 6]'nın altındaki koda blok eklemek için tekrar tıklarlar. Burada oyuncuya arka planı değiştirmek için ne yapması gerektiğini söylerler. Yani "S" tuşuna basın.

[Nihai Kod]



	[Ek Görevler] Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler: ● Yeni bir arka plan ekleyin ve bazı noktalar çizin. ● Noktaları birleştiren bir kod yazın. Bir arka plan çizebilir veya size verilenlerden birini kullanabilirsiniz.
Öğretmenler İçin Araçlar ve Kaynaklar	● Snap!'te yer alan tüm aktivite https://snap.berkeley.edu/project?user=mateja&project=Drawing%20with%20a%20chalk ● Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. ● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki Yarı Tamamlanmış Aktivite https://snap.berkeley.edu/project?user=mateja&project=Drawing%20with%20a%20chalk%20-%20Part ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.do

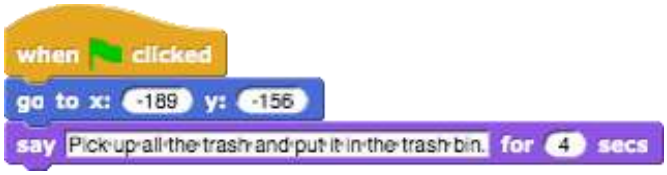
Öğrenme Senaryosu 9 - Çöpleri toplamak ve parkı temizlemek

Öğrenme Senaryosu Adı	Çöpleri toplamak ve parkı temizlemek
Geçmiş Programlama Deneyimi	Başlangıç koordinatlarını ayarlama Karakterin boyutu ayarlama Metin ekleme Ok tuşları kullanarak nesne hareket ettirme Nesnenin durumu için nesne dokunuyor (object is touching) şart ifadesini kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: • Değişkenler • Karakter göstermek ve gizlemek • Karakter kopyalamak • Kod blokları kopyalamak • Şart ifadeleri Algoritmik düşünceye odaklı özel öğrenme çıktıları. • Öğrenci toplanmış atık sayısını hesaplamak için değişken kullanır. • Öğrenci oyunun başlangıçta karakteri gösterip bir olaydan sonra onu gizler. • Öğrenci karakteri kopyalar (1 şişeden mesela 4 şişemiz olacak). • Öğrenci kod blokları kopyalayabilir (şişe karakterinden kâğıt karakterine). • Bulunan atığın sayısını ve karakterin durumunu kontrol etmek için öğrenci şart ifadeleri kullanabilir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Bahçe çöple doludur. Kız (karakter) onu temizlemeye karar verir. Kız dağınık çöpleri toplayıp çöp kutusuna atacak. **Görevler:** Öğrenciler kızın başlangıç koordinatlarını ayarlayacaklar. Oyun tüm çöp toplanınca sona erecektir. Bunu sağlamak için öğrenciler değişkenin yardımıyla çöp sayısını hesaplayacaklar (1 toplanmış atık = 1 puan). Kız çöpe dokununca onu toplayacak ve topladığı çöp yok olarak kıza bir puan kazandıracak. **Amaç:** Öğrencilerin değişkenleri kullanmaları, kod veya blok hatta tam

	bir karakter kopyalayabilmelerini sağlamak.
Faaliyetin Süresi	45 dk.
Öğrenme ve Öğretme Strateji ve Metotları	Aktif Öğrenme, Oyun Temelli Öğrenme, Problem Çözme
Öğretme Formları	Ön Öğrenme, Bireysel Çalışma
Öğretme Özeti	(Motivasyon-Giriş, uygulama, düşünme ve değerlendirme) Öğrencilere başlangıçta aşağıdakiler verilir. • Arka plan • Kız karakteri (hareket koduyla), şişe karakteri, kâğıt karakteri ve çöp kutusu karakteri Kız bahçede oynamak istemektedir. Oraya giderken bahçenin çöp ile dolu olduğunu fark eder. Bahçeyi temizleyecek, çöpleri toplayacak ve sonra da bahçede artık dinlenebilecektir.  [Adım 1] Arka plan verilir. Ayrıca kız karakterini tuşlarla hareket ettirebilmek için kod verilir. Bu kod kahverengi çizgiye dokunma koşulunu da içerir.

Öğrenciler, *x,y'ye git* (*go x, y)* bloğuna göre kız için başlangıç koordinatlarını ayarlamalıdır. Koordinatlar kendi başlarına seçilir. Sadece yolda olmaları önemlidir. Öğrenciler daha önceki aktivitelerden koordinatları nasıl ayarlayacaklarını zaten biliyorlar. Ayrıca bazı talimatlar da eklemeleri gerekir.

Örneğin:



[Adım 2]

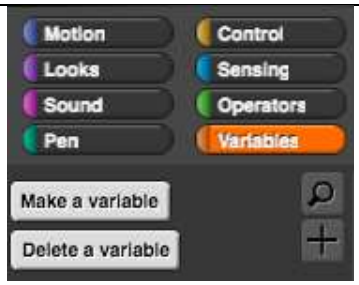
Kızın topladığı çöp sayısını saymak için değişkenleri kullanacağız.

Değişken nedir?

Değişken, bazı bilgileri sakladığımız bir kutu gibidir.

Bizim durumumuzda değişkenimizi puanlar olarak adlandırılmış bir kutu olarak görebiliriz. Kız bir çöpü aldığında çeşitli noktalarda bir çöp depolanır. Bu değişken, kızın kaç çöpü seçtiğini sayar.

Nasıl değişken yaparız?



Turuncu renkli bir Değişkenler (*Variables*) bloğunu seçiyoruz. Ardından Değişken oluştur (*Make a variable*) düğmesine tıklıyoruz. Bir Değişken adı (*Variable name*) yazıyor ve Tamam'a tıklıyoruz. Ardından bir blok noktası belirir.

Eğer kutucuk tıklanarak seçildi ise değişken ve üzerindeki değer ekranda görünür.

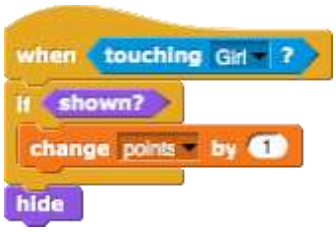
Oyunun başında alınan çöp olmadığından değişkenin değeri 0 olmalıdır. [Adım 1] kodunun altına *__yı 0 ayarla*(*set __ to 0*)' a blok seti ekler. Açılır menüye tıklayarak uygun değişkeni, yani puanları seçerler.

[Adım 3]

Öğrenciler bir şişe için bir kod yazarlar. Buradaki fikir, karakterin kıza dokunduğunda ortadan kaybolmasıdır (bu, gizlemek anlamına gelir).

Yani kod, karakter kıza dokunduğunda başlayacak. Öyleyse hangi durumda çöpü alacağını düşünmeliyiz. Çöp kutusu alındığında gizlendiğini söylersek, onu ancak = gösteriliyorsa yani hala oradaysa alabiliriz. Karakter (şişe) hala oradaysa, onu alıp "değişken kutusuna koyarız". Değişken noktalarında önceden 0 elemanımız vardı. Şimdi 1'e sahibiz. Çöpü aldığımızda değişken sayısını (puanları) 1 değiştirdiğimizi, yani 1 arttığını

görebiliriz. Çöp toplandığında çöpü gizleriz.



Şimdi kodumuzun doğru olup olmadığını test edebiliriz.

Yeşil bayrağa tıklayıp şişeyi alıyoruz. Şişe kaybolmalı ve puan sayısı 1 olmalıdır. Sonra oyunu tekrar oynamak istiyoruz ve yeşil bayrağa tekrar tıklıyoruz. Ne oluyor? Şişe şimdi nerede?

Şişe gizlendi. Daha önce saklamıştık. Yani oyunun başında şişenin gösterilmesini programlamalıyız. Bunu *göster (show)* bloğunu seçerek yapıyoruz.

[Adım 4]

Artık öğrenciler oyunlarında daha fazla şişe olmasını istiyor. Böylece karakterlerini kolayca kopyalayabilirler. Karaktere sağ tıklayıp *kopyala (duplicate*) seçerler.

Şimdi fare ile yeni şişeye tıklarlar ve onu labirentin içinde bir yere sürüklerler.

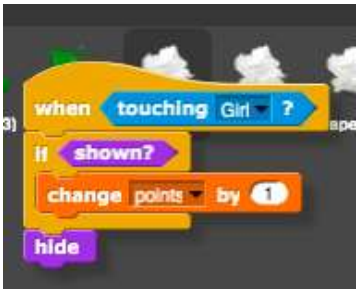
Bu adımı tekrarlayıp şişeyi tekrar kopyalayabilirler.

[Adım 5]

Şimdi öğrenciler kâğıt karakter için aynı koda sahip olmalılar. Kod

bloğuna sağ tıklayarak şişenin kodunu kopyalayabilirler.

Kağıt karakter üzerine fareyle tıklayarak onu kâğıt karakterin içine bırakın.



Yeşil bayrak tıklandığında-göster (when green flag clicked – show) kod bloğunu kopyalamak için bu adımı tekrarlarlar.

Ayrıca labirentte daha fazla kâğıt çöpü olması için [Adım 4] 'ü tekrar edebilir ve tüm kâğıt karakterleri kopyalayabilirler.

[Adım 6]

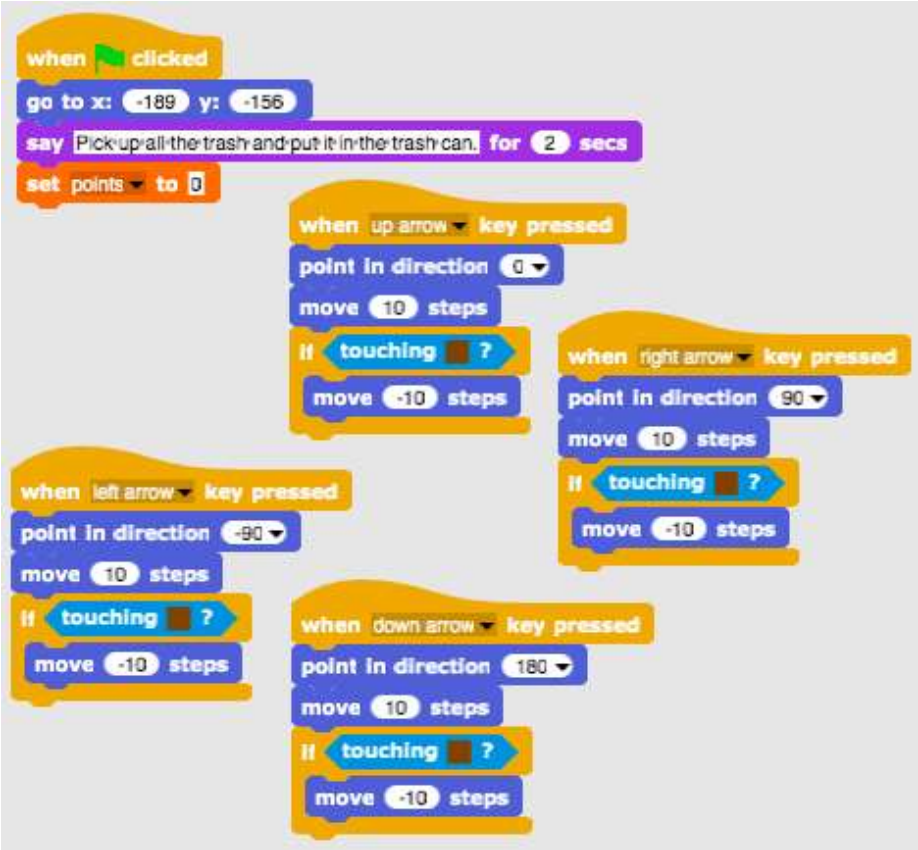
Öğrencilerin yapması gereken son şey, çöp tenekesi için bir kod yazmaktır. Çöp tenekesi karakteri zaten verilmiştir. Labirentin içinde her yere taşıyabilirler.

Ayrıca bu kod kız ona dokunduğunda aktif hale gelecektir.

Çöp kutusu, tüm çöplerin alınıp alınmadığını kontrol etmek zorunda kalacak. Değişken noktalar sayesinde bunu yapmak kolay olacaktır. Diyelim ki oyunda 8 çöp var. Bu yüzden öğrencilerin puan sayısının 8'e eşit olup olmadığını kontrol etmeleri gerekiyor. Eğer öyleyse, bu tüm çöplerin toplandığı, aksi takdirde olmadığı anlamına gelir. Bunu programlamak için if/eğer ifadesini kullanacak ve oyuncuya tüm çöpleri alıp almadığını söylemek için bazı metinler ekleyecekler.

[Nihai Kod]

Kız



Şişe/Kâğıt

Çöp Kutusu

[Ek Görevler]

Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler.

- Başka bir atık türü ekleyin (örn. Biyolojik Atık).
- Çöp kutusu, örn. "X şişe, Y kâğıt ve Z karpuz aldınız".
- Bir oyuncu tüm çöpleri alırsa çöp kutusu şunu söyler: "Tebrikler! Tüm çöpleri aldın!"
- Bir oyuncu tüm çöpleri toplamazsa, çöp kutusu ona hangi çöpün alınmadığını söyleyebilir. Örn. “Tüm şişeleri almadın. Bütün karpuzları toplamadın ve "Tüm çöpleri aldığınızda geri gelin".

Öğretmenler İçin Araçlar ve Kaynaklar	● Snap!’teki tüm aktivite https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park

	● Snap!'teki ek aktivite (çözümlü) https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park%20%2B%20Add.%20Task ● Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. ● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki yarı tamamlanmış aktivite https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park%20-%20Part ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

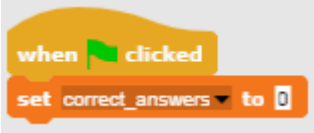
Öğrenme Senaryosu 10 - Kedileri Beslemek

Öğrenme Senaryosu Adı	Kedileri Beslemek
Geçmiş Programlama Deneyimi	● Koşullu ifadeler (if, if-başka blokları) ● Metni yazdırma (söyle bloğu)
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● değişken değerini ayarlama ve artırma ● döngünün içine / dışına değişken değer atama ● döngü ● rastgele sayılar ● dizi birleştirme ● operatörler: mantıksal, aritmetik, ● giriş Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, n kez tekrarlama döngüsünü kullanma durumunu tanır. ● Öğrenci, döngünün her yinelemesinde ve döngüden önce bir kez değer atama arasında ayrım yapar. ● Öğrenci, bir oyuncudan numara almak için giriş bloğunu kullanır. ● Öğrenci doğru cevabı üretmek için aritmetik operatörleri nasıl

	kullanacağını bilir. ● Öğrenci, oyuncu girdisinin doğruluğunu kontrol etmek için if - else cümlesini kullanır ve uygun bir yanıt verir. ● Öğrenci doğru cevapları saymak için bir değişkeni nasıl kullanacağını bilir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	***Kısa Açıklama:*** Oyuncunun on adet çarpma işlemi yapması ve doğru cevapları sayması gereken bir oyun programlayın. ***Görev:*** Barınak bekçisi Martha'nın oyuncudan belirli bir odada besleyebileceği kedi sayısını birkaç defa soracağı etkinliği programlayın. Sayı, kâselerin sayısına ve boyutuna bağlıdır. Her oda için bu iki numaranın rastgele atanması gerekir. Ayrıca doğru cevapları sayacak bir sayacımız da olmalı. İlk sığınak bekçisi, oyuncuya görevi açıklamak zorundadır. Ardından oyun başlar. Kedi sayısını 10 defa sorduğunda oyun biter. Girilen numaranın doğru olup olmadığı her seferinde yanıt vermelidir. Aktiviteden sonra oyuncunun ne kadar başarılı olduğunu özetlemesi gerekir. Oyuncunun kaç kez doğru cevap verdiğini ve kaç kez yanıldığını söyler. **Öğrencilere, bir döngü içinde çok değişkenli rastgele değer ataması kavramı ve bir döngü dışında yaptığımızdan nasıl farklı olduğu anlatılacaktır. Ayrıca doğru oyuncu girdilerinin nasıl alınacağını, test edileceğini ve sayılacağını da öğrenecekler.**
Faaliyetin Süresi	45 Dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme
Öğretme Formları	Ön öğrenme Bireysel çalışma/İkili çalışma/grup çalışması

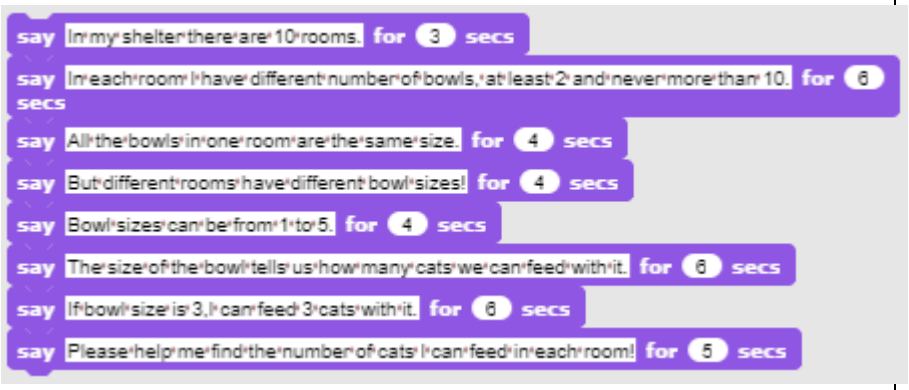
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Barınak bekçisi on farklı odada kedilerini beslemeye çalışıyor. Her odada rastgele sayıda (2'den 10'a kadar) farklı boyutlarda (1'den 5'e kadar) kâse bulunur, ancak her odada tüm kâseler aynı boyuttadır. Kâsenin boyutu, ondan kaç kedinin yiyebileceğini söyler. Örneğin kâse boyutu 3 ise, bu 3 kedinin ondan yiyebileceği anlamına gelir. Her odada besleyebileceği kedi sayısını bulmaya yardım edin. [Adım 1] İlk önce öğrencilere oyun için ilginç bir arka plan tasarlamaları talimatını veriyoruz. Zaman kazanmak istiyorsak, onlara biz verebiliriz.  [Adım 2] Varsayılan kaplumbağa karakteri için kedi barınağı koruyucusunu temsil edecek yeni bir kostüm seçmemiz gerekiyor.  [Adım 3] Gerekli değerleri saklamak için üç değişkene ihtiyacımız var. **1)** Doğru cevapların sayısını saklamak için **2**) Her odadaki kâse sayısı için rastgele değer atamak için (2-10) **3**) Kase kapasitesine (1-5) rastgele değer atamak için.

Doğru cevap sayacı 0 olarak ayarlanmalıdır. Diğer ikisinin döngüden önce ayarlanması gerekmez. Çünkü döngünün her yinelemesinde bunlara yeni rastgele değerler atayacağız. Odaları da saymak istiyoruz. Ancak saymak için özel bir değişkene ihtiyacımız yok. *For döngüsü (for loop)* ile aynı değişkeni kullanacağız. Numarası 1 değerine başlatılacak ve ardından 10 değerine ulaşılana kadar her yineleme için 1 artırılacaktır. Bu, oda sayımını kopyalar.



[Adım 4]

Daha sonra oyuncu için talimatları programlamalıyız. Bunu *Looks / say [string (Görünümler/söyle ..[string]) ve wait [n] seconds ([n] saniye bekle)* blokları kullanarak yaparız.



[Adım 5]

Öğrencilerle her odada olacak ve dolayısıyla aynı olacak eylemlerin neler olduğunu tartışıyoruz. Bunlar, döngünün her yinelemesinde

yürütülecek döngü bloğunun içine yerleştirilmesi gereken komutlardır.

Öncelikle, o odadaki kâse sayısı ve kâse boyutu (1-5) için rastgele bir değer (1-10) atamamız gerekecek. Sonra bir oyuncuya o odada kaç tane kedi besleyebileceğimizi sormamız gerekecek. Cevabın doğruluk için test edilmesi gerekecek. Uygun bir yanıt vermemiz ve doğru olup olmadığını hatırlamamız gerekecek (doğru cevap sayacı). Her yinelemenin sonunda oda numarasını da 1 artırmamız gerekecek.

[Adım 6]

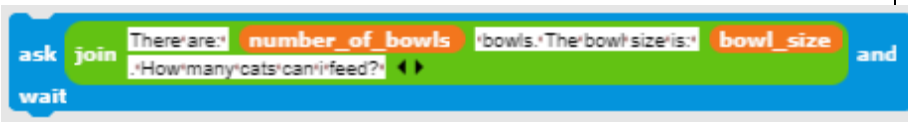
Kâselerin sayısı ve büyüklükleri için rastgele değerler atamak için, *Değişkenler / set [seçenekler] (Variables/set [options])* değeri ile *Operatörler / rastgele[n] ile [m] seç (Operators/pick random [n] to [m])* kullanacağız.

[Adım 7]

Oyuncuya *Algılama / [dizi] (Sensing/ask [string])* ve *bekleme bloğu (wait block)* içinde besleyebileceğimiz kedi sayısını sormak istiyoruz. Aksi takdirde belirli bir saniye boyunca görüntülenecek ve ardından yeni bir metin satırıyla güncellenecektir. Bu şekilde oyuncular mevcut odada kaç kâse / beden olduğunu kolayca unutabilirler. Bir metin kombinasyonundan ve değişkenlere referanslardan oluşacak bir dizge oluşturmak için Operatörler/birleştirme [dize1] [dize2] *(Operators/join [string1][string2])* bloğu kullanıyoruz. Tüm cümleye uyması için bu bloğu genişletmemiz gerekecek.

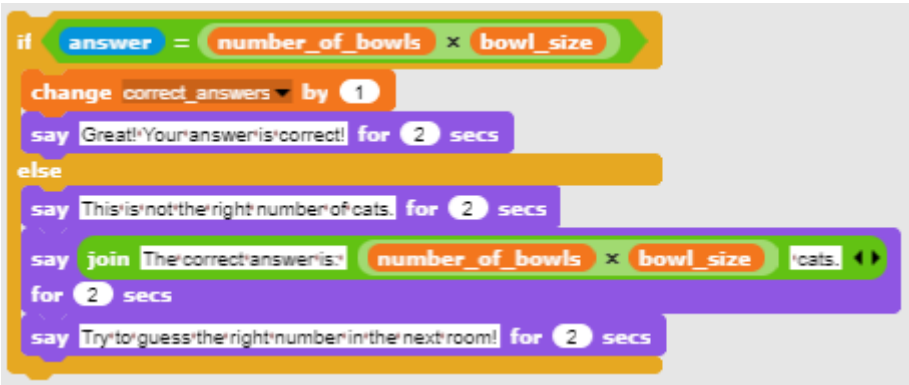
[Adım 8]

Bu uzun dizgiyi *Sense/Ask [string]* içine koymalı ve oyuncudan cevabı alabilmek için blok beklemeliyiz.



[Adım 9]

Oyuncu cevapladığında doğruluğunu kontrol etmeliyiz. Yalnızca iki olası durum vardır, oyuncu doğru veya yanlış olabilir, bu yüzden If-Else bloğunu kullanacağız. Doğru cevap, kâse sayısı ile kâse büyüklüğünü çarpmanın değeridir. Oyuncunun cevabının bu sayıya eşit olup olmadığını kontrol etmeliyiz. Cevap doğruysa, doğru cevap sayacını 1 artırıp yanıt veriyoruz. Değilse, sadece yanıt veriyoruz. Doğru cevap sayacından hesaplayabildiğimiz için yanlış cevapları saymak zorunda değiliz.

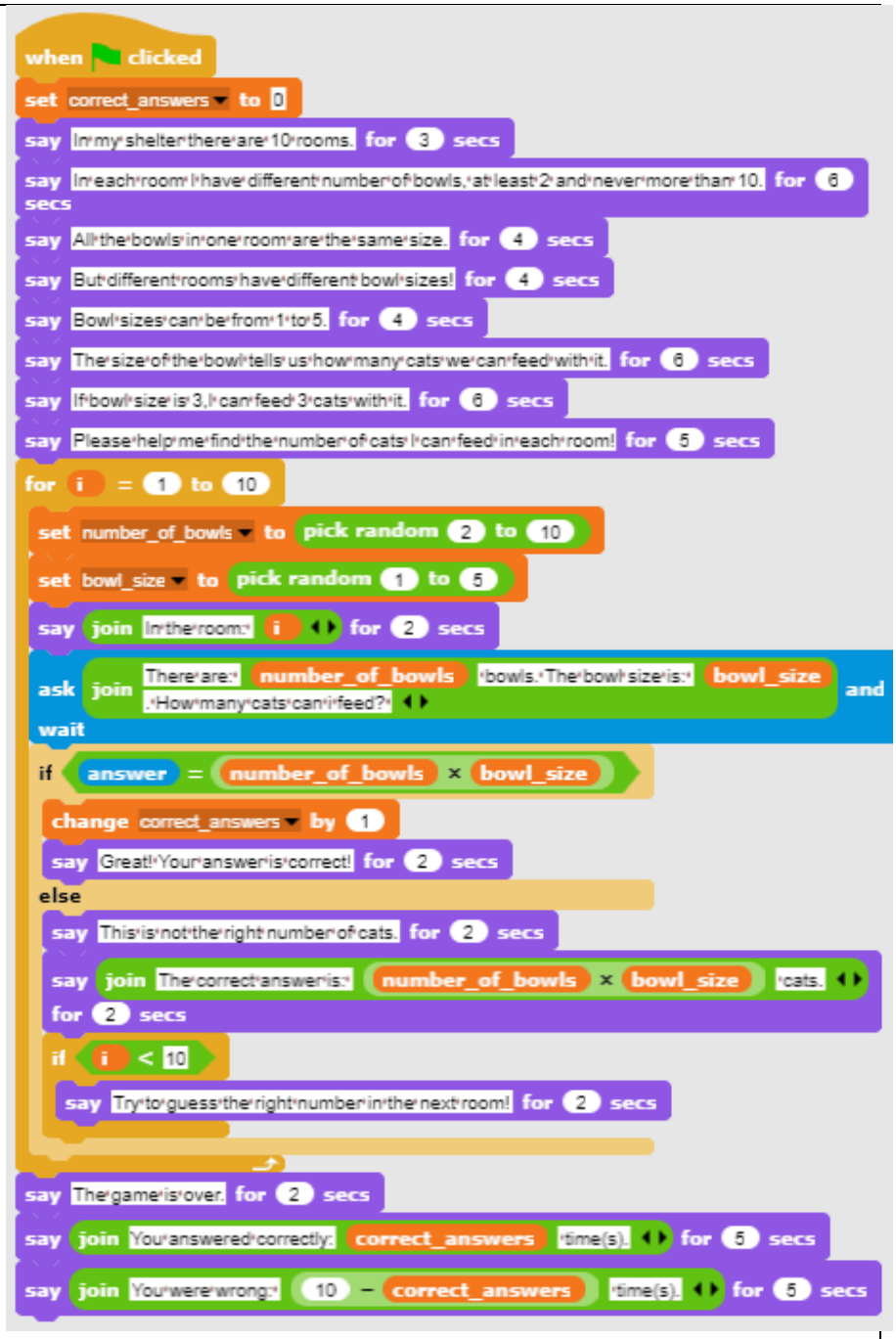


[Adım 11]

Şimdi bir döngü seçmemiz gerekiyor. Daha önce de belirtildiği gibi, döngüye göre seçmek en mantıklısıdır çünkü yineleme için kullanılan değişken odaların sayımını kopyalar.

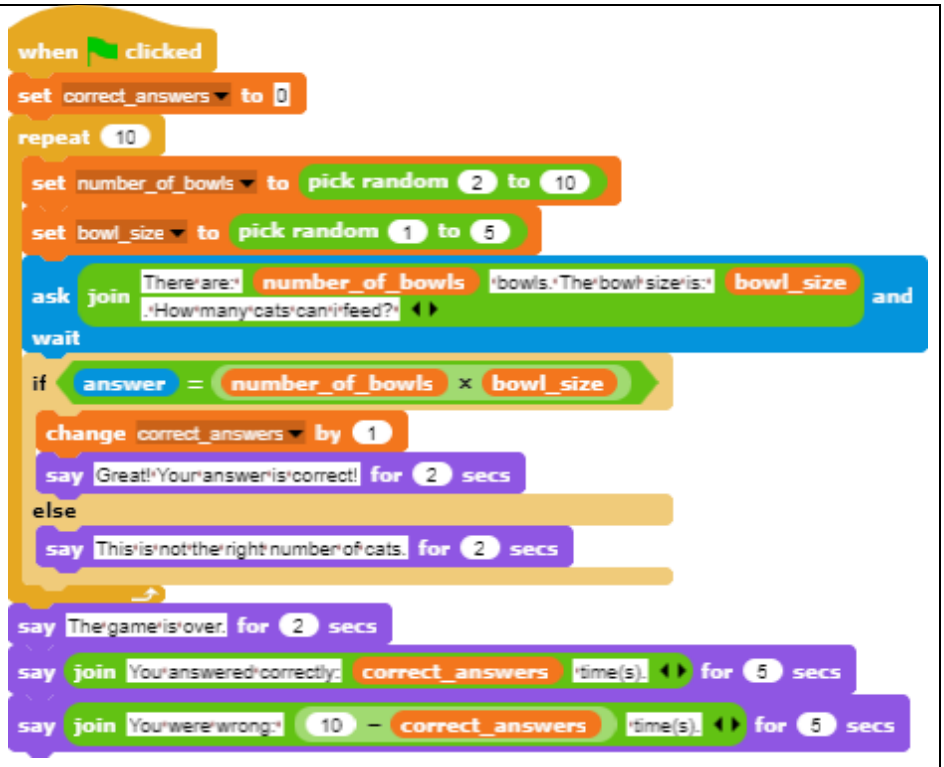
[Adım 12]

	Döngü durduğunda oyun biter. Oyuncu başarısı hakkında bilgi vermemiz gerekmektedir. Doğru cevap sayısı, doğru cevap sayacında saklanır. Yanlış cevapların sayısı hesaplanabilir. [Nihai Kod]



[Etkinliğin temel versiyonu]

Zaman kazanmak için senaryonun temel versiyonunu kullanabiliriz. Temel bir versiyonda tüm temel kavramlar dâhil edilmiştir. Yukarıda açıklanan diğer işlevler daha sonraki yükseltmeler olarak kullanılabilir.

	
Öğretmenler İçin Araçlar ve Kaynaklar	• Snapteki tüm aktivite https://snap.berkeley.edu/project?user=zapusek&project=cat_feeding_2 • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. • Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	• Snap!'teki şablon https://snap.berkeley.edu/project?user=zapusek&project=cat_feeding_template • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 11 - Bir barınaktaki kedi sayısını tahmin etmek

Öğrenme Senaryosu Adı	Bir barınaktaki kedi sayısını tahmin etmek
Geçmiş Programlama Deneyimi	● koşullu ifadeler (Eğer bloğu (if block)) ● metni yazdırma (Söyle Bloğu Say Block))
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● rastgele değerler ● değişken atama ● kullanıcı girişi ● döngüye kadar tekrarlayın ● karşılaştırma operatörleri ● sayaç Algoritmik düşünceye odaklı özel öğrenme çıktıları. ● Öğrenci değişkene rastgele değer atar. ● Öğrenci, bir oyuncudan numara almak için giriş bloğunu kullanır. ● Öğrenci, oyuncudan tekrar tekrar sayıyı girmesini ve bir değer testi gerçekleştirmesini istemek için döngüye kadar tekrarla kullanır. ● Öğrenci if cümle ve karşılaştırma operatörleriyle değer testi yapar ve uygun yanıt verir. ● Öğrenci oyunun bitip bitmediğini test etmek için tekrar döngüsünün koşulunu belirler. ● Öğrenci, dolaylı olarak koşullara bağlı olduğu için oyunun bittiğinde test etmesi gerekmediğini fark eder. ● Öğrenci, oyuncu tahminlerini saymak için bir sayaç uygular ve her iki olası sonucu ayırt etmek için nihai değeri kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	***Kısa açıklama:*** Başlangıçta 1'den 100'e kadar rastgele bir sayının bir değişkene rastgele atanacağı basit bir oyun programlayın. Oyuncu sayıları yazarak tahmin etmeye çalışacaktır. Girdi numarası: rastgele değerden daha fazla, daha az veya eşit ise bir yanıt alacaktır. ***Görev:*** Kedi barınağı Martha'yı rastgele olarak kedi sayısını belirlemesi, oyuncudan ismini sorması ve ardından görevi ona açıklaması için programlayın. Daha sonra Martha, oyuncuyu ismiyle

	selamlamak ve ardından tekrar tekrar bir numara sormak zorundadır. Oyuncu tahminini girdiğinde şu şekilde yanıt vermelidir: **1)** Giriş sayısı gerçek sayıdan düşükse, "kedi sayısı daha fazla" **2)** Giriş sayısı gerçek sayıdan yüksekse,: "Kedi sayısı daha az" 3) Giriş numarası doğruysa, "Mükemmel, doğru sayıyı tahmin ettiniz". Her oyuncunun denemesini sayacak bir sayaç programlayın. Oyuncu doğru sayıyı tahmin ettiğinde deneme sayısının 5'ten az olup olmadığını kontrol etmelisiniz. Bu durumda oyuncu kediyi alır. Aksi halde almaz. **Amaç: Öğrencilere döngüye kadar tekrar etmeleri ve oyunu durduran koşulu örtük olarak izlemek için koşulun nasıl ayarlanacağı tanıtılacaktır. Ayrıca değişkenleri farklı durumlarda nasıl kullanacaklarını da öğreneceklerdir. Örneğin, rastgele bir değer ayarlamak, sayaç olarak veya oyuncuların girdisini almak gibi.**
Faaliyetin Süresi	45 dk.
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme
Öğretme Formları	Ön öğrenme Bireysel çalışma/İkili çalışma/grup çalışması
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Kedi barınağı bekçisi Martha, barınağında bulunan kedilerin tam sayısını tahmin etmenizi istiyor. Sayı 1 ile 100 arasında herhangi bir yerde olabilir. Oyuncu sayıyı yazdığında, mevcut giriş sayısı doğru sayıda kedi sayısından az, daha fazla veya ona eşitse yanıtlar. Bir oyuncu beş denemeden daha az denemede kedi sayısını tahmin ederse, kediyi alır, aksi halde tekrar oynaması istenir.

[Adım 1]

İlk görev, oyun için ilginç bir arka plan oluşturmaktır. Öğrenciler kendileri çizebilir veya internetten ücretsiz lisanslı görüntüleri kullanabilir. Zaman kazanmak için arka planı önceden hazırlayabiliriz.

[Adım 2]

Varsayılan kaplumbağa karakteri için kedi barınağı koruyucusunu temsil edecek yeni bir kostüm seçmemiz gerekiyor.

[Adım 3]

Kedi sayısı rastgele atanmışsa, bu oyunun bir kereden fazla oynanmasının ilginç olabileceğini öğrenciler ile tartışıyoruz. Bu rasgele sayıyı tahmin etmek için *sayı karşılaştırma tahminin (guess numbers comparisons)* kullanılabilmesi için, onu bir değişkende de saklamamız gerekir. Değişkenler artık (listelerin kavramını henüz bilmediklerini varsayıyoruz) Snap'te belirli bir değeri hatırlamanın tek yoludur. Bu, program başladığında gerçekleşmelidir (Olay/Yeşil bayrak tıklandığında (Event/When green flag is clicked) komutlarını

hatırlayın).

[Adım 4]

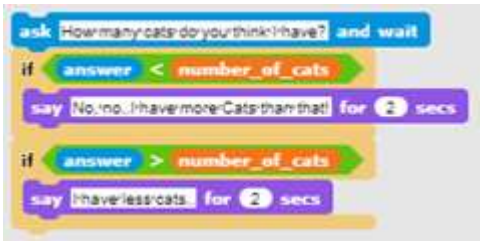
Barınak bekçisi, oyuncudan onu selamlamak için adını sorar. Bu, *Sense / ask [string]* ve *wait block* ile yapılır. Oyuncu cevabı, *answer* adlı yerleşik bir değişkende saklanır. Onu selamlamak için, değişken cevapta saklanan dizeyi bir miktar selamlamayla birleştirmeliyiz. Bu *Operatörler (Operators)/join [string1] [string2]* bloğu ile yapılır. Metni görüntülemek için, *n saniye için Looks /say [string] (Looks/say [string] for n seconds)* bloğunu kullanırız. Oyunu oynamak için talimatlar yazmak için de bu blokları kullanıyoruz. Metnin gösterim süresine dikkat etmenin önemli olduğunu da vurgulayabiliriz.

[Adım 5]

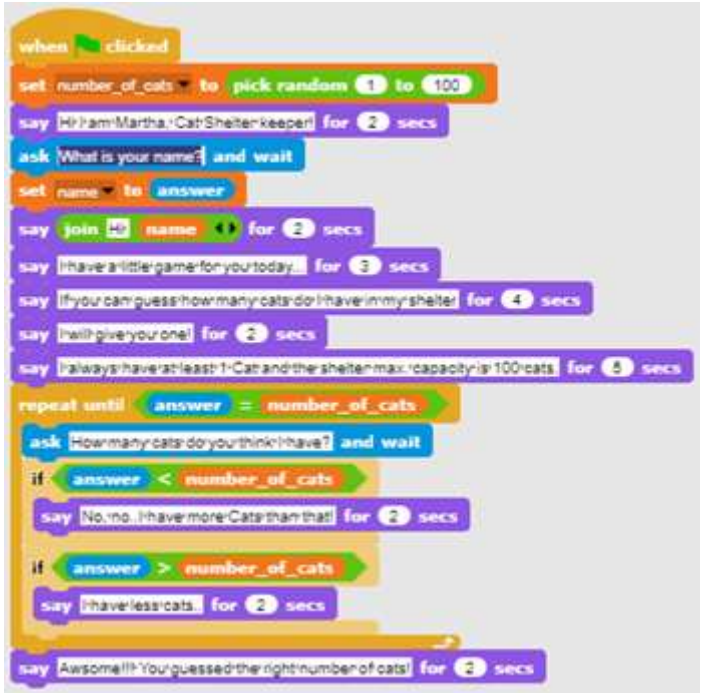
Öğrencilerle, doğru sayıyı bulmak için oyuncuların kaç kez tahmin etmek zorunda kalacağını tahmin etmenin mümkün olmadığını tartışıyoruz.

Çok şanslı ise ilk denemesinde tahmin edebilir, belki 5 belki çok daha fazla tahminde bulunmak zorunda kalabilir. Bunu hiç bir zaman bilemeyiz. Verilen görev için doğru döngüyü seçmemizin nedeni budur. Barınak bekçisi, oyuncu doğru sayıyı tahmin edene kadar tekrar tekrar bir numara istemeli ve uygun bir yanıt vermelidir.

İstenilen yürütmeyi gerçekleştirebileceğimiz tek döngü

	until[condition] (__e kadar [koşul]) döngüsüdür. Oyuncu doğru yanıt verinceye kadar döngüyü devam ettirmeliyiz. Yani değişkende bulunan cevap ile cat_number (kedi sayısı) değişkeni içindeki sayı aynı oluncaya kadar.  [Adım 6] Sonra, öğrencilere döngü gövdesine (loop body) girecek komutların ne olduğunu sormalıyız. Oyuncu doğru sayıyı tahmin edene kadar tekrarlanacak etkinlik veya komutlar nedir? İlk olarak, oyuncudan bir sayı girmesini istemeliyiz, sonra bu sayının değerine göre bir yanıt vermeliyiz.  [Adım 7] Öğrencilerle açıklanması veya tartışılması gereken son şey, bu döngünün ne zaman biteceği ve bunun ne anlama geldiğidir. Oyuncu cevapları kedi sayısına eşit olduğunda, döngü gövdesindeki her iki koşul da yanlış olacaktır. Bu nedenle döngü, döngü koşulunu kontrol ederek bir sonraki yinelemeye gidecektir. Bu sefer koşul doğru olacak, böylece döngü sona erecek ve döngüyü izleyen komutlar çalıştırılacak. Başka bir deyişle, döngü sona erdiğinde oyuncunun sayıyı doğru

tahmin ettiğini biliyoruz. Şimdi buna göre cevap verebiliriz.



[Adım 9]

Sayaç rolüne sahip olacak yeni bir değişken oluşturmalı ve onu 0 değerinden başlatmalıyız. Değişken başlatmanın önemini ve değeri belirlemek ile onu artırmak arasındaki farkı öğrencilerle tartışıyoruz. Bir değişkenin değerini belirlediğimizde, önceki değer kaybolur. Bu bir sayaç için uygun değil. Değişken değerini bir sayı kadar artırırsak, o sayıyı daha önce değişken olan değer ne olursa olsun ekleriz. Bu durumda tam olarak istediğimiz şey budur. Bir oyuncu her yeni sayı girdiğinde, onu 1 artırmak istiyoruz.

[Adım 10]

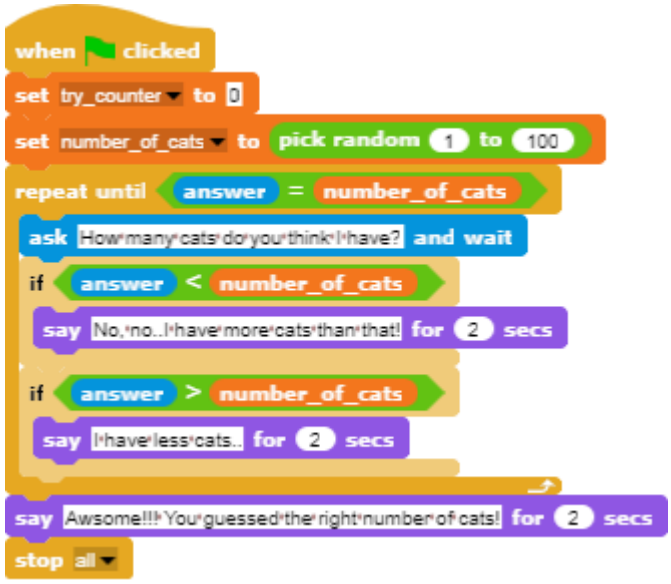
Doğru cevabın ardından, oyuncunun kediyi alıp almayacağına karar vermek için sayaç değişkeninin değerini kontrol etmeliyiz. Snap'in

mantıksal operatörlerden yalnızca *“daha az (<)”*a sahip olduğundan yani *“daha az veya eşit”* operatörlere sahip olmadığından, bir oyuncunun kediyi alıp alamayacağına karar verebilmemiz için *cat_counter <6* ‘yı kullanabiliriz. Bu aynı zamanda If-Else koşul bloğunu kullanmak için iyi bir örnektir çünkü aslında biz iki durum arasında bir ayrım yapmış olduk.

[Nihai Kod]

[Aktivitelerin Temel Versiyonu]

Zaman kazanmak için senaryonun temel versiyonunu kullanabiliriz.

	Temel bir versiyonda tüm temel kavramlar dâhil edilmiştir. Yukarıda açıklanan diğer işlevler daha sonraki yükseltmeler olarak kullanılabilir. 
Öğretmenler İçin Araçlar ve Kaynaklar	• Snap! te yer alan tüm aktivite https://snap.berkeley.edu/project?user=zapusek&project=cats_in_a_shelter • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. • Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	• Snap!'teki şablon https://snap.berkeley.edu/project?user=zapusek&project=cats_in_a_shelter_template • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

İleri Seviye Öğrenme Senaryoları

Öğrenme Senaryosu 12 - Sağlıklı yiyecekler yakalamak

Öğrenme Senaryosu Adı	Sağlıklı yiyecekler yakalamak
Geçmiş Programlama Deneyimi	Karakter için test ekleme Karakter gösterme ve gizleme Bir yönü göster kullanma Rastgele özelliğini kullanma Noktaları saymak için değişkenleri kullanma Döngü tekrarını kullanma Sonsuza kadar döngü kullanma Koşul ifadeleri kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullu ifadeler ● Döngü ● Bir yönü göster ● Rastgele (random) Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, oyunun kız konuşmayı bitirmeden başlamasını önlemek için değişken kullanır (isteğe bağlı). ● Öğrenci, yiyeceğin hareket etmeye başlayıp başlayamayacağını kontrol etmek için (bir değişken yardımıyla) if/eğer ifadesini kullanır. ● Öğrenci, puanlar 5'ten az olana kadar yiyeceğin hareketi için döngü tekrarını kullanır. ● Öğrenci aşağı hareket eden karakterler için 180 (aşağı) yönünü kullanır. ● Öğrenci, adım sayısını seçmek için rastgele kullanır.

	● Öğrenci rastgele pozisyona geçmek için rastgele kullanır. ● Öğrenci x (rastgele), y (sabit) konuma (isteğe bağlı) geçmek için rastgele kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	***Kısa açıklama:*** Kız yiyecek yakalayacak ama dikkatli olmalı, sadece sağlıklı yiyecekler puan kazandırır! ***Görevler:*** Öğrenciler iki farklı karakter programlamak zorundadır. Talimatlar veren, oyuna başlamak için ne yapması gerektiğini söyleyen ve puanları sayan bir kız ve ekranın üstünden rastgele düşen yiyecekler. Ek olarak, öğrenciler bir kız konuşmayı bırakmadan önce yemek hareketini önlemek için bir değişken ve if/eğer ifadesi ekleyebilirler. **Amaç: Öğrenciler, X Adımlar için rastgele hareket etmeyi ve bir konum seçmeyi ve ayrıca diğer olayları önlemek için değişkenleri ve koşulluları nasıl kullanacaklarını öğrenecekler.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma

Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Kız yiyecek yakalıyor. Her sağlıklı besin 1 puan kazandırırken sağlıksız her besin 1 puan kaybettir. Oyun bir kız tarafından verilen bazı talimatlarla başlar. Sonra kız kaybolur ve yiyecek belirir. Oyuncu 5 puan topladığında yiyecekler kaybolur ve bir kız tekrar ortaya çıkar.  [Adım 1] Bu aktivite bireysel çalışma veya ikili çalışma olarak planlanmıştır. Öğretmen birkaç ipucu verip zor kısımları açıklar ve gereken durumlarda yardım eder. Öğrencilere başlangıçta ● Arka plan ● Kız karakteri verilir. Öğrenciler arka planı seçer ve bir ana karakter ekler. Örneğin, bir kız. Kız başta bazı talimatlar verir ve sonra saklanır. Önceki aktivitelerden gördüğümüz gibi *Bayrak Tıklandığında Göster (show when the flag is clicked)* bloğu eklemek iyi olacaktır (tekrar oynarken, hareketli grafik gizli kalırsa). Aşağıdaki gibi bir kod olabilir.

Bu karaktere daha sonra döneceğiz. Şimdi bir meyve için bir kod yazalım.

[Adım 2]

Öğrenciler yeni bir karakter (sağlıklı bir yiyecek) ekler. Örneğin, bir elma. İlk olarak yukarıdan aşağıya doğru bir karakter hareketi programlarlar. Bu nedenle aşağıdaki blokları seçerler.

Elmalarının baş aşağı olmasını istemiyorlarsa, üçüncü seçeneği yani *döndürme (don't rotate)* seçeneğini seçebilirler.

Bir oyunu daha ilginç kılmak için adım sayısı rastgele seçilebilir. Böylece hız her zaman aynı olmayacaktır. Örneğin,

Sıradaki Adım, elma ekranın altına geldiğinde ne olacağını düşünmek. Bu durumda öğrenciler *if ifadesinin* (*if statement*) kombinasyonunda bir *dokunma kenarı (touching edge)* bloğu kullanabilirler. Elma kenara temas ederse, rastgele bir konuma hareket ettirilecektir. Hareket blokları bize bir sonraki bloğu sunuyor.

Bu komut rastgele olarak x herhangi bir y koordinatını seçer ve elma ekranın herhangi bir yerinde görünebilir (resimdeki kırmızı noktalara bakın).

Elmanın her zaman ekranın üst kısmında görünmesini istiyorsak y değeri sabitlenebilir ve sadece x değeri rastgele seçilir. Aşağıdaki kodla elma her zaman ekranın üst kısmında görünecektir (resimdeki kırmızı noktalara bakın).

[Adım 3]

Öğrenciler artık saymak için kullanacakları değişken *puanlar (points)* oluşturabilirler. Puanlar başlangıçta 0 olarak ayarlanmalıdır (kız karakterinin üzerinde).

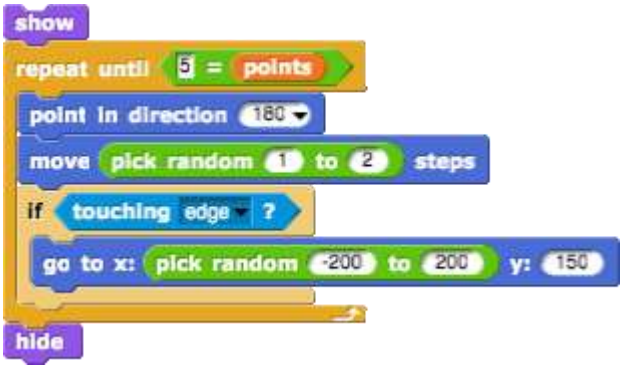
[Adım 4]

Elmanın tekrar tekrar hareket etmesini istiyorsak, bir döngüye ihtiyacımız var. Öğrenciler bir koşul belirleyene kadar döngü tekrarını kullanabilir. Örneğin, 5 puana ulaştıklarında oyunun bitmesini isteyebilirler. Yani koşul puan = 5 (*points* = 5) olacak ve koşul yanlış olana kadar döngü tekrarlanacaktır. Koşul doğru olduğunda, bu oyuncu 5 puana ulaşır, döngü durur.

[Adım 5]

Başta elmanın gösterilmesini istemiyoruz ama kız ona talimat verdikten sonra gösterilebilir. Öğrenciler elmayı tuşa basıldığında gösterecek şekilde programlayabilirler. Tabii ki, döngü tekrarından

önce bir blok gösterisi eklemeleri ve ondan sonra gizlemeleri gerekiyordu. Şimdilik kodun tamamı şöyle görünüyor:



[Adım 6]

Elmaya tıklandığında (veya fare-enter yapıldığında) ne olur?

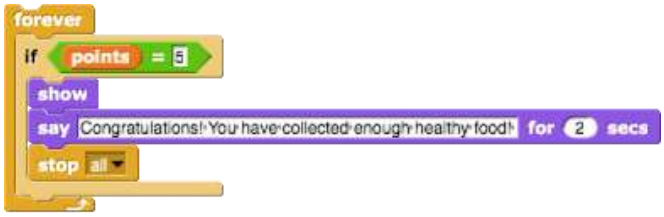
Elma saklanmalı, puan sayılmalı, pozisyon değiştirmeli ve tekrar göstermelidir. Puanlar 1 ile değiştirilecek ve pozisyon için öğrenciler öncekiyle aynı kodu kullanabilirler.

[Adım 7]

Şimdi kıza geri dönelim.

Kız yeniden ortaya çıkmalı ve şöyle demeli: Tebrikler!

Sonsuz döngüye ihtiyacımız olacak, bu da 5 puana ulaşıp ulaşmadığımızı kontrol etmeye yarayacak. Eğer 5 puana ulaşırsak, kız ortaya çıkacak ve bir şeyler söyleyecek. Bundan sonra *hepsini durdur* (*stop all*) bloğu ekleyeceğiz. Öğrencilerin bu durmanın ne anlama geldiğini çözmelerine izin verin (kız durmaksızın sonsuza dek "Tebrikler ..." diyecek).



[Adım 8]

Oyunu tekrar oynarken, öğrenciler tüm talimatları ([Adım 1] 'den itibaren) zaten bildikleri için bu kısmı atlamak isteyeceklerdir. Oyunun başlaması için önceden "S" ye basabilirler. Ancak kız hala konuşmaya devam edecek.

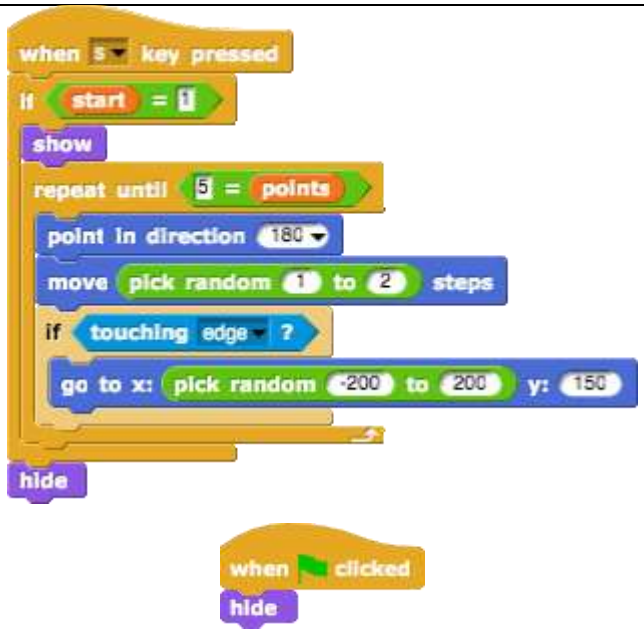
Bunu önlemek için, başlangıçta 0 olarak ayarlanması gereken başka bir değişken (*başlangıç (start)* adında) oluşturabiliriz. Ardından, kızın talimatlarından sonra, başlangıç değişkeni 1 olarak değişecektir.

.

Şimdi elmayı sadece başlangıç değişkeni 1'e eşitse başlayacak şekilde programlamalıyız ki öğrenciler bunu if ifadesiyle yapacaklar. Bununla, kız konuşmayı bırakmadan öğrenciler oyun oynayamayacak.

Oyunu tekrar oynadığımızda başka bir şey daha olabilir. Örneğin 3 puanımız varken oyunu durdurursak elma kaybolmayacaktır. Bu durumda oyuna yeniden başlarken, kız talimat vererek bitmeden elma görülecektir. Bunu istemediğimiz için oyunun başlangıcında elmanın gizlendiği bir kodu ekliyoruz.

Şimdi Elma kodu;



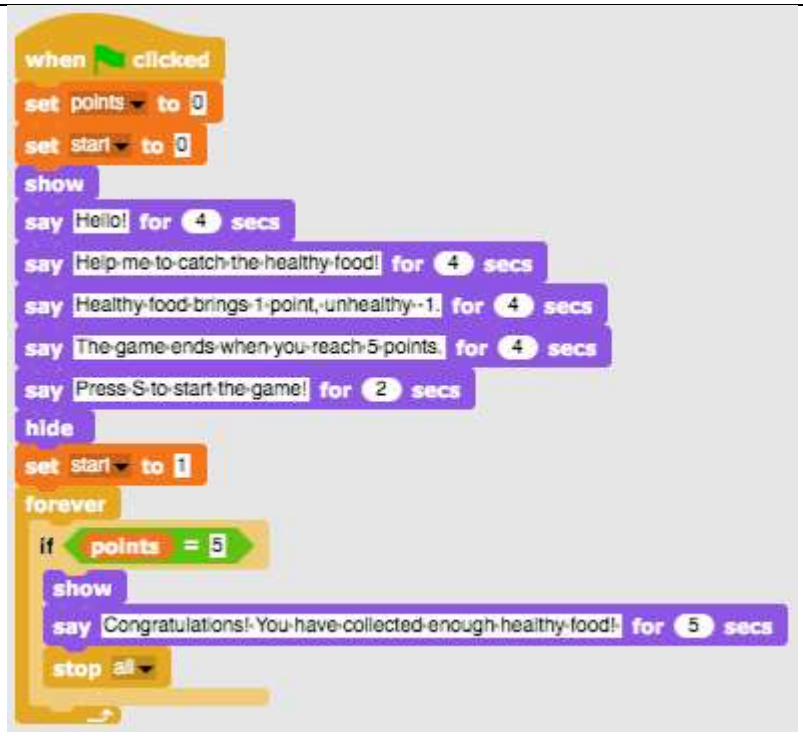
[Adım 9]

Öğrenciler artık elma karakterini birçok kez kopyalayabilir ve kostümü değiştirebilirler (eğer isterlerse). Kod aynı olacak.

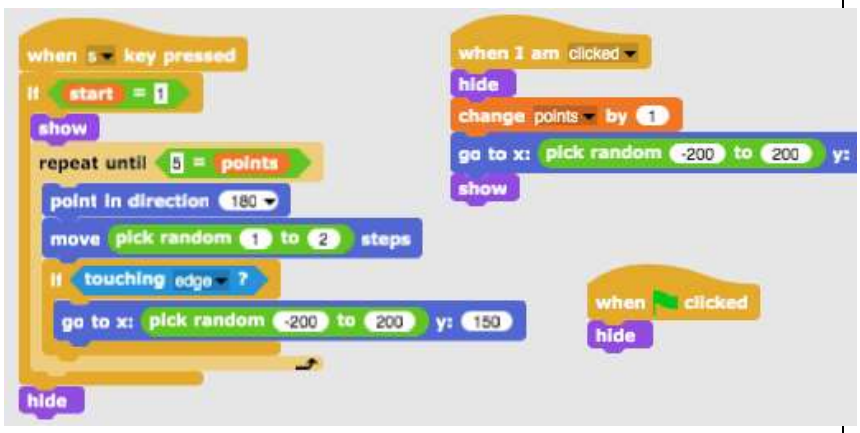
Tek değişiklik, sağlıksız yiyeceklerde tıklandığında 1 puan kaybedecekler.

[Nihai Kod]

Kız



Elma



[Ek Görev]

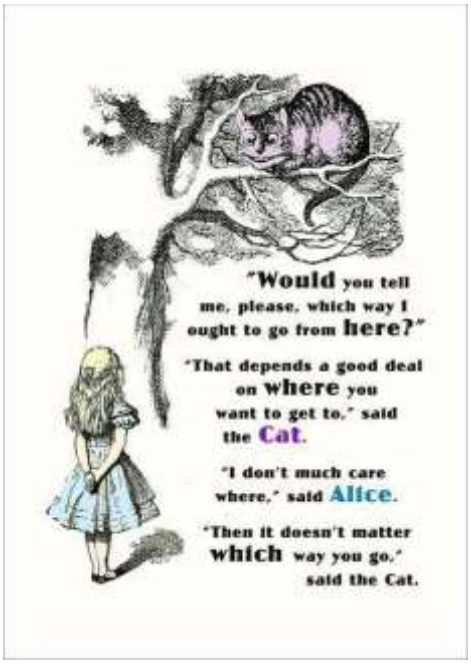
Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler.

- Oyunu bir kâse karakterinin yiyeceği yakalayacağı şekilde değiştirin.
- Yeni bir karakter (kâse) ekleyin. Kendiniz çizin ya da çevrim içi bulun.
- Kâsenin başlangıç konumunu ayarlayın (örneğin ekranın

	altında). Kâsenin hareketi için bir kod yazın (yukarı ve aşağı da isterseniz sol ve sağ). Yiyecek karakterleri, kâseye dokunarak rastgele bir yerde kaybolmalı ve yeniden ortaya çıkmalıdır (ve daha önce olduğu gibi yiyeceğe fare tıklandığında değil). ● Kuralları değiştirin - bir oyuncu 20 puan aldığında (kazandığında) veya 3 sağlıksız yiyecek aldığında (kaybettiğinde) oyunun bitmesine izin verin. ● Oyunu daha ilginç hale getirmek için daha fazla yiyecek karakterleri ekleyin. ● Bir oyuncu 5, 10, 15 puan skor elde ettiğinde kâse kostümünü değiştirin.
Öğretmenler İçin Araçlar ve Kaynaklar	● Snap!'teki Tüm Aktivite https://snap.berkeley.edu/project?user=mateja&project=Catching%20healthy%20food ● Snap!'teki Çözümlü Ek Görevler: https://snap.berkeley.edu/project?user=mateja&project=Catching%20healthy%20food%20%2B%20Add.%20Task ● Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. ● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki Yarı Tamamlanmış Aktivite https://snap.berkeley.edu/project?user=mateja&project=C4G12_Catching%20healthy%20food%20-%20Part ● Öğrenci İçin Talimatlar (C4G2_InstructionsForStudent.docx) ● Images: bowl1.png, bowl2.png, bowl3.png, bowl4.png

Öğrenme Senaryosu 13 - Hikâye anlatma

Öğrenme Senaryosu Adı	Hikâye Anlatma
Geçmiş Programlama Deneyimi	Karakteri gizleme ve gösterme Koşulluları kullanma Söyle (say) kullanma (Görünüşler grubundan) .....*saniye bekle'*yi (wait for…seconds) kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Taşıma ve boyut değiştirme ● Yayın yapma ● Hikâye anlatımının yapısını oluşturma ● Sahnelerin arka planını değiştirme Algoritmik düşünceye odaklı özel öğrenme çıktıları. ● Öğrenciler hikâyedeki karakterlerin diyaloglarını ve etkinliklerini planlar. ● Öğrenciler, karakterler arasındaki diyalog için yayın göndermeyi kullanır. ● Öğrenciler, karakterler için hareket ettirme ve boyut değiştirmeyi kullanır. ● Öğrenciler karakter göster ve gizle kullanır. ● Öğrenciler, karakter kodunu yeniden düzenler ve genişletir.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	***Kısa Açıklama:*** Tavşan, Alice Harikalar Diyarında'nın hikâyesini anlatıyor. Hikâye anlatımına Alice Harikalar Diyarında etiketli arka planda birkaç cümle ile başlar. Alice'in hikâyesi ormanda başlar. Alice yürüyor ve "Neredeyim?" Diye merak ediyor. / Alice'in uzaklaştığını fark etmek için, hareketle yavaş yavaş boyutu küçülür /. Alice bir kavşağa gelir ve bir ağaçta Cheshire

	Kedisi'ni görür. Alice ve Cheshire Kedisi arasında bir konuşma başlar. Konuşma resimde sunulmuştur. ***Görevler:*** Öğrenciler, bir bekleme bloğu aracılığıyla diyaloğu senkronize etmeye dayalı olarak Alice ve Kedi arasındaki toplantının öyküsünün kısa bir örneğini denemelidir. Daha sonra, yayın mesajlarını kullanarak hikâyenin ikinci bir versiyonunu gözden geçirirler. Mesajlaşma komutları girilir. Öğrenciler resimdeki metne göre karakterlerin kodunu tamamlarlar. Kediyle tanışmadan önce Alice'i ormanda hareket ettirerek sahne dekorunu değiştirmenin getirilmesi görevi karmaşıktır. **Amaç: Öğrenciler hikâye anlatımını nasıl planlayacaklarını, karakter etkinliklerinin ve sahne değişikliklerinin senkronizasyonu için yayın mesajlarının nasıl kullanılacağını öğrenecekler.** 
Faaliyetin Süresi	90 Dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, oyun temelli öğrenme, problem çözme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma/ Ön Tartışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) 1- Öğretmen öğrencilerle Alice Harikalar Diyarında'nın hikâyesini tartışır ve Alice'in Cheshire kedisi ile tanışmasının resmini gösterir. Alice'in hikâyesinin kodlama kullanılarak yeniden oluşturulabileceğini açıklar. Ardından öğrenciler, projeyi başlatmak ve sprite'ların kodlarına bakmakla görevlendirilir.

https://snap.berkeley.edu/project?user=ddureva&project=Alice_1

Tartışma: *İlk önce kim konuşmaya başlar? Alice ve Kedi ne zaman konuşmaya dahil oluyor? Karakterlerin diyaloğunda neden senkronizasyon yok?*

Cevap: *Karakterlerin her birinin "konuştuğu" ve "bir karakterin yanıtlarını bitirmesini bekleyecek bir zaman aşımının olmaması" zamanlamanın*

yanlış hesaplanmasından kaynaklanmaktadır.

Kodlar yorumlanır ve tablo tamamlanır.

Karakter	**Aktivite**	**Başlangıç Zamanı**	**Bitiş Zamanı**	**Süre**
Tavşan	Merhaba de! Alice ve Harikalar Diyarı'ndaki maceralarını duydunuz mu? Şimdi onun hikâyelerinden birine bakalım.	0	14	14
Alice	**Şunu söyle:** Lütfen bana	9	21	12

	söyler misiniz, buradan hangi yöne gitmeliyim?			
Kedi	**Şunu söyle:** Bu, nereye gitmek istediğinize bağlıdır.	10	20	10

Sonuç şudur: *…. Saniye bekle (wait for… second) bloğu ile* senkronizasyon, hikaye anlatımında karakterlerin davranışlarında hatalara yol açabilir.

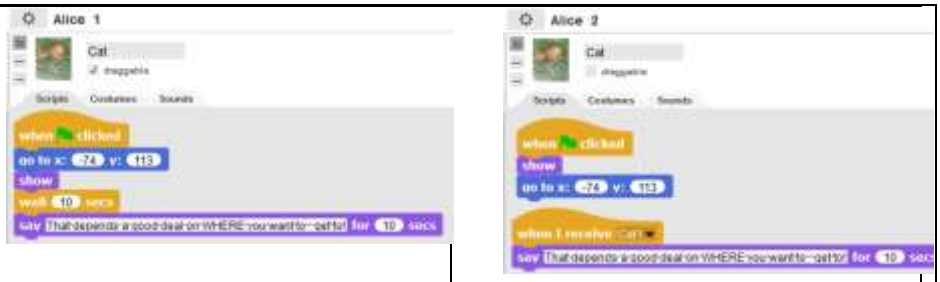
2. Öğretmen, proje kodunu başlatmak ve gözden geçirmekle görevlendirilir.

https://snap.berkeley.edu/project?user=ddureva&project=Alice_2

Şimdiye kadar bilinmeyen komutlar neler?

Alice_1 ile Alice_2 kodlarının mukayesesi

Alice_1	Alice_2
	
	



2. Yayınlama (broadcasting) blokları gösterilir:

Yayın mesajlarının tüm karakterleri hedeflediği, ancak sadece bazı karakterler tarafından alınabileceği öğrenciler ile tartışılmalıdır. *Yayın…* ve *bekleme (broadcast…* ve *wait*) bloğu, mesajı alan tüm karakterlerin eylemlerini gerçekleştirmesini gerektirir ve ardından mesajı gönderen hareketli grafiğin eylemleri devam eder.

Öğretmen yayın mesajını nasıl isimlendirileceğini ve event (olaylarda) yer alan geldiğinde (when I receive) komutunun nasıl kullanılacağını gösterir.

1. 

-

İsimlendirmenin yapılışı

,

2. Karakter tarafından alınması gereken mesajın seçilmesi.

3. Grup, resimdeki hikâyenin nasıl tamamlanacağını tartışır. Mesajlar nasıl adlandırılır? Örneğin, Kediden Alice'e veya Alice2'ye ve Alice'ten kedi veya kedi1'e mesaj gönderilmesi.

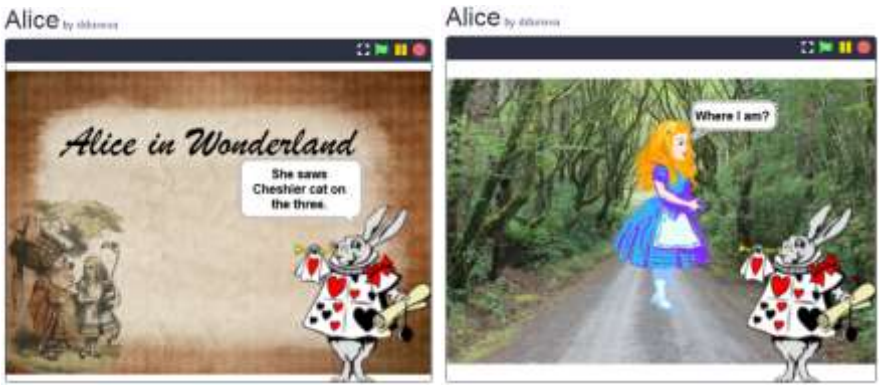
4. Öğrenciler hikâyelerindeki bu kısmı ikili gerçekleştirebilirler.

5. *Öğretmen, hikâye anlatmanın genellikle sahne kostümlerinde bir*

değişiklik gerektirdiğini söyler. "Tavşan hikâyesini giriş zemininde başlatıp, aksiyonu Alice'in yürüdüğü ormana taşıyıp" Neredeyim? Diye merak ederek Alice'in hikâyesini daha eksiksiz hale getirelim. Alice sahneden hareket ederek uzaklaştıkça karakterin boyutu giderek küçülmelidir. Sonra kendini bir dönüm noktasında bulur ve Cheshire kedisini görür. Konuşma ikisi arasında başlar.

6. Öğretmen aşağıdaki projeyi gösterir.

https://snap.berkeley.edu/project?user=ddureva&project=Alice

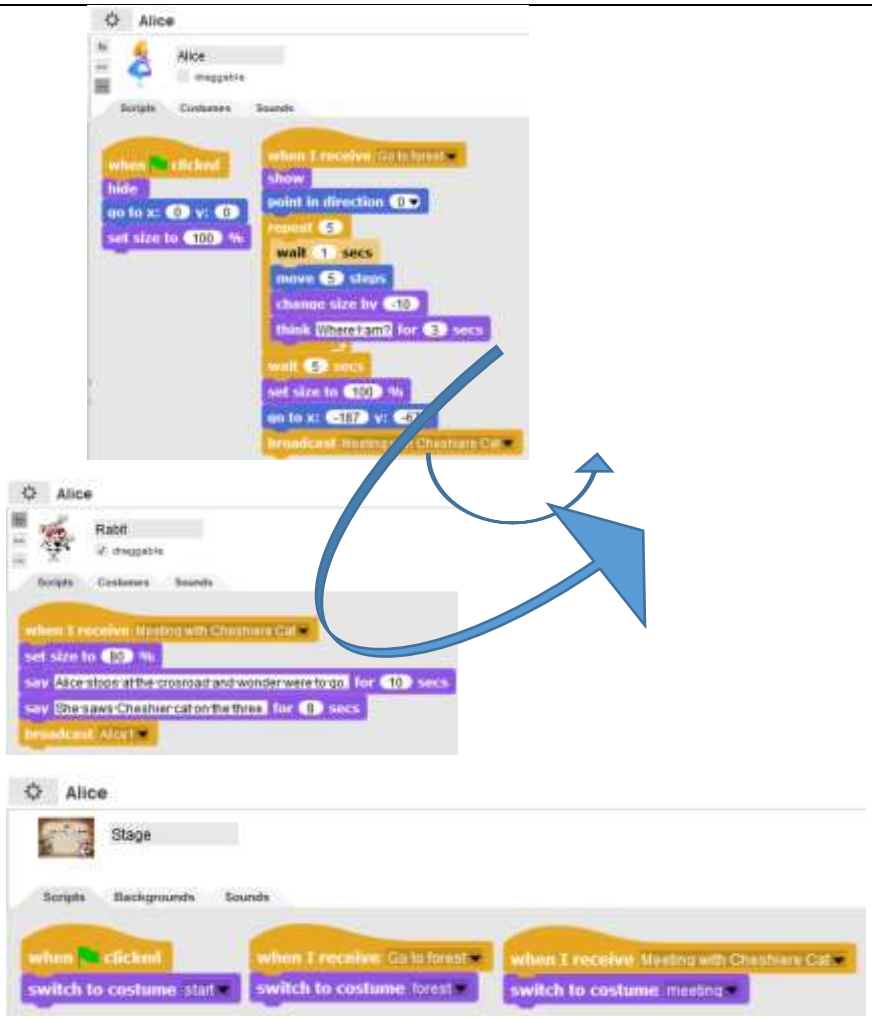


Karakterlerin hareketlerindeki ve sahnelerdeki değişiklikler yorumlanır.

"Sahne ne zaman değişir? Alice ne zaman ortaya çıkıyor ve eylemleri neler? Kedi ne zaman ortaya çıkıyor ve eylemleri neler?"

Alice_2 projesindeki sahneler öğrenciler ile tartışılır. Biri zaten kullanılmış olan toplam 3 sahne var. Başlamak için hangi sahne kullanılmalı? Başlangıçta Alice ve Kedi'nin karakterlerinin görünmesini önlemek için ne yapılmalıdır? Sahne dekoru nasıl değiştirilir?

Tavşanın giriş sözlerinden sonra sahneyi değiştirmek için bir yayın kullanılabilir. Sahne değiştirildiğinde Ormana git mesajından sonra Alice görünebilir.

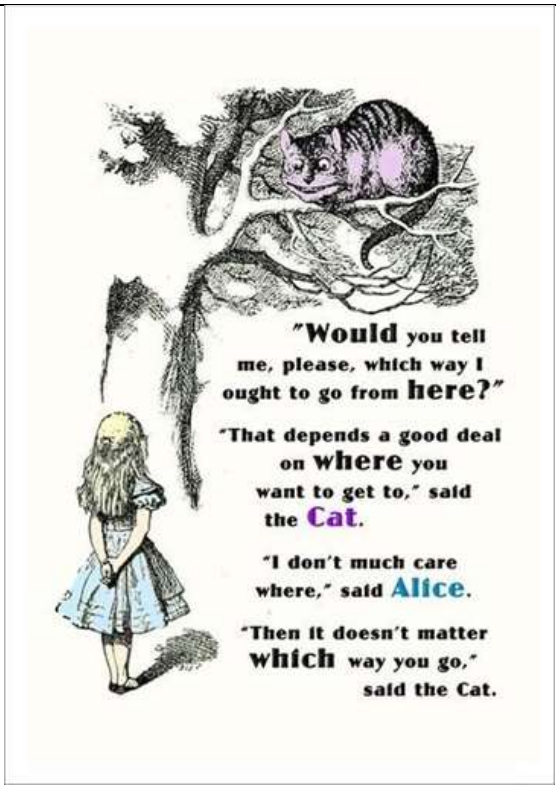


Alice ormanda yolundayken yürür ve "dolanır", bu nedenle daha fazla gerçekçilik için boyutu -% 10 azalır. Bu, *tekrar döngüsü (repeat loop)* kullanılarak 5 kez tekrarlanır.

Kavşağa ulaştığında, sahne "Cheshire Kedisiyle Buluş" mesajı ile değiştirilir. Bu mesaj aynı zamanda büyüklüğünü% 80'e düşüren Tavşan tarafından da alınır ve küçültülmüş hali ile hikâyeyi anlatmaya devam eder. Bu aşamada, dekorun bir parçası olduğu için kedi karakteri gösterilmez. Kedi1 (Cat1) mesajında görünür.

Öğretmen, harici bir grafik düzenleyici kullanarak kedinin dekordan çıkarıldığını açıklayabilir. (Ne yazık ki Snap!, Scratch 3.0'dan farklı olarak çok fazla grafik düzenleyici yeteneği sağlamaz).

Tavşan mesajının yayınlanmasının ardından Alice 1 hikâyesi Alice 2

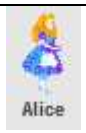
	projesinde olduğu gibi devam eder. 3. Öğretmen, bir hikâye anlatmak için kişinin önce bir olay örgüsü icat etmesi gerektiğini söyler. Hikâyenin senaryosunu açıklamak için ek bir tablo kullanılabilir. (Ek 1) öğretmenin takdirine bağlı olarak, bitmiş tablo verilebilir veya kısmen doldurulabilir ve öğrenciler, resimle yönlendirilerek tamamlayabilir. 4. Öğrencilere ikişerli gruplar halinde, incelenen senaryoları açıklama ve Alice_2 projesinin hikâyesini tamamlama görevi verilir.
Öğretmenler İçin Araçlar ve Kaynaklar	Snap teki tüm aktivite https://snap.berkeley.edu/project?user=ddureva&project=Alice
Öğrenciler İçin Kaynaklar ve Materyaller	 ● https://snap.berkeley.edu/project?user=ddureva&project=Alice_1 ● https://snap.berkeley.edu/project?user=ddureva&project=Alice_2 ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx))

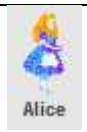
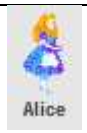
Appendix 1. Story Plots/Scenarios

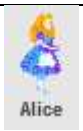
Ad	Tasarım	Etkinlik	Notlar
1. Başlangıç		Hikâye Yeşil bayrak tıklandığında yandaki sahneyle başlar.	Bu arka planda, Tavşan hikâyeyi tanıtır.
2. Orman		Manzara, Tavşan girişini tamamladığında ortaya çıkar (Ormana git mesajı gönderilmiştir).	Bu arka planda, Alice sahnenin ortasında konumlanmış görünür. "Neredeyim?" Diye merak ederek hareket etmeye başlar. Karakterinin boyutu kademeli olarak 5 defa %10 küçültür. Yolun sonuna ulaştığında (bir kavşakta), sahne buluşma olarak değişir. (Alice mesaj – yayın (message -broadcast) gönderir. Cheshire Kedisi ile Buluşma)

3. Buluşma		Alice'in Cheshire Kedisi ile Toplantı mesajı alındığında görünür.	Burada Alice ve kedi arka planın bir parçasıdır. Alice'in hareketli karakterini kullanmak için mesajdan önce, imajı dekordan kaplayacak şekilde konumlandırılır. Kedi karakteri daha sonraki bir aşamada görünür. Sahne değiştikçe Tavşan hikâyeyi anlatmaya devam eder. Daha sonra Alice ve Cheshire Kedisi arasında bir konuşma gerçekleşir.

Karakterler

Karakter	**Etkinlik**	**Arka Plan**
	Başlangıçta: - Merhaba! (2 sn.) -- Alice ve Harikalar Diyarı'ndaki maceralarını duydunuz mu? (6 sn.) --- Şimdi onun hikâyelerinden birine bakalım! (6 sn.) "Ormana git" (*Go to forest*) mesajını gönderir.	
	Başlangıçta: Sahneden gizlenir. Orta sahne konumunda ve % 100 boyutta, yeni arka planda görüntülenmeye hazır.	
	Başlangıçta: Sahneden gizlenir. x: -74, y: 113 noktasında konumlandırılmıştır. (Konumlar, Kedi karakteri buluşma	

	sahnesinde ayarlandıktan sonra önceden belirlenir).	
	"Ormana git" mesajını alır. Karakter sahnede gösterilir. 5 kez tekrarlanır: 1 saniye bekle; 5 Adım hareket; boyut küçültme (%-10 kadar değişiklik); merak ediyorum: Neredeyim ben? Bir sonraki dekor için hazırlanır: 5 saniye beklenir; karakterin boyutunu eski haline getirme (% 100 değişiklik) ve x: -187, y: -67 konumunda konumlandırma Mesaj Gönderir: "Cheshire Cat ile buluşma"	
	Herhangi bir eylem yok. Yalnızca bir önceki dekordan görünür hale gelir.	
	*Mesajı alır: "Cheshire Kedisi ile buluşma"* *% 80'e yeniden boyutlandırır.* *Der ki: "Alice dönüm noktasında durur ve nereye gideceğini merak eder" (10 saniye).* *Der ki: "Alice Cheshire Kedisini ağaçta gördü" (8 saniye).* *"Alice1" mesajını gönderir.*	
	Alice1 mesajını alır. Öne doğru hareket eder (Bu gereklidir. Çünkü Kedi ondan sonra görünür. Bbu da Alice'in ön katmanda değilse bir diyalog kutusunda görünmesini engeller). Der ki: "Merhaba!" (2 sn.). Der ki: "Lütfen bana buradan hangi yöne gitmem gerektiğini söyler misin?" (10 saniye). Kediye "Cat1" yayın mesajını gönderir.	

	Cat1 mesajını alır. Karakter sahnede gösterilir. Der ki: "Bu, NEREYE gitmek istediğinize bağlı!" (10 saniye). “Alice2” mesajını gönderir.	
	*“Alice 2” mesajını alır*. Der ki: …… *“Cat2”* mesajını gönderir.	
	*“Cat2”* mesajını alır. Der ki: …… *“Rabbit1”* mesajını gönderir.	
	*“Rabbit1” mesajını alır*. Der ki: “Hikâyenin dersi nedir?” (8 saniye.) Der ki: “Hangi yoldan gideceğini bilmek için önce hedefini belirlemeli”.	

Öğrenme Senaryosu 14 - Çizim yapma

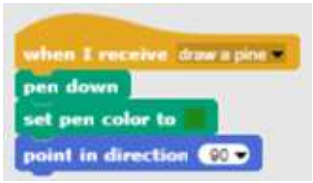
Öğrenme Senaryosu Adı	Çizim Yapma
Geçmiş Programlama Deneyimi	Karakter ekleme Bir yöne bak komutu kullanma Sayma noktası için değişkenleri kullanma Döngü tekrarını kullanma Koşul ifadeleri kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullu ifadeler ● Döngü ● Bir yöne bakma ● Operatörler – Dört İşlem Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, çizim yapmak için kalem kullanır. ● Öğrenci, çizim yapmak için döngüler kullanır. ● Öğrenci, çizim yaparken bir değişkenin değerini değiştirir. ● Öğrenci, sahnede nesneler çizmek için işaret kullanır. ● Öğrenci, hareketli grafiği kontrol etmek için yayını kullanır. ● Öğrenci, aşamayı değiştirmek için koşullu ifadeler kullanır. ● Öğrenci, sahneyi değiştirmek için > operatörü (işlemleri) kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	***Kısa Açıklama:*** İklim çok değişti. Hava sanayi nedeniyle kirlendi. Hava kalitesini iyileştirmek için ağaçların dikilmesi gerekiyor! ***Görevler:*** Hava kalitesini iyileştirmek için öğrenciler, bir karakter programlayarak iki tür farklı ağaç çizmelidir - çam ve meşe ve bu tür ağaçları sembolize eden düğmeler. Bir düğmeye tıklandığında, belirli bir ağaç türü çizilir. **Amaç: Öğrenciler Snap!'te çizim yapmayı, rengi ve kalem kalınlığını değiştirmeyi ve yeni bir olaya neden olan değişkenleri ve koşullu ifadeleri nasıl kullanacaklarını öğrenecekler.**

Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, oyun temelli öğrenme, problem çözme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Oyunun başında sahnede iklim değişikliğine neden olan bir fabrika ve hava kalitesini gösteren bir değişken gösteriliyor. Hava kalitesini iyileştirmek için ağaçların dikilmesi gerekiyor. Çam ve meşe olmak üzere iki farklı ağaç türü çizilebilir. Bir çam eklendiğinde hava 3, meşe eklendiğinde hava 2 birim iyileştirilir. Hava kalitesi 10 birime ulaştığında sahne arka planı bir çayıra dönüşür. [Adım 1] Öğrenciler, arka planlardan (fabrika ve çimen) ve karakterlerden (kurşun kalem, çam ve meşe olarak adlandırılan) bir şablon içeren İklimi İyileştirme (Improve *the Climate*) programını açmalıdır. Ayrıca, yeni karakter olarak kalem eklemelidirler (önerilen karakter pencil a" (kalem a)). Karakter çok büyük olduğu için% 50'ye düşürülmelidir. Kalemin başlangıç konumu (koordinatlar) da belirtilmelidir. Örneğin X = -10, y = -10.  

[Adım 2]

Kalem karakteri "meşe" ve "çam" mesajları almalı ve mesaja yanıt olarak uygun ağaçları çizmelidir. İlk olarak, kurşun kalem karakterini işaretleyin ve karakter "çam" mesajını aldığında bir kalem kullanarak çam çizimini etkinleştirecek kodu ekleyin.

Kanopiyi (Çeviri Notu: Kapalılık, ağaç tepelerinin yeri örtme derecesi) üçgen şeklinde çizmek için *bir yöne bak (Point in Direction)* 90 olarak ayarlanmalı ve rengi belirlenmelidir.



Çam ağacından bir Kanopi çizmek için karakteri 40 adım hareket ettirip ardından 120 derece sola döndürün.

Bu hareket üç defa tekrarlanmalı.

Kanopiden sonra gövde de çizilmelidir. Gövdenin doğru pozisyonda olması için 22 adım hareket ettirin.

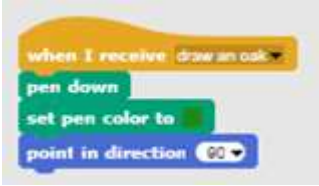
Bundan sonra kalem rengini kahverengiye ayarlayın.

Sağa 90 derece dönün ve 10 adım hareket edin.

Bu hareket üç defa tekrarlanmalı.

Sonunda, karakter bir sonraki harekette iz bırakmayacak şekilde kalemi yukarı kaldırmak gerekir. Ayrıca kalem rastgele pozisyona getirilmelidir.

[Adım 3]Benzer şekilde, meşe çizmek için kodu kalem karakterine eklemek gerekir. Karakter "meşe" mesajını aldığında meşe çizilmelidir. Kanopiyi yuvarlak tutmak için *bir yöne bak (Point in Direction*) 90 olarak ayarlanmalı. *Kalem aşağıda olmalı (pen down*) ve renk ayarlanmalıdır.



Meşe ağacı kanopisi çizmek için karakter 1 adım hareket etmeli ve her adımdan sonra 3 derece sola dönmeli.

Bu hareket 120 defa tekrarlanmalı.

Kanopi bittikten sonra gövde çizilmelidir. Kurşun kalem karakteri çizilen dairenin ortasına -3 adım ile hareket ettirilmeli ve kalemin rengi kahverengiye dönmelidir.

Gövde çizmek için karakter 90 derece sağa döndürülmeli ve 10 adım hareket ettirilmelidir.

Bu hareket 3 defa tekrarlanmalı.

Çizim tamamlandığında, kalem karakterini hareket ettirirken çizgiyi çizmemesi için kalemin yukarı (pen up) kaldırılması gerekir.

Meşe çekildikten sonra kalem rastgele pozisyona getirilmelidir.

[Adım 4]Daha sonra, oyuncu *Karakteri Sil'i (Clear sprite)* tıkladığında çizilen tüm ağaçların silinmesini sağlayan kod öğrenciler tarafından eklemelidir. *Karakteri Sil (Clear sprite)* tıklandığında, tüm ağaçları temizlemek için bir mesaj yayınlar. Kalem karakteri bir mesaj aldığında,

	çizilen tüm ağaçları siler.  [Adım 5] Mevcut hava kalitesini göstermek için yeni bir değişken "temiz hava (clean air)" oluşturun. Başlangıç değerini 0 olarak ayarlayın ve değişkeni sahnede gösterin.  Bir çam ağacı dikildiğinde hava 2 birim iyileşir. Bu nedenle, çamın her tıklanışında "temiz hava" değişkeninin değerini 2 değiştirecek olan bloğu çam serisine ekleyin.  Bir meşe her çekildiğinde hava 3 birim iyileşir. Bu nedenle "temiz hava" değişkeninin değerini çam her tıklandığında 3 değiştirecek olan bloğu meşe karakterine ekleyin.  [Adım 6]"Temiz hava" değişkeni 10'a ulaştığında, sahne çim olarak değişmelidir. Bu nedenle indirilen materyallerden sahne için yeni bir arka

plan yani "çimen" ekleyin (arka plan indirilen materyallerden alınmıştır).

Kalem karakterine “Control“ (kontol) paletinden “When“ (..iken/…zaman) şapka bloğunu ekleyin.

Sonra, > işlemini (operator) ekleyin.

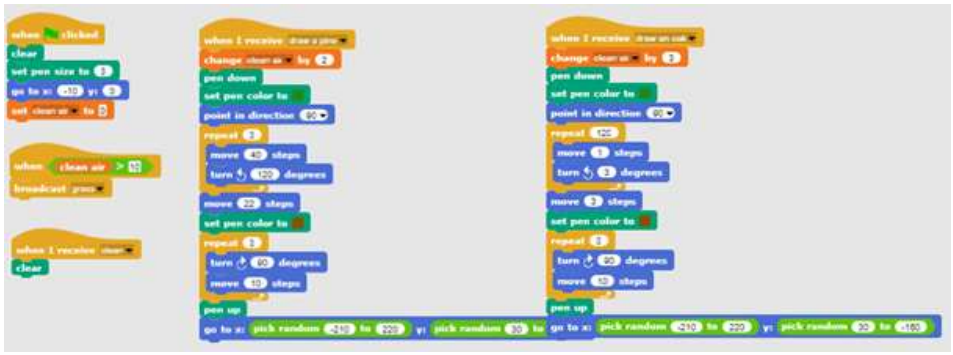
Karakterin "temiz hava" değişkeni 10'dan büyük olduğunda "çim" mesajını yayınlamasını (broadcasts) ayarlayın.

"Çim" mesajı alındığında kostümü "çim" olarak değiştirmek için sahneye kodu ekleyin.

[İLAVE GÖREVLER]

Hava artık kirlenmediğinde ortaya çıkan hayvanları ekleyerek oyunu yükseltebilirsiniz.

[NİHAİ KOD]

	Çam  Meşe  X  Kalem  Sahne 
Öğretmenler İçin Araçlar ve Kaynaklar	Snap! project “Drawing”: https://snap.berkeley.edu/project?user=tadeja&project=Improve%20the%20climate (9.1.2020)
Öğrenciler İçin Kaynaklar ve Materyaller	• Programming language Snap!: https://snap.berkeley.edu/ (9.1.2020) • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 15 - Fareyi yakalamak

Öğrenme Senaryosu Adı	Fareyi yakalamak
Geçmiş Programlama Deneyimi	● Öğrenci bir arka plan ekleyebilir. ● Öğrenci, yeni bir hareketli grafik ekleyebilir. ● Öğrenci yeni bir ses ekleyebilir. ● Öğrenci, karakterin nasıl bir şey söylemesini sağlayacağını bilir. ● Öğrenci, animasyon yapmak için karakterin kostümünü nasıl değiştireceğini bilir. ● Öğrenci, olayları kullanarak ok tuşlarıyla nesne hareketini gerçekleştirebilir ve kısıtlamaları dikkate alır. ● Öğrenci, iki farklı durumu ayırt edebilir ve bunları mantıksal ifadelerle nasıl ifade edeceğini bilir. ● Öğrenci, koşul ifadelerini nasıl kullanacağını bilir.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Sonsuz döngü; ● Rastgele sayılar; ● Sayaç ● Zamanlayıcı. Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, karakterleri hareket ettirmek için sonsuz döngüyü kullanır. ● Öğrenci, karakterin konumunu belirlemek için, karakterin rastgele hareket etmesi için ve karakterin rastgele dereceler ile dönmesini sağlamak için rastgele sayılar (random numbers) kullanır. ● Öğrenci, yakalanan fareleri saymak için sayaç uygular ve oyuncunun ne kadar başarılı olduğunu ölçmek amacıyla sayaçtaki son değeri kullanır. ● Öğrenci oyunun sonunu belirlemek için zamanlayıcıyı kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Tanıtım**: Oyuncunun (kedi) fareyi yakalaması gereken bir program yazar. **Görev**: Kedinin fareyi yakalayacağı etkinliği programlayın. Kedi, ok

	tuşlarına sahip bir oyuncu tarafından hareket ettirilecek ve fare rastgele hareket edecektir. Kedi fareye dokunduğunda, fare gizlenecek ve rastgele bir yerde görünecektir. Ayrıca, kedinin fareyi kaç kez yakaladığını sayan bir sayaca sahip olmamız gerekir. Oyunu bitirmek için ayrıca bir zamanlayıcıya ihtiyacımız var. Aktiviteden sonra kızın oyuncunun ne kadar başarılı olduğunu özetlemesi gerekir. Bunu oyuncunun fareyi kaç kez yakaladığını söyleyerek gerçekleştirebiliriz. **Amaç: Öğrencilere çok değişkenli rastgele değer atama kavramı tanıtılacaktır. Öğrenciler, operatörleri (işlemler) nasıl kullanacaklarını / rastgele [x] - [y] bloklarını nasıl seçeceklerini öğrenecekler.**
Faaliyetin Süresi	45 dk.
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme, oyun temelli öğrenme.
Öğretme Formları	Ön Öğrenme İkili Çalışma, Grup Çalışması.
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) **Motivasyon-Giriş** Oyunu göstererek öğrencileri motive ediyoruz. Onlarla bu oyunu programlamaya nasıl başlayacaklarını tartışıyoruz. Öğrencilerle birlikte adımların sırasını belirliyoruz. Örneğin: 1. Arka planı seçin ve karakteri ekleyin. 2. Kediyi ok tuşlarıyla hareket etmesi için programlayın. 3. Fareyi rastgele hareket edecek şekilde programlayın. 4. Fareyi kediye dokunduğunda gizlenecek (ve rastgele bir yerde görünecek) şekilde programlayın.

5. Program sayacını ekleyin.

6. Zamanlayıcı ekleyin ve oyunun sonunu belirleyin.

7. Kızı ekleyin ve oyuncunun ne kadar başarılı olduğunu özetlemesi için programlayın.

8. Kızı fareye dokunduğunda atlayacak şekilde programlayın.

9. Kedi / fare sesini ekleyin.

10. Vb.

Öğrenciler adımlarda yardımcı olabilir veya oyunun kendi kurallarını koyabilirler (ancak adımları takip etmeleri gerekir).

Rastgele değer ataması için operatörü tanıtıyoruz.

Öğrenciler, öğretmenin desteği ile aşağıdaki görevleri çiftler / gruplar halinde programlar.

Uygulama

[Adım 1]

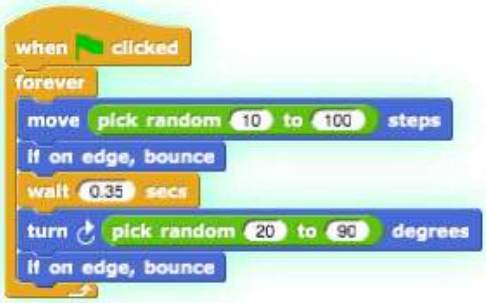
İlk adım, oyunun arka planını belirlemektir. Öğrenciler çevrimiçi olarak ücretsiz resim ararlar. Ardından, yeni karakter eklerler - kedi ve fare.

[Adım 2]

Öğrenciler kediyi ok tuşlarıyla hareket etmesi için programlar. Burada, kedinin kenarlarda olması durumunda ne olacağını belirlemek zorundalar.

[Adım 3]

Öğrenciler fareyi rastgele hareket edecek şekilde programlamalıdır. Bu durumda mantık, sonsuz döngüdeki farenin rastgele sayıda adım atması ve rastgele bir dereceye kadar dönmesidir. Öğrenciler bunu Hareket/hareket et [x] Adımlar (*Motion/move[x]Adıms*) bloğu ve Hareket/dönüş [x] (*Motion/turn[x]degrees*) derece bloğu ile yaparlar ve içine rastgele [x] ila [y] (*pick random[x]to[y]*) operatörünü yerleştirirler.

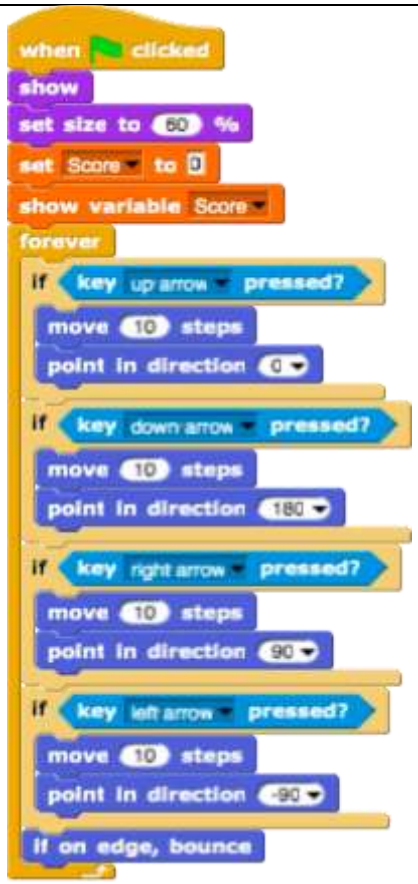


[Adım 4]

Sonraki adım, fareyi kediye dokunduğunda gizlenecek şekilde programlamaktır. Buradaki mantık, farenin kediye dokunduğunda rastgele bir yerde gizlenmesi ve görünmesidir. Bu durumda oyun farenin ilk yakalandığı anda bitmemektedir. Öğrenciler buraya kendi kurallarını ekleyebilirler. Her durumda rastgele [x] - [y] seçme (*pick random[x]to[y]*) operatörünü kullanmaları gerekir.

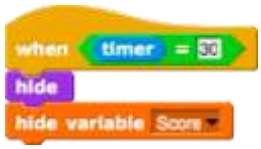
[Adım 5]

Farenin kaç kez yakalandığını bilmek istememiz durumunda, bir sayaç eklememiz gerekir. Öğrenciler yeni bir değişken oluşturur "– score" (-puanla) ve bunu kedinin koduna ekler. Oyunun başındaki skor her zaman sıfır olmalıdır. Öğrenciler bunu *Değişkenlerle/*[x]'e a*yarla[değişken] (Variables/set[variable]to[x])* bloğu ile yaparlar. Skorun oyunculara gösterilmesini istiyorsak, öğrencilerin *değişken göster [değişken*] (*show variable[variable]*) bloğunu eklemesi gerekir. Daha sonra öğrenciler, kedinin fareye dokunup dokunmadığını kontrol etmek için yeni bir kontrol bloğu "*(Kontrol / ne zaman) ((Control/when))"* ekler. Kedi fareye dokunursa, sonuç 1 artar (*Variables/change[score]by[x]*).



[Adım 6]

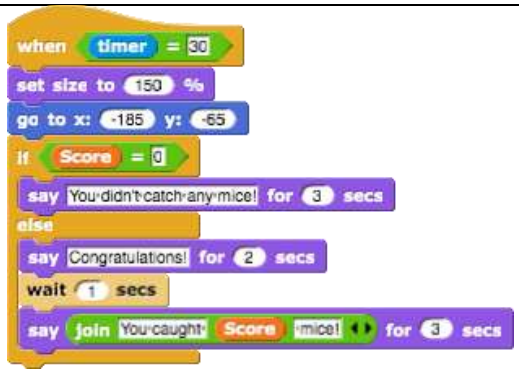
Öğrenciler oyunun ne zaman biteceğini belirlerler. Bunu zamanlayıcıyı ekleyerek yapabilirler. Bir süre sonra (örneğin 30 saniye) fare ve kedi kaybolur. Puan değişkeni gizlenir ve oyun sona erer.



Öğrenciler bu blokları kedi ve farenin komut dosyalarına eklemelidirler.

[Adım 7]

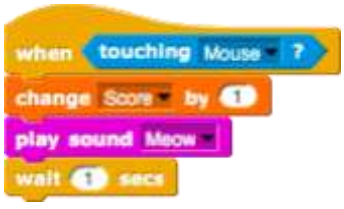
Öğrenciler, oyuncunun ne kadar başarılı olduğunu özetlemek için kızı programlamalıdır. Eğer oyuncu fare yakalamazsa, kız "Sen hiç fare yakalayamadın!" der. Aksi takdirde, "Tebrikler! X fare yakaladınız!" der.



[İLAVE GÖREVLER]

Öğrenciler oyunlarına herhangi bir öğe ekleyebilir. Örneğin, bir fareye her dokunduğunda sıçrayan kız.

Öğrenciler ses ekleyebilir. Örneğin, kedinin sesini eklerler. Fare yakalandığında ses çıkar.



Düşünme ve Değerlendirme

Öğrenciler bu kodları ayarlayacaklar.

- Fare 20 den 60 adıma sonsuza dek gider.

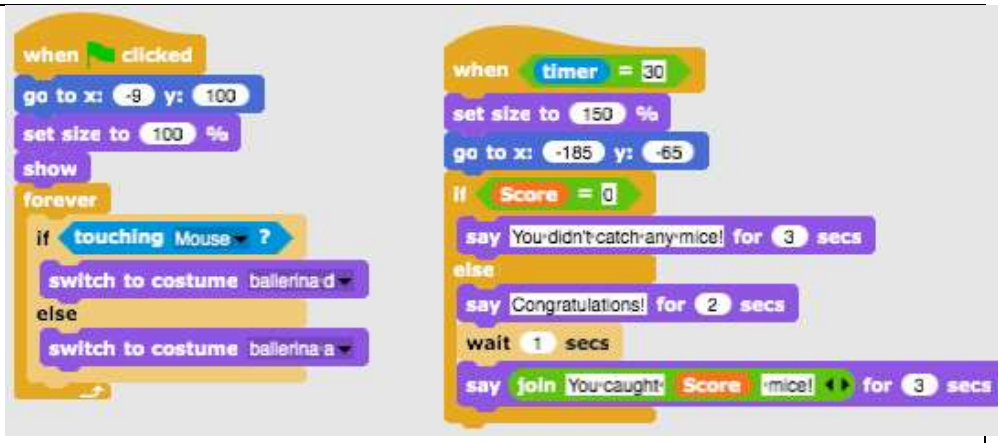
- Fare kediye dokununca x = 100 konumuna gider.
- Fare sonsuza dek 90 derece döner.
- Vb.

[Nihai Kod]

Fare

Kedi

Kız

	 *Arka plan* 
Öğretmenler İçin Araçlar ve Kaynaklar	• Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=tadeja&project=Catch%20the%20mouse • Ücretsiz resim için: https://pixabay.com/ • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. • Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	• Snap!'teki şablon https://snap.berkeley.edu/project?user=tadeja&project=Catch%20the%20mouse_0 • Ücretsiz resim için: https://pixabay.com/ • Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 16 - Piknik için yiyecek satın almak

Öğrenme	Piknik için yiyecek satın almak

Senaryosu Adı	
Geçmiş Programlama Deneyimi	Karakter için metin ekleme Karakter gösterme ve gizleme Operatörleri kullanma Değişkenleri kullanma Dize birleştirme kullanma Koşul ifadeleri kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullar ● Operatörler Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, farklı karakterlerin fiyatını belirlemek için değişkenleri kullanır. ● Oyuncu yiyecek satın aldığında bütçe değiştiği için öğrenci değişkenlerin değerini değiştirir. ● Öğrenci, paranın kullanılabilirliğini kontrol etmek için “if” ifadesini kullanır. ● Öğrenci, fiyatları ve bütçeyi karşılaştırmak için operatörleri kullanır. ● Öğrenci, değişkenlerin değerini değiştirmek için operatörleri (çıkarma) kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Açıklama:** Kız pikniğe gidiyor ve yiyecek almak için yardıma ihtiyacı var. 15 EURO var ve daha fazla harcayamıyor. Bir şey satın aldığında bütçenin değeri değişir. Bütçesi çok düşükse, seçtiği yemeği satın alamaz. **Görevler:** Öğrenciler üç farklı karakter programlamak zorundadır. Bir “kız”, “yiyecek” (küçük değişikliklerle çoğaltabilecekleri) ve “bitir düğmesi”. Kız talimatlar verir. Oyuncunun ne kadar parası olduğunu söyler ve sonunda (bitir düğmesine tıklayarak) oyuncunun kaç tane sağlıklı ve sağlıksız ürün aldığını söyler. Yiyecek, fare işaretçisi üzerine geldiğinde fiyatını söyler. Oyuncunun yeterli parası varsa, bir ürün

	satın alabilir ve bütçenin değeri değişir. Aksi takdirde yiyecek satın alınamaz. **Amaç: Öğrenciler değişkenlerle nasıl çalışılacağını öğrenecekler. Farklı başlangıç değerleri belirleme, değişkenlerin değerini karşılaştırmak için koşullu ifadeler kullanma, değişkenlerin değerini değiştirme, sağlıklı yiyecekleri saymak (olmayan) için değişkenler kullanma. Ek olarak, metin eklemeyi, metinleri ve "if" ifadesini birleştirmeyi tekrarlayacaklar.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme, oyun temelli öğrenme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Kız bir bakkaldan piknik için yiyecek alıyor. 15 EURO var. Fare işaretçisi üzerine geldiğinde yiyeceğin fiyatını görebilir ve seçili yiyeceği tıklayarak satın alabilir. Sadece yeterli parası olana kadar yiyecek satın alabilir. Bitir düğmesine tıklayarak, oyuncunun kaç tane sağlıklı ve sağlıksız ürün aldığını söyler.

[Adım 1]

Bu faaliyet, bir bireysel çalışma veya çiftler halinde çalışma olarak planlanmıştır. Öğretmen bazı ipuçları verir. Bazı zor kısımları açıklar ve gerektiğinde yardımcı olur. Öğrenciler arka planı seçer ve bir ana hareketli grafik ekler. Örneğin bir kız. Kız başta bazı talimatlar verir. Örneğin:

[Adım 2]

Bu oyunda birkaç değişkene ihtiyacımız olacak.

● Kullanılabilir para miktarını ayarlamak için bütçe.

● Oyuncunun kaç sağlıklı öğe satın aldığını saymak için healthy_food (sağlıklı_yiyecek).

● Oyuncunun kaç sağlıksız öğe satın aldığını saymak için unhealthy_food (sağlıksız_yiyecek).

● Her yiyecek ve her yiyeceğin fiyatını belirlemek için bir değişken (örneğin, karpuz_fiyatı).

Başlangıçta bütçe değişkeni örneğin 15 EURO olarak ayarlanabilir. Diğer iki değişken 0 olarak ayarlanmıştır. Bu kod [Adım 1]'den kız

kodunun önüne eklenebilir.

[Adım 3]

Öğrenciler bir karakter (yiyecek) ekler ve kostümünü seçerler.

Gıdanın (karpuz) kodu üç kontrol olayına ihtiyaç duyar:

a) Yeşil bayrak tıklandığında: Gıdanın fiyatını göstermek ve ayarlamak için.

Değişkenin fiyatının makul bir şekilde belirlenmesine izin verin (tabii ki 0 değil ve 1'den büyük).

b) Fareyle girildiğinde: oyuncuya ürünün maliyetinin ne kadar olduğunu söylemek için.

Öğrenciler Görünümler-Düşünce (*Looks – thinking*) bloklarını kullanabilirler. - birleştirme metni - değişken değeri - metin kullanımıyla düşünme bloğunu kullanabilir. Örneğin:

.

c) *Tıklandığında: burada küçük bir yansıma yapmaları gerekir.*

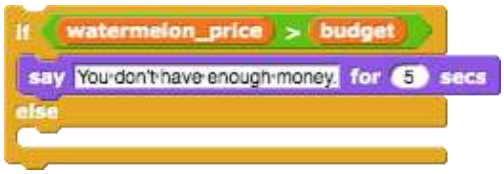
1) Oyuncu ürünü hangi durumda satın alabilir ve hangisinde alamaz?

2) Yiyecekleri satın alırsa bütçeye ne olur?

3) Satın alınan ürünleri nasıl sayarız?

4) Raftaki yiyeceklere ne olur?

1) Oyuncu yeterli parası varsa ürünü satın alabilir. Bu nedenle öğrencilerin iki değişkeni karşılaştırması gerekir. Bunlar bütçe ve karpuz_fiyatıdır. Karpuz, sahip olduğundan daha pahalıysa satın alamaz. Öğrenciler oyuncuya bu ürünü satın alamayacağını söylemek için metin ekleyebilirler.



2) Oyuncu 15 EURO'ya sahipse ve 4 EURO karşılığında bir karpuz satın alırsa, şimdi 15 - 4 = 11 EURO'dur. Yani bütçe değeri artık: 11'dir.
3) Önceki Büte Değeri – Karpuz Fiyatı (previous *budget value* – *watermelon_price*)

Öğrenciler bazı metinleri de ekleyebilir.

4) Satın alınan ürün adedinin sayılması sağlıklı gıda (*healthy_food*) değişkeni 1 değiştirilerek gerçekleştirilecektir.

5) Yiyecek tıklandığında gizlenir.

Mümkün olan çözümlerden biri;

[Adım 4]

Raflarda daha fazla yiyecek bulundurmak için öğrenciler karpuz karakterini kopyalayabilir. Diyelim ki, ikinci yemek pasta olacak. [Adım 3] kodunda bazı değişikliklerin yapılması gerekiyor.

Öğrencilerin yapması gerekenler:

- Kostümü değiştirmek.
- Yeni değişken oluşturmak kek_fiyatı (*cake_price).*
- Kek_fiyatı'na bir değer belirlemek.
- Bloklarda yer alan tüm (karpuz_fiyatı) *watermelon_price* kodunu kek_fiyatı (*cake_price)* ile değiştirmek.
- Kek satın almadaki geri dönüşleri değiştirmek.
- *Change healthy_food by 1* komutunu *change unhealthy_food by 1 ile değiştirmek.*

Örneğin, Kek için tıklandığında(*when clicked* code) kodu:

[Adım 5]

Oyuncu satın almayı bitirdiğinde, bitir (finish) düğmesine tıklar. Programa, oyuncunun düğmeye tıkladığını (yiyecek satın almakla bittiğini) söylemek için bir mesaj yayınlıyoruz.

[Adım 6]

Sonunda kız karakterine dönüyoruz. Oyuncu alışverişini bitirdiğinde

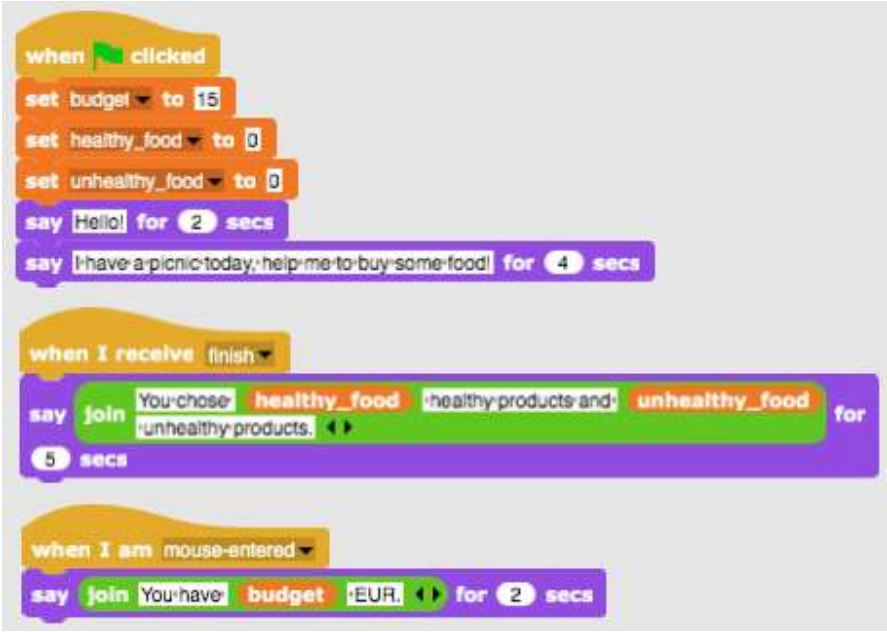
kızın kendisine kaç tane sağlıklı ve sağlıksız ürün aldığını söylemesini istiyoruz. Oyuncu bitir düğmesine tıkladığında, bir bitiş mesajı gönderilir. Kız, mesajın bittiğini aldığında, "X sağlıklı ürün ve Y sağlıksız ürün seçtiniz" gibi mesaj gönderilir.

[Adım 7]

Oyuncu, oyun sırasında herhangi bir zamanda kızın bütçesini bilgisayarın faresini kullanarak kontrol edebilir. Örneğin, şöyle bir şey söyleyebilir/düşünebilir:

[Nihai Kod]

Kız



	Yiyecek  Bitiş Düğmesi  [İlave Görevler] Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler: ● Oyunu her yiyeceği 3 kez satın alabilmek için değiştirin. ● Başlangıçta oyuncuya daha fazla para verin. ● Sonunda kız ayrıca kaç ürün satın aldığınızı da söyler. Örneğin, “2x karpuz, 1x üzüm, 2x patates kızartması aldınız”.
Öğretmenler İçin Araçlar ve Kaynaklar	● Snap!’teki tüm aktivite: https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic

	● Snap!'te çözümleri ile birlikte ek görevler: https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic%20%2B%20Add.%20Task ● Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček.* Ljubljana: Pasadena. ● Vorderman, C. (2017). *Računalniško programiranje za otroke.* Ljubljana: MK.
Öğrenciler İçin Kaynaklar ve Materyaller	Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 17 – Operasyonlar/İşlemler

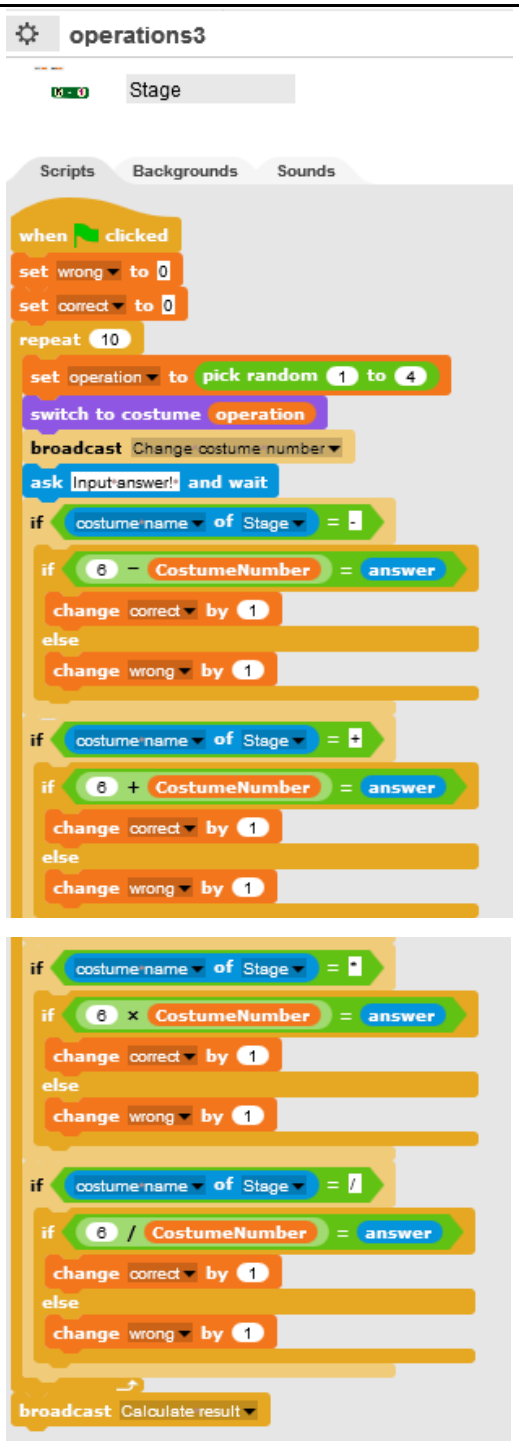
Öğrenme Senaryosu Adı	Operasyonlar/işlemler
Geçmiş Programlama Deneyimi	Puanları saymak ve sahne ile karakter kostümünü seçmek için değişkenler kullanma Karakter için sahne dekoru ve kostüm seçmek için rastgele sayı kullanma Tekrar döngüsünü kullanma Koşul ifadeleri kullanma Karşılaştırma için işlemleri kullanma Diyalog için algılamayı kullanma (sorun… ve bekleyin/ ask ….and wait) Yayın (broadcast) etkinliklerini kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullu ifadeler ● Döngü ● Algılama blokları (Sensing blocks) ● Yayın etkinlikleri Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenciler, puan sayımı için değişkenleri kullanır. ● Öğrenciler, puan sayımı için değişkenleri başlatır. ● Öğrenciler koşullu ve mantıksal işlemleri kullanır. ● Öğrenciler, karakterlerin grafiğini değiştirmek ve nihai sonucu hesaplamak için yayın olayını kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Açıklama:** Bir oyun oynarken oyuncunun Snap!'deki aritmetik işlemlerde ustalaşıp ustalaşmadığını kontrol edelim. Kurallar aşağıdaki gibidir: İlk işlenen sayısı 6 olan bir aritmetik işlem rastgele seçilir. İkinci işlenen 1'den 3'e kadar bir sayı olacak şekilde rastgele seçilir. Oyuncu doğru cevabı girmelidir. Doğru ve yanlış cevaplar sayılır. Oyunun sonunda doğru sonuç bildirilir.

	Görev: Öğrenciler sahneyi / sahne dekorunu / ve karakterin kostümünü tanımlamalıdır. Gerekli değişkenleri planlamaları, hangi bloklara ihtiyaç duyduklarını belirlemeleri, sonunda sahne ve hareketli grafik için kodlar oluşturmaları gerekir. Ek görevler şunlar olabilir: ● Sonuca bağlı olarak karaktere aşağıdakiler söyletilebilir: "Sizin için iyi! / “Good for you!” " veya "Snap'deki aritmetik işlemleri iyi bilmiyorsunuz! (You don’t know well the arithmetic operations in Snap) **Amaç: Öğrenciler değişkenler, rastgele sayılar, döngüler, yayın hakkında önceden edindikleri bilgileri geliştirecekler.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif Öğrenme (tartışma, bir önceki oyundan öğrenilenleri deneyimleme), oyun temelli öğrenme, problem çözme
Öğretme Formları	Bireysel çalışma / İkili çalışma/ Tüm sınıf ile ön çalışma
Öğretme Özeti	**(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme)** Öğretmen, Snap'deki aritmetik işlemlerde ustalaşılıp ustalaşılmadığını ve projede gösterilip gösterilmediğini belirlemek için bir oyunun ihtiyacına ilişkin problemi ortaya koyar. https://snap.berkeley.edu/project?user=ddureva&project=operations3

1. Öğretmen, görevin koşulunun nasıl formüle edileceğini tartışır. Görev formüle edilmiştir.
   Rastgele bir şekilde on kez birinci sayının 6 olduğu bir aritmetik işlem seçilir. Ayrıca 1'den 3'e kadar sayılardan rasgele bir şekilde ikinci sayı seçilir. Oyuncu doğru cevabı girmelidir. Doğru ve yanlış cevaplar sayılır. Sonuç, oyunun sonunda bildirilir.
2. Değişkenler, değişkenlerin tanımlanması, başlatılması ve değiştirilme şekilleri öğrenciler ile beraber yorumlanır.
3. Rastgele sayı komutları, aritmetik ve mantık işlemleri, olay (event) yayın komutları öğrenciler ile revize edilir.
4. Temel kodun sahne mi yoksa karakter mi olduğu tartışılır. Örnekte, ana kod sahne içindir ve hareketli karakterin kodunda kostümü değiştirmek ve nihai sonucu hesaplamak için komut dosyaları vardır.

5. Operasyon/işlemi seçmek için aşağıdaki komutlar kullanılır:

Sahne için kod	Sahne/sahne dekor/



Sahne kodu, doğru ve yanlış cevap değişkenleri için başlatmaları (initializations) içerir.

Bir işlemi seçmek için aşağıdaki komutlar kullanılır:

Karakter için kostüm seçimi, Sayı karakterine yayın kullanılarak (broadcast) yapılır. Seçilen kostüm numarası, projedeki tüm nesneler için tanımlanan Kostüm Numarası (Costume Number) değişkeninde saklanır ve bu nedenle sahne kodunda kullanılabilir.

Sahne / sahne dekoru / ve karakter kostümü rastgele seçildikten sonra, oyuncuya aşağıdaki komutla işlem için doğru yanıtı girmesi için bir soru sorulur.

Girilen yanıt, seçilen işlemlerin sonucu ile karşılaştırılır.

Aşağıdaki komut kullanılır:

Eğer (Koşul) if (conditional)

başka6 / (else6 /)

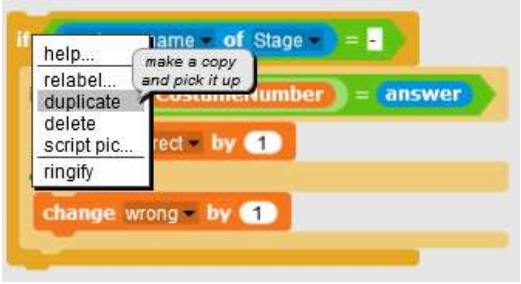
"-" işlemi seçilirse, *6 - "Karakterin kostüm numarası"(6 - "Sprite's costume number)* sonucunun cevapla eşleşip eşleşmediği kontrol edilir. Eğer eşleşirlerse, doğru değişken artar, aksi takdirde yanlış cevapların sayısı için değişken artar.

Komutların geri kalanı için komut dosyası (script) benzerdir. Ancak seçilen işlemler farklıdır.

İşlemlerin geri kalanı için tekrarlanan kod sıralamasından kaçınmak için öğrencilere kodun bir bölümünü nasıl kopyalayacakları ve  'da yer alan aritmetik işlemi nasıl değiştirecekleri öğretilebilir.

Kod kopyalama:

1. Farenin sağ tuşu ile komut dosyasına tıklayın.
2. Çoğalt (duplicate)'ı tıklayın.



3. Kopyalanan komut dosyasını ilgili konuma yerleştirmek için fareyi kullanın.

Öğretmenin takdirine bağlı olarak, öğrenciler kodun bir kısmını kendilerinin nasıl kopyalayacaklarını bulmakla görevlendirilebilir.

İşlemleri değiştirmek.

1. İşlem işaretini farenin sağ tuşuyla tıklayın. Context (Bağlam) menüsü görünecektir.

2. Yeniden etikelendir (relabel)'i seçin, işlemler (operations) görünecektir.

3. İşlemleri seçin.

Not: Öğrencilerin yaşı ve aritmetik işlem bilgisi izin veriyorsa, görev işlemler, derecelendirme (^) ve modüle göre bölme (mod) ile genişletilebilir.

	6. Öğrenciler, karakter için kendi sahnelerini / sahne dekorlarını / ve kostümlerini oluşturan takımlar halinde çalışırlar. Zaman kısıtlamaları varsa, sahne ve hareketli grafiği içeren "yarı destekli" bir proje kullanılabilir.
Öğretmenler İçin Araçlar ve Kaynaklar	Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=ddureva&project=operations3 Scratch'teki tüm aktivite ● Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia)
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki yarı bitmiş aktivite https://snap.berkeley.edu/project?user=ddureva&project=operations_half ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 18 – Geri dönüşüm

Öğrenme Senaryosu Adı	Geri dönüşüm
Geçmiş Programlama Deneyimi	Bir karakteri gösterme ve gizleme Noktaları saymak için değişkenleri kullanma Sonsuza kadar döngüsünü kullanma Koşul ifadeleri kullanma Karşılaştırma için işlemleri kullanma Renk algılama özelliğini kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullu ifadeler ● Döngü ● Bir yöne dönme ● Algılama blokları ● Kodları yeniden düzenleme Algoritmik düşünceye odaklı özel öğrenme çıktıları: • Öğrenciler, oyunu bitirmek için bekleme ve mantıksal işlemlerini kullanır. • Öğrenciler, sahneyi değiştirmek için bekle ve engelle komutunu kullanır. • Öğrenciler, puanları saymak için değişkenleri kullanır. • Öğrenciler, koşul ve mantıksal işlemleri kullanır. • Öğrenciler, benzer karakterlerin kodlarını karşılaştırırlar. • Öğrenciler, kodları yeniden düzenlerler. • Öğrenciler, karakter konumlandırmayı kullanır (ek bir görevde rastgele konumlandırma kullanır).
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Tanım:** Birisi okulun önüne çöp attı. Oyuncudan, çöp toplamaya kâğıt ve camı geri dönüşüm için ayırarak yardım etmesi istenir. Çöp doğru konteynere yerleştirildiğinde çöp gizlenir. Çöp yanlış konteynere

	yerleştirilirse "Bu bir kağıt konteyneri değil" veya "Bu bir cam konteyneri değil" mesajları belirir ve çöp orijinal konumuna geri döner. Tüm çöpler doğru konteynerlere konduğunda oyun sona erer. **Görev:** Öğrenciler, sahne ve karakterlerin kodlarını keşfetmeli. Atık kağıt ve atık cam şeklindeki karakterlerin kodlarını karşılaştırmalı. Yeni karakterler ile kodları eklemeli ve sahnede yeni karakterlere göre kodları değiştirmeli. Ek görevler şunlar olabilir: ● Karakterlerin rastgele koordinat seçimi ile atık karakter pozisyonunu değiştirmek. ● Sahne sayısını azaltmak ve robotu ayrı bir karakter olarak çıkartmak (Robot, sahne arka planının bir parçasıdır). **Amaç: Öğrenciler önceden edindikleri bilgileri geliştirecekler ve oyun senaryosunu yeni nesneler, kodlar ve yeni karakterlere göre değişen kodlarla genişletecekler. Ayrıca öğrenciler kodları yeniden düzenleme konusunda eğitileceklerdir.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme (tartışma, bir önceki hazırlanan oyun ile deneyimleme), oyun temelli öğrenme, problem çözme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma/ Tüm Sınıfla Ön Çalışma

Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) 1. Öğretmen, çöpleri ayırma problemini ortaya koyar ve farklı çöp türleri için kutuların renkleri hakkında yorum yapar (kâğıt için mavi, plastik için yeşil). 2. Öğrencileri oyunu oynamaya ve kelimelerle açıklamaya yönlendirir. Kaç sahne izliyorlar ve kaç tane karakter var? Oyun nasıl başlıyor? Hangi karakter oyuncunun adını sorar? Kaç değişken kullanılır ve bunlar nasıl adlandırılır? Kâğıt, bir cam konteynerine konulduğunda ve kâğıt konteynerine konulduğunda ne olur?  1. Çalışılan komutların güncellenmesi Kullanıcı ile diyaloğa girme komutları geri çağrılır. Değişen sahneler hakkında bir yorum yapılır. (Robot ile Sahne 1, okul ve çöp ile Sahne 2 ve Robot ile Sahne 3 ve Bravo!) Yazısı. 

Çöpün bir konteynere uygun şekilde yerleştirildiğinin kontrolünün, bir koşullu blok ve sensing grubundan dokunma koşullarına sahip bir blokla yapılması gerektiği tartışılmaktadır. Sözlü bir açıklama yapılır. Kâğıt konteynerine bir parça kâğıt çöp değerse çöp gizlenerek (doğru çöp kutusuna yerleştirilir) toplanan kâğıt atığının puanları 1 artar. Cam konteynerine bir parça kâğıt çöp değerse, konteyner "-" Bu bir kâğıt konteyneri değil" der. Aynı şey cam çöp için de geçerlidir.

6. Sahne ve karakter kodlarının incelenmesi.

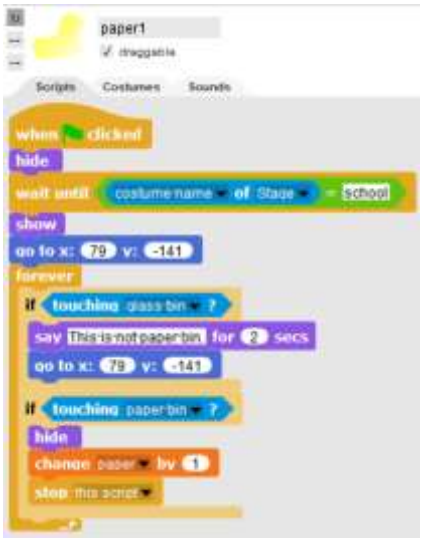
Problemi çözme olasılıkları tartışıldıktan sonra sahne ve karakterler için kodlar tartışılır.

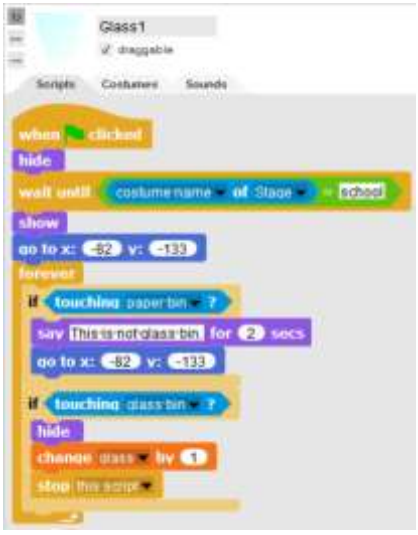
Sahne kodu şu hususlara vurgu yapılarak yorumlanmıştır:

- İsim değişkeninin başlangıç değerinin ayarlanması ve kullanıcı ile diyalog içerisinde kullanılması.
- Sahne dekorunu (kostümleri) ve oyunu bitirme koşulunu

değiştirmek.

Karakter kodlarına bakarken, bunların tek bir slaytta gösterilmesi veya basılı hurda kâğıt ve hurda cam parçalarının her birine iki kod verilmesi tavsiye edilir. Kodlardaki ortak ve farklı unsurlar arasında karşılaştırma yapılır.





7. Oyunu iki yeni karakter (kâğıt çöp ve cam çöp) ile tamamlamak için bir görev verilmesi. Bu karakterlere bir kod atama ve sahne ile çöp kutusu kodlarını değiştirilmesi.

İki yeni karakterin nasıl oluşturulacağı tartışılır. Seçenekler:

1- Mevcut olanları çoğaltın ve Snap!'de düzenleyin.

2- Bir grafik düzenleyicide yenilerini oluşturun veya internette ücretsiz dağıtılan görüntüleri arayıp oyuna aktarın.

Ayrıca oyunun tamamlanmasıyla ilgili sahne kodundaki değişiklikler hakkında yorum yapmak gerekir.

Değişkenlerin başlangıç değerlerinin iki konteynerin kodunda değil, sahne kodunda ayarlanması ve buna göre bir ayarlamanın mümkün olup olmadığı da tartışılmalıdır.

Öğretmenin takdirine bağlı olarak görevin durumu karmaşık hale getirilebilir.

- Oyun başlarken çöpler uygun bir yere bırakılmalıdır. Burada çöpün dağıtılabileceği koordinatların gerçek görünmesi için sınırlı bir alanın seçilmesi gerektiğini not etmeliyiz. Örneğin, çöplerin alanı kırmızı dikdörtgenin koordinatları ile sınırlanmıştır.

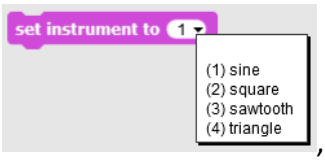
- Yeni bir Robot karakteri tanıtın ve sahnedeki öğelerin sayısını azaltın. Mavi bir konteyner karakteri yerine oyuncu ile diyaloğa girmesi için uygun kodu Robota yazın.

Öğretmenler İçin Araçlar ve Kaynaklar	Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=ddureva&project=recycling Scratch'teki tüm aktivite ● Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia)
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki yarım tamamlanmış aktivite https://snap.berkeley.edu/project?user=ddureva&project=recycling ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

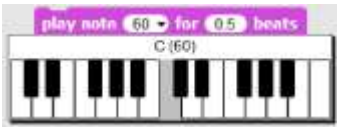
Öğrenme Senaryosu 19.1 - Piyano çalma

Öğrenme Senaryosu Adı	Piyano çalma
Geçmiş Programlama Deneyimi	Noktaları saymak için değişkenleri kullanma. Düğmesine basıldığında olayını (event *When I am pressed*) kullanma. Tekrar döngüsünü kullanma. Koşul cümlelerini kullanma. Sahne / sahne dekorunu değiştirmek ve hareketli karakterlerin etkinliklerini yönetmek için yayın etkinliklerini kullanma.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler; ● Koşullu ifadeler; ● Döngü; ● Yayın etkinlikleri; ● Sesler; ● Müzik programlama; Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenciler, puan sayımı için değişkenleri kullanır. ● Öğrenciler, puan sayımı için değişkenleri başlatır. ● Öğrenciler, elde edilen puanları tahmin etmek için koşullu ifadeler kullanırlar. ● Öğrenciler, sahne / sahne dekorunu değiştirmek ve karakterlerin hareketleri için yayın etkinliğini kullanır. ● Öğrenciler, melodileri oluşturmak için Ses grubundaki blokları kullanırlar. ● Öğrenciler, komut dosyalarındaki blok sayısını azaltmak için tekrar döngüsü belirler. ● Öğrenciler, oyunun işlevselliğini genişletir.
Amaç, Görevler ve Faaliyetlerin	**Kısa Açıklama:** Kraliçe Mary'nin harika dünyasına girelim. Oyuncuyu müzik dinlemesi için sarayına davet ediyor. Balo salonunda küçük dinozor arkadaşı Dino

Kısa Tanıtımı	piyano çalıyor. Oyunda Dino birkaç müzik tonu çalar ve oyuncular bunun hangi ton/nota olduğunu anlamalıdır. Doğru tahmin ederlerse, doğru cevap için bir puan alırlar. Bilmiyorlarsa yanlış cevap için puanlar azaltılır. Tonları tanımladıktan sonra, daha karmaşık bir görev belirlenir. Dino bir melodi çalar ve oyuncunun hangi şarkı olduğunu tanıması gerekir. Düzgün tanımlanmış bir melodi için oyuncu 5 puan alır. **Görev:** Öğrenciler, sahne / sahne dekoru / ve karakter kostümlerine sahip yarı destekli bir dosya kullanır. Gerekli değişkenleri planlamaları, ihtiyaç duydukları blokları belirlemeleri, ses grubunun bloklarını ve notaları çalmanın yolunu tanıyın. Birkaç melodiyi çalmak için komut dosyaları oluşturun. **Amaç: Öğrenciler melodileri kodlama ve çalma hakkında bilgi edinecek ve değişkenler, döngüler, koşullu, yayın ve diğer olaylar hakkında önceden edindikleri bilgileri geliştireceklerdir.**
Faaliyetin Süresi	90 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme (tartışma, bir önceki hazırlanan oyun ile deneyimleme), oyun temelli öğrenme, problem çözme
Öğretme Formları	Bireysel Çalışma / İkili Çalışma/ Tüm Sınıfla Ön Çalışma

Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) 1. Öğretmen, oyunu oluşturma görevini belirler. Görevin tamamlanabileceği araçlar tartışılır. Bir melodiyi programlamak için mevcut kod yazma kaynaklarından şu anda haberdar olmadıkları sonucuna varılmıştır. 2. Öğretmen bir melodi programlayarak oyunun bir bölümünü gösterir. https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1  3. Öğretmen kodu göstererek ses grubu komutlarının nasıl kullanılabileceğini açıklar. Snap!'teki kitaplıktaki sesler ve bilgisayardaki sesler ile çeşitli enstrümanlardan çıkan sesler kullanılabilir. Aracı (tool) seçmek için komutları kullanın (command):  , Not: Scratch'da çok daha fazlası mevcuttur. Öğrenciler birçok enstrümanın sesini deneyebilirler. 4. Öğretmen müzik notalarının nasıl ayarlanacağını girer: "play note 60 for 0.5 beats" *(Notayı….saniye çal)* komutu kullanılabilir. Burada ilk numara notayı ayarlar ve ikinci numara notanın ne kadar süreyle çalınacağını belirtir.

İlk sayının yanındaki oka tıkladığınızda, bir piyano klavyesi belirir ve ondan bir nota seçilebilir. Bu piyano klavyesi iki oktava yayılır.



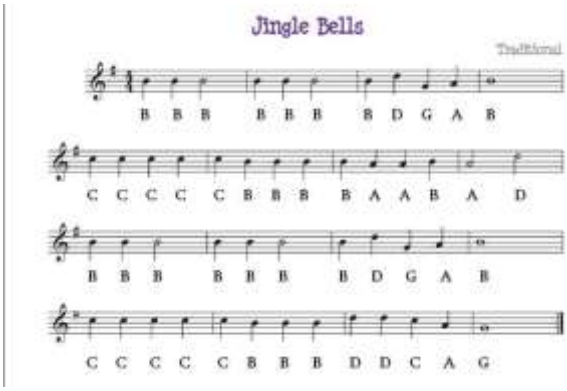
C	C #	D	E b	E	F	F #	G	G #	A	B b	B	C
60	61	62	63	64	65	66	67	68	69	70	71	72

Her bir notanın uzunluğu rakamlar ile belirtilir. 1- tam nota, 05- yarım nota, 0,25 çeyrek nota. (Ondalık sayıları bilmeyen öğrencilere kesirli olarak gösterilebilir : ½, ¼, 1/8 gibi)

,


Öğretmenin takdirine bağlı olarak öğrenciler komutları kendi başlarına deneyebilir veya destek talep edebilir.

5. Jingle Bells (yılbaşı şarkısı) melodi kodları, müzik notaları kullanılarak da tartışılır.



6. Görev, koddaki tekrarlanan satırların sayısını azaltmak olarak belirlenmiştir. Kullanılacak komut (*döngü tekrarlama / repeat loop*) tartışılmıştır. Öğrenciler, dersin başında belirlenen oyunu oluşturmaları gereken takımlara ayrılırlar. Her takım oyun

	senaryosunu tartışır ve açıklama sayfasında oyun planını açıklar (Ekli SNAP_Program_Design_ve_Planning Çalışma Sayfası.docx). Sahnelerdeki ve karakterlerdeki eylemlerin detayları için açıklamaya tablolar eklenebilir. Dinozorun oynarken dans etmesi için bir koşul (condition) eklenebilir (Önceden hazırlanmış dosyada dinozor birkaç kostüme sahiptir). 7. Öğretmen dosyadan senaryoların bazı bölümlerini görüntüleyebilir. https://snap.berkeley.edu/project?user=ddureva&project=PlayAPiano
Öğretmenler İçin Araçlar ve Kaynaklar	Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1 https://snap.berkeley.edu/project?user=ddureva&project=PlayAPiano
Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki yarı tamamlanmış aktivite https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_Half_backed ● Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 19.2 - Piyano çalma

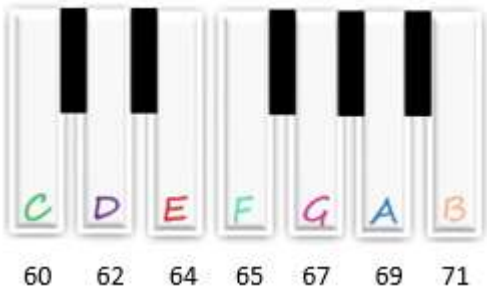
Öğrenme Senaryosu Adı	Piyano çalma
Geçmiş Programlama Deneyimi	Döngü tekrarını kullanma Değişkenleri kullanma Koşul ifadeleri kullanma
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Koşullar ● Döngüler

	Algoritmik düşünceye odaklı özel öğrenme çıktıları: • Öğrenci, müzik çalmak için döngü tekrarını kullanır. • Öğrenci, karakterlerin girdiye tepki vermesini sağlamak için kod kullanır. • Öğrenci, karaktere sesler ekler. • Öğrenci, bir karakterin kostümünü değiştirmek için kod kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa açıklama:** Öğrenci, verilen notlara göre piyanoda şarkı çalmak zorundadır. **Görevler:** Öğrenciler piyano tuşlarını programlamalıdır (her tuşun belirli bir tonu çalması gerekir). Sahnede, biri notaları görüntülemek ve diğeri melodiyi çalmak için iki farklı düğme gösterilmelidir. **Amaç: Öğrencilerin bir karaktere tıklayarak müzik çalmayı ve kostümü değiştirmeyi öğrenmeleridir.**
Faaliyetin Süresi	45 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Bireysel öğrenme, oyun temelli öğrenme, problem çözme
Öğretme Formları	Aktif Çalışma / İkili Çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve Değerlendirme) Başlangıçta sahnede bir piyano gösterilir. Piyanonun yanında iki düğme olmalıdır. İlk düğmeye tıklandığında şarkının notaları ve sözcükleri görüntülenmeli ve ikinci düğmeye tıklandığında tekrarlanması gereken melodi çalınmalıdır. Ek olarak, piyanonun yanında projeyi yeniden başlatacak "X" düğmesi bulunmalıdır. [Adım 1] Piyano çalma (Play a piano) programını açın. Program, bu görev için

gereken tüm arka planları ve karakterleri içerir.

Arka plan ile C tuşu ve bir siyah anahtar için karakter (sprite) verilir.

Öğrencilerin C anahtarını kopyalaması, doğru konuma getirmesi ve yeniden adlandırması gerekir. Tuşlar şu sırada olmalıdır: C, D, E, F, G, A, B. Klavye aşağıdaki resimdeki gibi görünmeli ve tuşların altında yazılı tonları yeniden üretmelidir.



5 siyah anahtar almak için “black_key (siyah anahtar)” karakterini 4 kez çoğaltın ve bunları siyah anahtar 2’den siyah anahtar 5’e kadar adlandırın. D ve E, F ve G, G ve A ve A ve B tuşlarının arasına yeni siyah anahtarlar yerleştirin.

Eğer siyah anahtar beyaz anahtarın arkasında gizli isi şu kodu kullanın:

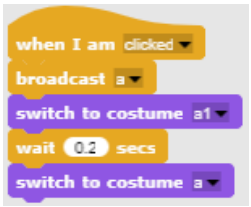
B tuşu için de aynısını yaparak bunları klavyenin sonuna yerleştirin. "Sürüklenebilir/draggable" düğmesinin işaretini kaldırın. Böylece tuş karakter oynatma sırasında taşınamaz.

[Adım 2]

Karakter tuşlarına basarak sesleri çalmayı etkinleştirin. "C" anahtarı için "*Tıklandığında (when I am clicked)"* şapka bloğunu ekleyin ve "C" mesajını yayınlamasına izin verin.

Bir tuşa basıldığında ses üretmek için, "*C mesajı aldığımda/When I receive c"* şapka bloğu ekleyin ve ardından 0.5 vuruş için 60 çalma notası ekleyin.

Hangi tuşa basıldığını vurgulamak için o karakterin kostümü geçici olarak değiştirilmelidir. C1 kostümünü C karakterine aktarın. "Tıklandığımda (When I am clicked)" bloğunda, kostümü 0,2 saniyeliğine C1 olarak değiştirin. Sonra C kostümüne dönün.

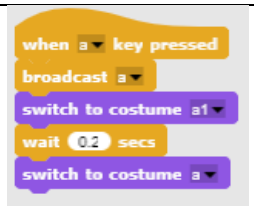


[Adım 3]

İkinci adımı tüm beyaz anahtarlar için tekrarlayın.

[Adım 4]

Klavyeyi kullanarak piyano çalmak için c tuş karakterine *"c tuşuna basıldığında - When c key pressed"* bloğu ekleyin ve "*Tıklandığında - When I am clicked*" bloğundan kodun geri kalanını kopyalayın.



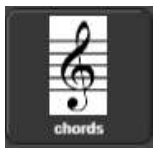
Klavye üzerindeki c tuşu basılı tutulursa tuşa basıldığı sürece sesin tekrarlanacağına dikkat edin. Bunun nedeni "a" mesajının tekrar tekrar yayınlanmasıdır. Bir mesajı yayınlamayı durdurmak için kodun sonuna Kontrol paletinden bir "*… e kadar bekle - wait until*" bloğu ekleyin.

Bir mesajı yayınlamayı bitirmek için, "*değil" "not*" operatörünü kullanın ve "a *düğmesine basıldı- key a pressed" bloğuna ekleyin.*

Beyaz anahtarlar için de aynısını yapın.

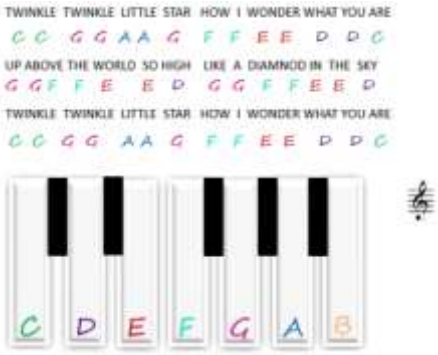
[Adım 5]

Yeni bir karakter oluşturun ve bir keman anahtarının resmini kostüm olarak içe aktarın. Bu, çalınacak kelimeleri ve notaları görüntülemek için bir düğme olacaktır.



Notaları görüntülemek için düğmeye tıklandığında "*akorlar- chords*" mesajının yayınını etkinleştirin.

Sahne için yeni bir "*akorları- chords*" kostümünü içeri aktarın.



Sahnenin bir "akor- chords" mesajı aldığında kostümü "akorlara- chords" çevirmesini sağlayan bir kod ekleyin.

[Adım 6]

Kostüm olarak nota şeklinde bir karakter bulun. Bu, tekrarlanması gereken şarkıyı çalmak için bir düğme olacaktır.

Kod, şarkının ilk iki dizesi için yazılmış olup diğer dizelerin kodunu yazmanız gerekmektedir. Bu kod müziklerde görünen şarkının aynısıdır.

[Adım 7]

Projeyi sıfırlayacak yeni bir X düğmesi oluşturun (notalar olmadan).

Yeni bir karakter oluşturun – sıfırlayın. "X" kostümünü seçin ve

boyutunu% 50'ye ayarlayın. Düğmeye basıldığında "*boş- blank* " mesajının yayınını etkinleştirin.

"Boş-blank" mesajını aldıktan sonra kostümü "boş-blank" olarak değiştirmek için sahneye bir "Aldığımda - When I receive" şapka bloğu ekleyin.

[Ek görevler]

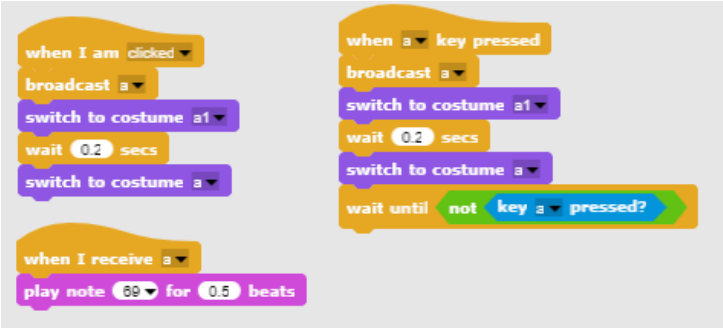
Öğrenciler isteklerine göre ek görevler ekleyebilir veya aşağıdaki görevleri takip edebilirler.

● Nota karakterini çoğaltın (ve arka planda konumunu değiştirin) ve başka bir şarkı için bir program yazın.

● Yeni şarkı için akorları olan bir arka plan ekleyin.

[Nihai Kod]

Anahtar



Keman anahtarı

Nota


when I am clicked
repeat 2
play note 60 for 0.5 beats
repeat 2
play note 67 for 0.5 beats
repeat 2
play note 69 for 0.5 beats
play note 67 for 0.5 beats
wait 0.1 secs
repeat 2
play note 65 for 0.5 beats
repeat 2
play note 64 for 0.5 beats
repeat 2
play note 62 for 0.5 beats
play note 60 for 0.5 beats
wait 0.1 secs
repeat 2
play note 67 for 0.5 beats
play note 67 for 0.5 beats
play note 65 for 0.5 beats
play note 65 for 0.5 beats
play note 64 for 0.5 beats
play note 64 for 0.5 beats
play note 62 for 0.5 beats
wait 0.1 secs
repeat 2
play note 60 for 0.5 beats
repeat 2
play note 67 for 0.5 beats
repeat 2
play note 69 for 0.5 beats
play note 67 for 0.5 beats
wait 0.1 secs
repeat 2
play note 65 for 0.5 beats
repeat 2
play note 64 for 0.5 beats
repeat 2
play note 62 for 0.5 beats
play note 60 for 0.5 beats

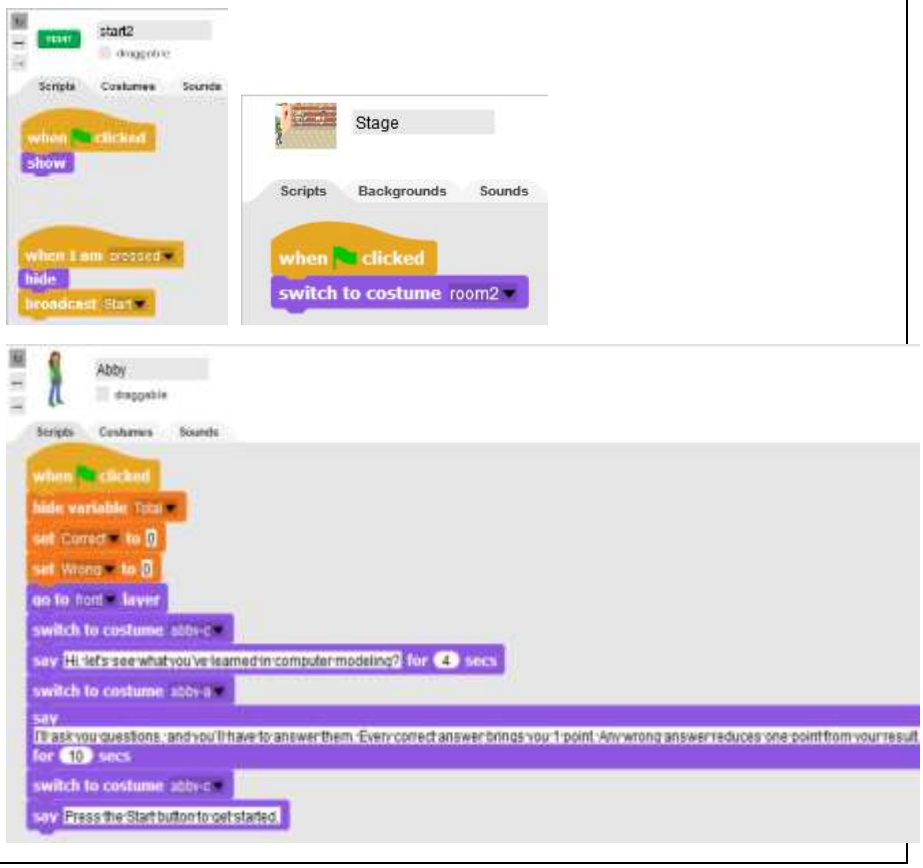
	X  Sahne 
Öğretmenler İçin Araçlar ve Kaynaklar	Snap! – Piyano çalma projesi: https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20a%20Piano

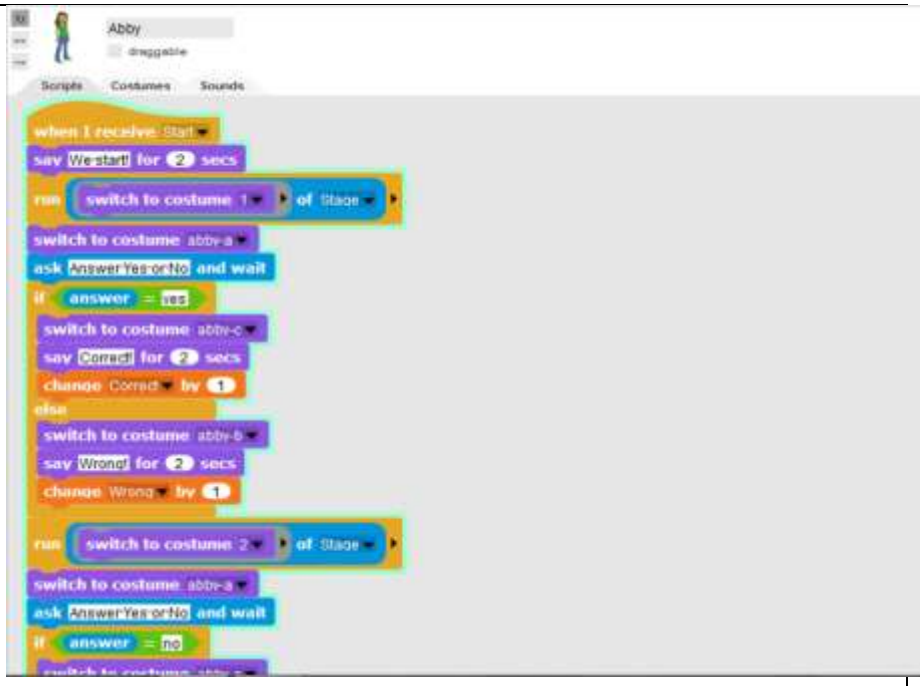
Öğrenciler İçin Kaynaklar ve Materyaller	Snap!'teki yarı tamamlanmış aktivite https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20Piano (27.1.2020)
	Resimler:
	• Karakter resimleri:
	  ◦ a.png, a1.png
	  ◦ b.png, b1.png
	  ◦ violin_key.png
	• Arka planlar: notes.png

Öğrenme Senaryosu 20 - Test

Öğrenme Senaryosu Adı	Test
Geçmiş Programlama Deneyimi	Karakter gösterme ve gizleme Noktaları saymak için değişkenleri kullanma Döngüyü sonsuza kadar kullanma Koşul ifadeleri kullanma Karşılaştırma için işlemleri (operator) kullanma Renk algılama özelliğini kullanma Sahneyi değiştirme
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Değişkenler ● Koşullu ifadeler ● Döngü ● Algılama blokları Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenciler, cevabı tahmin etmek için koşullu ifadeler kullanır - Doğru veya Yanlış. ● Öğrenciler, sahnenin kostüm değişikliği için bloklar kullanır. ● Öğrenciler, puan sayımı için değişkenleri kullanır, ● Öğrenciler, mantıksal işlemleri kullanır. ● Öğrenciler, aşamaların karmaşık arka planlarını hazırlamak için harici grafik düzenleyici kullanır.
Amaç, Görevler ve Faaliyetlerin Kısa Tanıtımı	**Kısa Açıklama:** Öğretmeninizin Snap'de kullanılan komutları test etmek için Quest Tabanlı Oyun oluşturarak Snap! Bilginizi ölçmesine yardım edin. **Görev:** Öğrenciler örnek oyunu keşfetmeli, "yarı destekli" oyundan seçim yapmalı, soruları belirleyecek kendi karakterlerini bulmalı veya tasarlamalı, "yarı destekli" oyundan seçim yapmalı veya ilk aşama arka planını ve uygun sorularla sahne arka planlarını tasarlamalıdır.

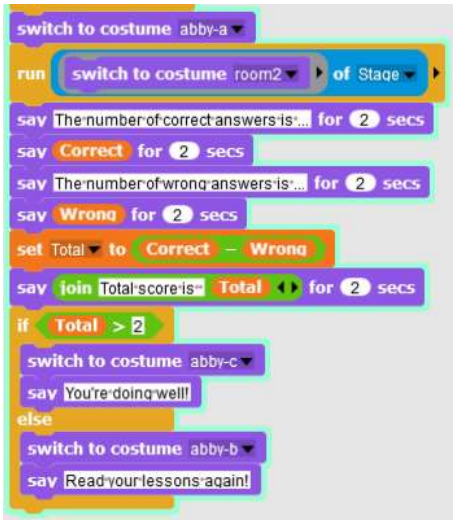
	Amaç: Öğrenciler önceden edindikleri bilgileri geliştirecekler ve oyun senaryosunu yeni aşamalara göre yeni arka plan, kod ve değişen kodlarla genişletecekler.
Faaliyetin Süresi	90 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme (tartışma, bir önceki oyundan öğrenilenleri deneyimleme), oyun temelli öğrenme, problem çözme.
Öğretme Formları	Bireysel çalışma / İkili çalışma/ Tüm sınıfla ön çalışma
Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) 1. Öğretmen, programlama bilgisini test etmek için bir oyun testi oluşturma ihtiyacını ortaya koyar. 2. Öğrencileri oyunu oynamaya ve kelimelerle açıklamaya yönlendirir. Kaç sahne dekorasyonu ve karakter gözlemliyorlar? Oyun nasıl başlıyor? Kaç değişken kullanılır? Nasıl adlandırılır? Ne için kullanılır? Cevap doğru / yanlış olduğunda ne olur? Testte sorular nasıl sunulur? 3. Soru sormak ve cevaplamak için kullanılan algoritma hakkında yorum yapın / • Bir sahne kostümüne geçme (soruyu içerir). • Abby'ye soru sorması için bir kostüm atamak. • Abby der ki! - Evet veya Hayır olarak yanıtlayın. • Oyuncu bir cevap girer - Evet veya Hayır. • Cevap doğruysa, Abby "Doğru" der ve doğru cevapların sayısı artar. Aksi takdirde Abby "Yanılıyorsun" der ve yanlış cevapların sayısı artar.

	4. Tüm soruları yanıtladıktan sonra ne olacağı hakkında yorum yapın. • Sahnede kostüm / arka plan değişikliği. • Abbey, doğru ve yanlış cevapların sayısını gösterir ve bir tahmin verir. 5. Oyundaki kodları test etme "Kullanıcı ile diyaloğa grime", "sahne dekorunu ve karakter kostümünü değiştirme komutları" ve "koşullu komutlar" yorumlanır. Her karakterin kodları incelenir. Bir değişken oluşturma üzerine yorum yapılır. 



Doğru cevabın EVET olduğu ve doğru cevabın HAYIR olduğu durumlar yorumlanır.

Not verme kodu ve toplam değişkeninin neden kullanıldığı ayrıntılı olarak tartışılır.



Bireysel sorular için sahne dekorunu tasarlamanın yolu tartışılır.

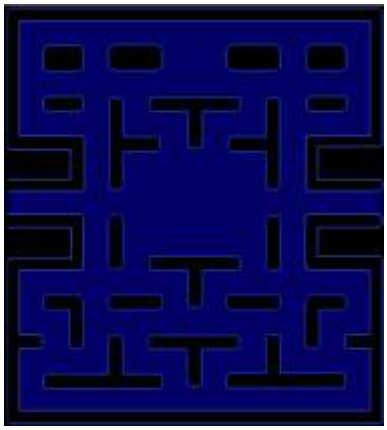
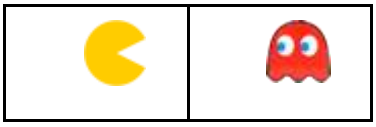
Çünkü Snap'te! Kostümlerde ve sahnelerde yazı yazmak mümkün değil. Harici bir grafik editörü kullanmak gerekmektedir. Diğer bir seçenek de soruyu oluşturmak ve karşılık gelen metin kutusunu grafik

	biçiminde dışa aktarmak için MS Powerpoint kullanmaktır. Snap'teki kostüm ekleme özelliği gözden geçirilebilir. 1. Grubu 2 veya 3 öğrenciden oluşan takımlara ayırmak. 2. Test soruları için konunun yayınlanması. Örneğin - Değişkenleri Kullanma; Döngüler; Hareket, Algılama, Aritmetik ve Mantıksal İşlemler. 3. İlgili ekip tarafından bir konu üzerine sorular içeren sahnelerin tasarlanması. Gerekirse öğretmen öğrencilere soruların içeriği hakkında tavsiyelerde bulunur. Sorular tartışılır ve her ekip en az iki soru için bir sahne oluşturur. 4. Kodu oluşturma. Öğrencilerin kullanması için yarı tamamlanmış sahne ve karakter kostümlerinden oluşan bir dosya verilir. İsterlerse kendilerine ait bir dosya da oluşturabilirler. Çalışma, model testine benzetilerek yapılır.
Öğretmenler İçin Araçlar ve Kaynaklar	Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=ddureva&project=test2 Scratch'teki tüm aktivite ● Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia)
Öğrenciler İçin Kaynaklar ve Materyaller	Öğrenci için talimatlar (C4G2_InstructionsForStudent.docx)

Öğrenme Senaryosu 21 - Basitleştirilmiş PACMAN oyunu

Öğrenme Senaryosu Adı	Basitleştirilmiş PACMAN oyunu
Geçmiş Programlama Deneyimi	● Koşullu ifadeler. ● Birden çok nesneyi kodlama. ● Tek renk algılama. ● Döngüler (sonsuza kadar, şu ana kadar tekrarlayın). ● Olaya dayalı nesne hareketi. ● Rastgele sayılar.
Öğrenme Çıktıları	Genel Öğrenme Çıktıları: ● Bir nesneyi klonlamak. ● Bir klonun davranışını tanımlama. ● Yayın mesajları. ● Mantıksal ifadelerde Boolean değer okumaları. ● İki farklı oyun durumunu tanımlama, ayırt etme, dinamik olarak kontrol etme ve bunlara yanıt verme. Algoritmik düşünceye odaklı özel öğrenme çıktıları: ● Öğrenci, olayları kullanarak ok tuşlarıyla nesne hareketini gerçekleştirir ve kısıtlamaları dikkate alır. ● Öğrenci, orijinal nesnenin örneklerini oluşturmak için klonlar kullanır. ● Öğrenci her klonun bir davranışını nasıl kodlayacağını bilir. ● öğrenciler mesaj göndermenin anlamını bilir. ● Öğrenci klondan artırma sayacına bir mesaj göndermeyi uygular. ● Öğrenci, nesne tarafından alınan mesajın nasıl tespit edileceğini bilir ve uygun bir yanıt verir.
Amaç, Görevler ve Faaliyetlerin Kısa	**Kısa açıklama:** Ana karakterin rastgele konumlandırılmış yıldızları toplayıp bir hayalet tarafından kovalandığı oyunu programlayın.

Tanıtımı	**Görevler:** Öğrenciler, ana karakteri bir labirentin içinde hareket edecek şekilde programlamalıdır. Ana karakterin duvarlardan geçmemesi için hareket kısıtlamaları uygulamalılar. Daha sonra, oyun başladığında kendini klonlayacak bir yıldız nesnesi programlamaları ve ardından karakterin yıldızı her topladığında rastgele yeni bir konumda belirecek şekilde programlamaları gerekiyor. Toplanan yıldızların değerini saklamalı ve oyuncu 20 yıldız topladığında oyunu bitirmelidir. Oyunu daha ilginç hale getirmek için labirentte rastgele hareket edecek kötü bir hayalet programlayabilirler. Bir oyuncu hayalete dokunursa oyun biter. **Bu aktivite ile öğrenciler, önceki aktivitelerde öğrendikleri duyu renk bloğunu kullanarak bir labirent içindeki hareket hakkındaki bilgilerini gözden geçirecekler. Nesneyi konum kısıtlamaları ile klonlama kavramına ve kendi rastgele hareketiyle çok basit bir oyuncu olmayan karakterin nasıl yaratılacağına tanıtılacaklar.**
Faaliyetin Süresi	90 dakika
Öğrenme ve Öğretme Strateji ve Metotları	Aktif öğrenme, işbirlikçi öğrenme, problem çözme
Öğretme Formları	Ön öğretme Bireysel çalışma/ ikili çalışma/ grup çalışması

Öğretme Özeti	(Motivasyon-Giriş, Uygulama, Düşünme ve değerlendirme) Oyuncu, kırmızı bir hayalet tarafından kovalanırken rastgele yerleştirilmiş yıldızları toplar. Bir oyuncu ve hayalet çarpışırsa oyun biter. Bir oyuncu 20 yıldızı toplarsa kazanır. [Adım 1] Öğrencilere, oyuncunun hareket etmesine izin verilen alanın tek renkli (örneğin mavi) olduğu bir labirent ve oyuncu hareketini durduran başka bir renkte (örneğin siyah) renklendirilmiş duvarlar tasarlamaları talimatını veriyoruz. Zaman kazanmak için labirentin arka plan resmini önceden hazırlayabiliriz.  [Adım 2] Öğrencilerin Pacman ve kırmızı hayaleti çizmeleri gerekiyor. Yıldız çizmek için Snap! içine bir daire çizebiliriz.  [Adım 3] Pacman'i hareket ettirmek için farklı olasılıklar kullanabiliriz. Aşağıdaki örnek bunlardan biridir. İçinde hangi tuşa, sola, sağa, yukarı

veya aşağı basıldığını tespit etmek için bir event (olay) sistemi kullanıyoruz. Bu olayların her biri gerçekleştikten sonra, hareket etmesine izin verilen alanın rengine dokunup dokunmadığını test etmeliyiz. Doğru renge dokunduysa önce o yöne döner ve hamleyi yapar. Ancak duvarların rengine dokunursa geri hareket etmesi gerekir. Zira ilk belirlediğimiz koşul nedeniyle duvara sıkışıp kalacaktır.

[Adım 4]

Sonraki görev yıldızları programlamaktır. Birçok yıldız olacak ancak hepsi aynı olmalı. Bu durumda, birden çok özdeş nesne yapmaktansa (bizim durumumuzda 20), bir nesne yapıp sonra onun klonlarını oluşturmak daha iyidir. Oyunun başında ilk klon labirentin içinde rastgele görünecek, ardından oyuncu onu topladığında kaybolacak ve farklı rastgele bir yerde yeni bir tane oluşturulacak. Oyunun başında ilk klonu oluşturmak için bu kodu bir Sahne kod alanına koyuyoruz.

Orijinal bir nesneyi gizlemek ve sadece klonları göstermek için bunu programın başında yapmalıyız.

Uygun rastgele konumlar bulmak için belirli kısıtlamalara uymamız gerekir. Bir duvarda bir yıldız yaratılırsa oyuncu ona ulaşamaz. Yani onu oraya yerleştiremeyiz. Burada kullanacağımız strateji:

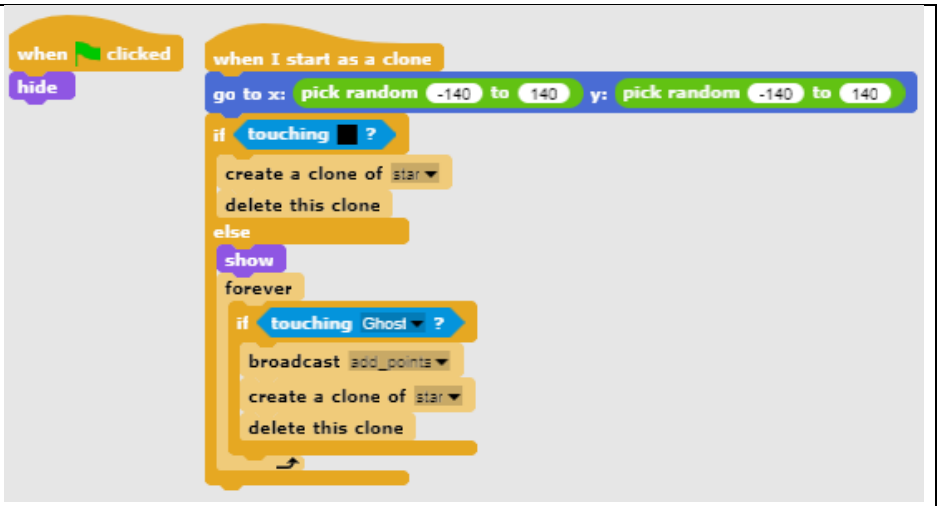
1. Yıldız klonunun rastgele x, y konumunu bulmalıyız. Hem x hem de y koordinatları aynı aralıktadır [-140, 140]. Bu nedenle, her ikisi için de bu aralıktan rastgele bir sayı seçiyoruz.

2. Daha sonra bu klonun duvarın renkli kısmına değip değmediğini kontrol ederiz. Bu durumda konumu uygun değildir.

3. Konum uygunsa, klonu göstermeliyiz (unutmayın, orijinal olan gizlidir ve klon da gösterme bloğunu kullanmazsak gizlenecektir) ve sonsuz döngüde oyuncuyla çarpışma olup olmadığını kontrol etmeliyiz.

4. Konum uygun değilse, yeni bir klon oluştururuz (yeni klon için rastgele sayıların seçileceğini ve böylece uygun bir yere yerleştirileceğini umuyoruz) ve bunu siliyoruz.

5. Toplanan klonları saymak için, klonun dışında (örneğin oynatıcıda) tanımlanması gereken toplam yıldız sayısı bilgisini vermemiz gerekir. Bu, çarpışmanın meydana geldiğine dair bir mesaj yayınlayarak yapılabilir. Sonra onu silebiliriz.



[Adım 5]

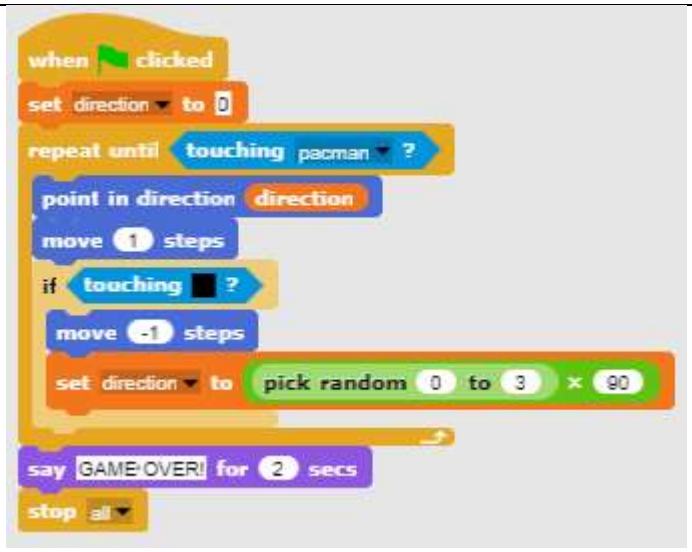
Sonra bir hayalet (ghost) programlıyoruz. Labirentte rastgele hareket etmesi ve duvara çarptığında yönünü değiştirmesi gerekiyor. Hareketini rastgele hale getirmek için, çarpmadan sonra rastgele bir yönde hareket etmesini istiyoruz.

Snap'!te yönler derece ile ifade edilir.

1. 0 derece - YUKARI (UP)
2. 180 derece - AŞAĞI (DOWN)
3. 90 derece - SAĞ (RIGHT)
4. 270 derece – SOL (LEFT)

Diğer bir deyişle, 0'dan 3'e rastgele bir sayı seçip 90 (derece) ile çarparsak, rastgele bir yön elde ederiz.

Bir Pacman ile çarpışana kadar hareket etmesi gerekiyor. Çarpışınca oyun bitecektir.



[Adım 6]

Şimdi oyuncunun oyunu ne zaman kazanacağını programlamalıyız. Oyuncu 20 yıldız topladığı zaman oyunu kazanmış olacaktır. Pacman senaryosunda bir yıldız sayacımız var. Başlangıçta onu 0 olarak başlatırız ve ardından klon, oyuncunun topladığına dair bir mesaj gönderdiğinde sayaç değerini 1 arttırırız. Sayaç 20'ye gelirse Pacman kazanır ve oyunu durdurmamız gerekir.

Öğretmenler İçin Araçlar ve Kaynaklar

- Snap!'teki tüm aktivite https://snap.berkeley.edu/project?user=zapusek&project=pacman_clone
- Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.
- Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.

Öğrenciler İçin Kaynaklar ve Materyaller	● Snap!'teki şablon: https://snap.berkeley.edu/project?user=zapusek&project=pacman_template ● Öğrenciler için talimatlar (C4G2_InstructionsForStudent.docx)

EK-2 BAŞLANGIÇ VE İLERİ SEVİYE SENARYO ÖĞENCİ KODLARI

N.	BAŞLANGIÇ SEVİYE SENARYO ADLARI	KOD
1	**Snap! ara yüzünü tanıyalım** Öğrenciler yeni bir karakter (sprite/kukla) ekler, karaktere bir kostüm ekler, kostümü düzenler ve bunlardan birini siler. Öğrenciler sahne için yeni bir arka plan oluşturur, düzenler ve istenmeyenleri siler.	**başla**
2	**Snap!'i keşfet - Karakteri hareket ettirme** Öğrencilerin Snap! arayüzünü keşfetmelerine ve ilk kodlarını oluşturarak hareket eden ve konuşan bir karakter oluşturmalarına yardımcı olun.	**keşfet**
3	**Sahnede hareket etme** Öğrenciler sahnede karakteri x ve y yönünde hareket ettirmeyi öğrenir, verilen görevleri çözmek için basit bir program programlar, karakteri farklı yöne çevirmeyi öğrenir.	**hareket**
4	**Kostüm değiştirme ve dönüşler** Öğrenci, farklı rotasyon türleri arasında geçiş yaparak bir animasyon yapmak için karakerin kostümünü nasıl değiştireceğini öğrenir.	**dans**
5	**Çiftlikteki sesler** Öğrenciler, oyuncunun belirli tuşlara basarak hayvanların seslerini tanıyabileceği basit bir oyun programlamayı öğrenirler.	**çiftlik**
6	**Bukalemenun yaz tatili** Bir nesnenin kostümünü arka planın rengine göre değiştireceği basit bir oyun programlayın	**bukalemun**
7	**Prens ve Prensese evcil hayvanlarını bulmaları için yardım et** Kız, seyahatleri sırasında hayvanların birbirleriyle karşılaşmasını engelleyerek Prenses'in kedisini bulmasına ve Prens'in köpeğini bulmasına yardım etmelidir.	**prenses**
8	**Tebeşir ile çizim yapmak** Öğrencilere bir kodla farklı şekiller çizme tanıtılacaktır. Kodu kısaltmak ve arka planı değiştirmek için döngü tekrarını kullanmayı öğrenecekler.	**çizim**
9	**Çöpleri toplamak ve parkı temizlemek** Öğrenciler değişkenleri nasıl kullanacaklarını ve bir kod bloğunu veya hatta bütün bir hareketli grafiği nasıl kopyalayacaklarını öğrenecekler.	**çöpler**

10	**Kedileri besleme** Öğrencilere, bir döngü içinde çok değişkenli rastgele değer atama kavramı ve bir döngü dışında yaptığımızdan nasıl farklı olduğu anlatılacaktır. Ayrıca doğru oyuncu girdilerinin nasıl alınacağını, test edileceğini ve sayılacağını da öğrenecekler.	**kediler**
11	**Barınaktaki Kedi Sayısını Tahmin Etmek** Öğrencilere "----e kadar tekrar et" döngüsü ve oyunu durduran koşulu örtük olarak izlemek için koşulun nasıl ayarlanacağı tanıtılacaktır. Ayrıca değişkeni; rastgele bir değer belirlemek, sayaç olarak veya oyuncuların girdisini almak gibi farklı durumlarda nasıl kullnacaklarını da öğrencekler.	**barınak**
N.	**İLERİ DÜZEY SENARYO ADLARI**	**KODLAR**
12	**Sağlıklı yiyeceği yakalamak** Öğrenciler, belirli bir adım atmak için rastgele hareket etmeyi, bir konum seçmeyi ve ayrıca diğer olayları önlemek için değişkenleri ve koşulluları nasıl kullanacaklarını öğreneceklerdir.	**yemek**
13	**Hikaye Anlatma** Öğrenciler hikaye anlatımını nasıl planlayacaklarını, karakterin aktiviteleri ile sahne değişikliklerinin senkronizasyonu için yayın mesajlarının nasıl kullanılacağını öğrenecekler	**hikaye**
14	**İklimi İyileştirmek-Çizim Yapma** Öğrenci, Snap!'te çizim yapmayı, rengi ve kalem kalınlığını değiştirmeyi ve yeni bir olaya neden olan değişkenler ile koşullu ifadeleri nasıl kullanacağını öğrenecek.	**iklim**
15	**Fareyi yakalamak** Öğrencilere çok değişkenli rastgele değer atama kavramı tanıtılacaktır. Operatörleri nasıl kullanacaklarını ve rastgele [x] ila [y] bloğu seçmeyi öğrenecekler.	**fare**
16	**Piknik için yemek almak** Öğrenciler değişkenlerle nasıl çalışılacağını öğrenecekler: farklı başlangıç değerleri belirleme, değişkenlerin değerini karşılaştırmak için koşullu ifadeler kullanma, değişkenlerin değerini değiştirme, sağlıklı yiyecekleri saymak (olmayan) için değişkenler kullanma. Ek olarak, metin eklemeyi, metinleri birleştirmeyi ve Eğer (if) ifadesini tekrarlayacaklar.	**piknik**
17	**İşlemler** Öğrenciler değişkenler, rastgele sayılar, döngüler, yayın hakkındaki bilgilerini geliştireceklerdir.	**işlem**

18	**Geri Dönüşüm** Öğrenciler bilgilerini geliştirecek ve oyun senaryosunu yeni karakterlere göre yeni nesneler, kodlar ve değişen kodlarla genişletecekler. Kod yeniden düzenleme konusunda eğitilecekler	**dönüşüm**
19.1	**Piyano Çalmak** Öğrenciler melodileri kodlama ve çalma hakkında bilgi edinecek ve değişkenler, döngüler, koşullu, yayın ve diğer olaylar hakkındaki bilgilerini geliştireceklerdir.	**Piyano1**
19.2	**Piyano Çalmak** Öğrenciler bir karaktere tıklayarak müzik çalmayı ve kostümü değiştirmeyi öğrenecekler.	**piyano2**
20	**Test- Kostümleri değiştirmek ve Dönüşler** Farklı döndürme türleri arasında geçiş yaparak bir animasyon yapmak için karakterin kostümünü nasıl değiştireceğinizi öğrenin	**test**
21	**Basitleştirilmiş PACMAN oyunu** Öğrenciler duyu renk bloğu kullanarak bir labirent içindeki hareketin nasıl kodlanması gerektiği hakkındaki bilgilerini gözden geçireceklerdir. Nesneyi konum kısıtlamaları ile rastgele hareket edecek şekilde basitçe kodlayacakar ve karakterleri klonlamayı öğrenecekler.	**pacman**

REFERANSLAR

A project of the K-12 Initiative at Stanford University. (2009, 05 06). *Taking Design Thinking to School: Approaches to Integrating the Design Process in K-12 Teaching and Learning.* Retrieved 09 10, 2020, from Stanford Graduate School of Education: https://web.stanford.edu/dept/SUSE/taking-design/presentations/Taking-design-to-school.pdf

Alserri, S., Zin, N., & Wook, T. (2018). Alserri, S. A., Zin, N. A. M., & Wook, T. S. M. T.Gender-based Engagement Model for Serious Games. , , 8(4). . *International Journal on Advanced Science, Engineering and Information Technology, 8*(4), 1350-1357.

Baptista, R., & Vaz de Carvalho, C. (2010). Role Play Gaming and Learning. *Learning Technology, 12*(1).

Combéfis, S., Beresnevičius, G., & Dagiene, V. (2016). Learning Programming through Games and Contests: Overview, Characterisation and Discussion. *Olympiads in Informatics, 10*.

Cooper, S., Dann, W., & Pausch, R. (2000). Alice: A 3-D tool for introductory programming concepts. *Journal of Computing Sciences in Colleges, 15*(5), 107-116.

Doorley, S., Holcomb, S., Klebahn, P., Segovia, K., & Utley, J. (2018). *Design Thinking Bootleg.* Retrieved from https://static1.squarespace.com/static/57c6b79629687fde090a0fdd/t/5b19b2f2aa4a99e99b26b6bb/1528410876119/dschool_bootleg_deck_2018_final_sm+%282%29.pdf

Eurostat. (2020, 09 25). *ICT specialists in employment.* Retrieved 11 12, 2020, from eurostat Statistics Explined: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=ICT_specialists_in_employment#Number_of_ICT_specialists

Feldgen, M., & Clua, O. (2004). Games as a motivation for freshman students to learn programming. *34th ASEE/IEEE Frontiers in Education Conference*, (pp. 1079–1084).

Frankovic, I., Hoic-Bozic , N., & Nacinovic-Prskalo, L. (2018). Serious Games for Learning Programming Concepts. *8th Conference edition The Future of Education.* Florence, Italy.

Garris , R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation and learning. Simulation & Gaming. *An Interdisciplinary Journal of Theory, Practice and Research, 33*(4), 43-56.

Gee, J. P. (2003). *What Video Games Have to Teach Us About Learning and Literacy.* New York: Palgrave/Macmillan.

Gee, J. P. (2007). Are video games good for learning? *Curriculum & Leadership Journal, 5*(1).

Geist, E. (2016). Robots, Programming and Coding, Oh My! *Childhood Education, 92*(4), 298-304. doi:10.1080/00094056.2016.1208008

Georgieva, R., & Tuparova, D. (2019). Educatiomnal Computer Game-Based Environments for Teaching Programming and Developmnt of an Algorithmic Thinking - Comparative AnalysisITHMIC THINKING – COMPARATIVE ANALYSIS. *Mathematics and Education in Mathematics, Forty-eighth Spring Conferenceof the Union of Bulgarian Mathematicians* (pp. 150-156). Union of Bulgarian Mathematicians. Retrieved from http://www.math.bas.bg/smb/2019_PK/tom/pdf/150-156.pdf

Georgieva, R., & Tuparova, D. (2020). Design Thinking - helps in school eucation. *Anniversary International Scientific Conference "Synergeticsand ReflectioninMathematics Education"*, (pp. 341-347). Pamporovo.

Grivokostopoulou, F., Perikos, I., & Hatzilygeroudis, I. (2016). An Educational Game for Teaching Search Algorithms. *Proceedings of the 8th International Conference on Computer Supported Education (CSEDU 2016)* (pp. 129–136). Setubal. Portugal: SCITEPRESS - Science and Technology Publications, Lda.

Gross, B. (2003). The impact of videogames in education. 8(7). *First Monday, 8*(7).

Hijon-Neira, R., Velazquez-Iturbide, A., Pizarro-Romero, C., & Carrico, C. (2015). Serious games for motivating into programming. *Frontiers in Education Confere.*

IDEO Design thinking. (2008, 09 7). *Definitions of design thinking*. Retrieved 11 20, 2020, from https://designthinking.ideo.com/blog/definitions-of-design-thinking

Innovation Starter. (2015, 06 04). *What is design thinking?* Retrieved 11 20, 2020, from http://innovationstarterbox.bg/resources/kakvo-e-dizain-mislene/

Jemmali , C. (2016). May's Journey: A serious game to teach middle and high school girls programming. Worcester Polytechnic Institute. Retrieved from https://digitalcommons.wpi.edu/etd-theses/455

Johnson, C., & all, a. (2016). Game Development for Computer Science Education. *the 2016 ITiCSE Working Group Reports.*

Johnston, R. T., & de Felix, W. (1993). Learning from video games. *Computer in the Schools* , 199-233.

Kafai, Y. (1995). Making game artifacts to facilitate rich and meaningful learning. *Annual Meeting of the American Educational Research Association*, (pp. 1–20). San Francisco.

Karakehayova, M. (2019). Opportunities for Using Design Thinking in School Education. *Education and Development, 3*.

Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A Serious Game for Developing Computational Thinking and Learning Introductory Computer Programming. *Procedia - Social and Behavioral Sciences, 47*, 1991-1999.

Kelleher , C., Pausch, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. *Proc. CHI 2007.*, (pp. 1455-1464).

Luka, I. (2014). Design thinking in pedagogy. 2014, 5,. *The Journal of Education, Culture, and Society, 5*, 63-74.

Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2014). CMX: Implementing an MMORPG for learning programming, 8th European Conference on Games Based Learning. *8th European Conference on Games Based Learning - ECGBL 2014.* Berlin.

Malone, T. (1981). What makes computer games fun? *Byte, 6*(12), 258-277.

Malone, T. W. (1981b). Toward a theory of intrinsically motivating instruction. *Cognitive Science, 4*.

Mathrani, A., Christian, S., & Ponder-Sutton, A. (2016). PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Educational Technology & Society, 19*(2), 5-17.

Meerbaum-Salant , O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239-264. doi:10.1080/08993408.2013.832022

Michael, D. R., & Chen, S. (2006). *Serious Games: Games That Educate, Train, and Inform.* Boston: Course Technology PTR.

Miljanovic, M., & Bradbury, J. (2016). Robot on!: a serious game for improving programming comprehension. *Proceedings of the 5th International Workshop on Games and Software Engineering.* Austin, Texas, USA.

Miljanovic, M., & Bradbury, J. (2018). A Review of Serious Games for Programming. *4th Joint International Conference on Serious Games.* Darmstadt, Germany.

Naiman, L. (2019, 06 10). *Design Thinking as a Strategy for Innovation*. Retrieved 11 21, 2020, from Creativity at work: https://www.creativityatwork.com/design-thinking-strategy-for-innovation/

Nančovska Šerbec, I., Kaučič, B., & Rugelj, J. (2008). Pair programming as a modern method of teaching computer science. *International journal on emerging technologies in learning, 3*, 45-49.

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A., & Lahmine, S. (2015). Learning Basic Programming Concepts by Creating Games with Scratch Programming Environment. , 191, . *Procedia - Social and Behavioral Sciences, 191*, 1479-1482. doi:doi:10.1016/j.sbspro.2015.04.224

Paavola, S., & Hakkarainen, K. (2005). The Knowledge Creation Metaphor – An Emergent Epistemological Approach to Learning.14. *Sci Educ, 14*, 535-546.

Phan, M., Jardina, J., Hoyle, S., & Chaparro, B. (2012). Examining the role of gender in video game usage, preference, and behavior. *Human Factors and Ergonomics Society 51*, (pp. 1496–1500.).

Pivec, M., & Kearney, P. (2007). Games for Learning and Learning from Games. *Informatica, 31*, 419-423.

Pivec, M., Koubek, A., & Dondi, C. (2004). *Guidelines for Game-Based Learning. Lengerich.* Pabst Science Publishers.

Prensky , M. (2001). *Digital Game-Based Learning.* New York: Mc-Graw-Hill.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond , E., Brennan , K., . . . Kafai, Y. (2009). Scratch: Programming for All. *Communication of the ACM, 52*, 60-67.

Rieber , L. P., Smith , L., & Noah, D. (1998). The value of serious play. *Educational Technology, 38*(6), 29-37.

Rugelj, J. (2016). Serious computer games design for active learning in teacher education. (C. Carvalho , P. Escudeiro, & A. Coelho, Eds.) *Serious games, interaction, and simulation : revised selected papers. Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering, 161*, 94-102.

Rugelj, J., & Lapina, M. (2019). Game design based learning of programming. In J. Rugelj, & M. Lapina (Ed.), *Proceedings of the International Scientific Conference Innovative Approaches to the Application of Digital Technologies in Education and Research (SLET-2019), May 20-23, CEUR workshop proceedings. 2494.* Stavropol-Dombay, Russia: Aachen: CEUR-WS.

Shabanah, S., Chen, J., Wechsler, H., Carr, D., & Wegman, E. (2010). Designing Computer Games to Teach Algorithms. *Seventh International Conference on Information Technology: New Generations, ITNG 2010*, (pp. 1119-1126). Las Vegas, NV.

Shute, V. J., & Ke, F. (2012). Games, Learning, and Assessment. In D. Ifenthaler, D. Eseryel, & X. Ge, *Assessment in Game-Based Learning: Foundations, Innovations, and Perspective.* New York, USA: Springer.

Shute, V. J., Rieber, L. P., & Van Eck, R. (2012). Games ... and ... Learning. In R. A. Reiser, & J. V. Dempsey, *Trends and issues in instructional design and technology (3rd ed.).* (pp. 321-332). Boston: Pearson Education.

Sykes, E. (2007). Determining the Effectiveness of the 3D Alice Programming Environment at the Computer Science I Level. *Journal of Educational Computing Research, 36*(2), 223-244. doi:10.2190/J175-Q735-1345-270M

Tuparova, D., Nikolova, E., & Tuparova, E. (2020). Integrated game-based learning in an informatics secondary course: Is there a difference between girls' and boys' achievements? *CSEDU 2020 - Proceedings of the 12th International Conference on Computer Supported Education*, *1*, pp. 702-709. Retrieved 10 12, 2020, from https://www.scitepress.org/Link.aspx?doi=10.5220/0009818407020709

Tuparova, D., Tuparov, G., & Veleva, V. (2019 b). Girls' and boys' viewpoint on educational computer games. *European Conference on Games-based Learning*, (pp. 757–766).

Vahldick, A. (2017). Aperfeiçoamento das Competências de Resolução de Problemas na Aprendizagem Intodutória de Programação de Computadores usando um jogo sério digital. Faculdade de Ciências e Tecnologia da Universidade de Coimbra. Retrieved from http://hdl.handle.net/10316/79528

van Eck, R. (2006). Digital Game-Based Learning: It's Not Just the Digital Natives Who Are Restless. *EDUCAUSE Review, 41*(2).

Vermeulen, L., Van Looy, J., Courtois, C., & De Grove, F. (2011). Girls will be girls: a study into differences in game design preferences across gender and player types. *Under the mask: perspectives on the gamer conference*, (pp. 1-21).

Weintrop, D., & Wilensky, U. (2015). To Block or not to Block, That is the Question: Students' Perceptions of Blocks-based Programming. *Interaction Design and Children*, (pp. 199-208).

Whitton, N. (2009). *Learning with Digital Games: A Practical Guide to Engaging Students in Higher Education.* New York: Routledge.

Whitton, N., & Moseley, A. (2012). *Using Games to Enhance Learning and Teaching: A Beginner's Guide.* New York: Routledge.

Wolz, U., Barnes, T., & Bayliss, J. (2009). Girls do like playing and creating games. *ACM SIGCSE Bulletin, 41*(1), 199–200.

Wolz, U., Maloney, J., & Pulimood, S. (2008). 'Scratch' Your Way to Introductory CS. *SIGCSE Bulletin, 40*, 298-299.

Zapušek, M., & Rugelj, J. (2013). Learning programming with serious games. *EAI Endorsed Trans. on Game-Based Learning, 13*, 1-12.