**O3 – Öğretim Desteği İçeriği**

**ÖĞRETMEN HEDEFLİ OYUN TASARIMINA DAYALI ÖĞRENME SENARYOLARI**

**Belge Verileri**

**Fikri Çıktı:** O3/A1 - Öğretmen hedefli oyun tasarımına dayalı öğrenme senaryoları

**Fikri Çıktı No - Başlık**: O3 – Öğretim Desteği İçeriği

**Fikri Çıktı Lideri:** South-West University “Neofit Rilski” (Bulgaristan)

**Dâhil Olan Ortaklar:** University of Ljubljana (Slovenya), University of Rijeka (Hırvatistan)

**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.

Tüm hakları saklıdır. Kaynağın belirtilmesi kaydıyla ticari amaçlar dışında çoğaltılmasına izin verilir.

Copyright © Coding4Girls, 2018-2020

https://licensebuttons.net/l/by-sa/3.0/88x31.png

## Creative Commons Attribution-ShareAlike 4.0

## International Public License ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

**İÇİNDEKİLER**

[GİRİŞ 5](#_Toc51676559)

[ÖĞRENMME SENARYOLARI 6](#_Toc51676560)

[TEMEL ÖĞRENME SENARYOLARI 8](#_Toc51676561)

[Öğrenme Senaryosu 1 - Snap! ile tanışın - Arayüz 8](#_Toc51676562)

[Öğrenme Senaryosu 2 - Karakterinizi hayata döndürme zamanı 12](#_Toc51676563)

[Öğrenme Senaryosu 3 - Sahnede hareket etmek 16](#_Toc51676564)

[Öğrenme Senaryosu 4 - Kostüm değiştirme ve döndürmeler 22](#_Toc51676565)

[Öğrenme Senaryosu 5 - Çiftliğin sesi 29](#_Toc51676566)

[Öğrenme Senaryosu 6 - Bukalemunun yaz tatili, basit sürüm 37](#_Toc51676567)

[Öğrenme Senaryosu 7 - Prense ve prensese evcil hayvanlarını bulmada yardım etme 47](#_Toc51676568)

[Öğrenme Senaryosu 8 - Tebeşir ile çizim yapmak 54](#_Toc51676569)

[Öğrenme Senaryosu 9 - Çöpleri toplamak ve parkı temizlemek 65](#_Toc51676570)

[Öğrenme Senaryosu 10 - Kedileri beslemek 73](#_Toc51676571)

[Öğrenme Senaryosu 11 - Bir barınaktaki kedi sayısını tahmin etmek 81](#_Toc51676572)

[İleri Seviye Öğrenme Senaryoları 89](#_Toc51676573)

[Öğrenme Senaryosu 12 - Sağlıklı yiyecekler yakalamak 89](#_Toc51676574)

[Öğrenme Senaryosu 13 - Hikâye anlatma 99](#_Toc51676575)

[Öğrenme Senaryosu 14 - Çizim yapma 110](#_Toc51676576)

[Öğrenme Senaryosu 15 - Fareyi yakalamak 118](#_Toc51676577)

[Öğrenme Senaryosu 16 - Piknik için yiyecek satın almak 126](#_Toc51676578)

[Öğrenme Senaryosu 17 – Operasyonlar/İşlemler 135](#_Toc51676579)

[Öğrenme Senaryosu 18 – Geri dönüşüm 142](#_Toc51676580)

[Öğrenme Senaryosu 19.1 - Piyano çalma 148](#_Toc51676581)

[Öğrenme Senaryosu 19.2 - Piyano çalma 152](#_Toc51676582)

[Öğrenme Senaryosu 20 - Test 162](#_Toc51676583)

[Referanslar 174](#_Toc51676584)

# GİRİŞ

Geçen yüzyılın önde gelen psikologları, oyun oynamayı yaşa veya gelişim aşamasına bakılmaksızın yaşam becerilerinin geliştirilmesi için en önemli etkinliklerden biri olarak tanımladılar. Oyun boyunca çocuk, yeni koşullara hızlı bir şekilde adapte olarak değişiklikleri kolaylıkla ele alır. Oyun oynarken, kelimeden temel kavramları keşfeder ve aralarındaki ilk temel ilişkiler kurulur.

Günümüzde oyunlar, bir çocuğun gelişiminin ilk aşamalarında evde ve anaokulunda daha yaygın olarak kullanılmaktadır. Okulda öğrenme, çoğu zaman pasif öğrencilerle öğretmen merkezli bir modelde geleneksel bilgi aktarımına dayanmaktadır. Öte yandan, geçen yüzyılda geliştirilen öğrenme kuramları öğrenci merkezli, probleme dayalı, daha yüksek taksonomik seviyelerde, motivasyonel ve genellikle BİT tarafından desteklenen yeni öğretim ve öğrenim yaklaşımlarını teşvik ettiler.

CODING4GIRLS Projesi "düşük giriş yüksek tavan” yaklaşımından yola çıkarak daha az bilgi gereksinimi olan ve daha ileri düzeydeki öğrenciler için problem çözme zorluklarını sınırlamayan programlama etkinliklerini içerir. Öğrenciler, eksik kod yapı taşlarını ekleyerek veya kendi çözümlerini oluşturarak kısmen tamamlanmış olan çözümleri bitirmeye teşvik edilecektir. Aktiviteler, tek bir programlama konseptine sahip temel olanlardan çoklu programlama konseptleri ile daha ileri seviyeye kadar sırayla planlanır. Proje ekibi, Snap!'de öğrenme etkinlikleri hazırlarken kızların tercih ettiği oyunların belirlenen özellikleri ve gerçek dünya sorunları ile ilgili etkinliklerden yola çıktı.

Hazırlanan öğrenme senaryoları, düşünme öğrenme yöntemlerini ve önerilen ciddi oyunları derslerine entegre etmelerine yardımcı olacaktır. CODING4GIRLS, aktif ve oyun tabanlı öğrenme tasarımına göre hazırlanmıştır. Ayrıca hem kız hem de erkeklerin programlama becerilerini geliştirmek amacıyla üretilecek her bir öğrenme aktivitesi için bilgiler içerir.

Öğrenme senaryolarında aşağıdaki bilgiler mevcuttur:

• İlgili öğrenme faaliyetinin genel eğitim hedefi

• Öğrenme faaliyetinin kapsadığı kavramlar

• Belirli öğrenme hedefleri

• Beklenen öğrenme çıktıları

• CODING4GIRLS oyun tasarımına dayalı öğrenme yaklaşımının adım adım kullanımı

• Geliştirilen bilgiyi değerlendirme yöntemleri

• Sınıf işbirliği bağlamında öğrenciler arasında münazara başlatmak için sorular

Öğrenme faaliyetlerine karşılık gelen 21 öğrenme senaryosu hazırlanmıştır. Öğretmenler senaryoları ve oyunları önerilen sırayla kullanabilir veya tercihlerine ve ihtiyaçlarına göre özgürce seçebilirler. Öğrenme senaryoları, kullanıcı etkileşim süreçleri, geri bildirim oluşturma dâhil önerilen ciddi oyunun genel işlevselliğini ve önerilen ciddi oyunda uygulanacak tüm öğrenme etkinliklerinin açıklamalarını içerir.

Öğrenme senaryoları İngilizcenin yanı sıra proje ortaklarının ulusal dillerinde de (Türkçe, Bulgarca, Hırvatça, Yunanca, İtalyanca, Portekizce ve Slovence) mevcuttur.

# ÖĞRENME SENARYOLARI

Hazırlanan öğrenme senaryoları, temel bir programlama konseptinden çoklu programlama konseptiyle daha ileri seviyeye kadar devam eder. Aşağıdaki tablo önerilen faaliyet sırasını göstermektedir.

|  |  |  |  |
| --- | --- | --- | --- |
| **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** | SWU |
| 14 | **Çizim yapma** | UNIRI |
| 15 | **Fareyi yakalamak**  Döngüler, koşullu ifadeler, değişkenler kullanma | UL |
| 16 | **Piknik için yiyecek satın almak**  Değişkenler, koşul ifadeleri, 4 işlemi kullanma | UL |
| 17 | **Operasyonlar** | SWU |
| 18 | **Geri dönüşüm** | SWU |
| 19.1 | **Piyano çalma 1** | SWU |
| 19.2 | **Piyano çalma 2** | UNIRI |
| 20 | **Test** | SWU |
| 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 Metotları** | Öğretmen Gösterimi  Bireysel Çalışma |
| **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;  C:\Users\cerarsp\Downloads\C4G_dog_goes_home script pic(2).png  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.  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.   1. Buradan hangi yöne doğru dönmesi gerektiğini söyleyebilirsiniz. 2. 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=spelac&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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 12.26.19.png  [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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 12.24.23.png    [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.  /Users/tadejanemanic/Downloads/farmer.png  [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.  /Users/tadejanemanic/Downloads/Farm script pic.png  **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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-12 at 17.44.15.png  Akabinde çocuklar köpeğin sesini seçerler (Dog 1 veya Dog 2).  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-12 at 17.40.11.png  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.  /Users/tadejanemanic/Downloads/Farm script pic (3).png  [Adım 5]  Öğrenciler hayvan seslerini programlamalıdırlar. İlk olarak bilgisayarlarından ses eklemeleri gerekir. Bunu arka plan sesleri sekmesindeki sesleri sürükleyerek yaparlar.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.10.03.png  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.  /Users/tadejanemanic/Downloads/Farm script pic (7).png /Users/tadejanemanic/Downloads/Farm script pic (6).png    /Users/tadejanemanic/Downloads/Farm script pic (5).png /Users/tadejanemanic/Downloads/Farm script pic (4).png  [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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 14.34.32.png  Ses kaydı tamamlanınca sağ tıklayıp ismi değiştirebilirler. Bizim örneğimizde isim çiftlik olacak.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 14.47.02.png  Şimdi öğrenciler sesi kadın çiftçinin sesinin dosyasına (script) kaydetmeliler. Bunu *sound/play sound[name\_of\_sound]* *(ses/ sesi çal [sesin adı] )* bloğu /Users/tadejanemanic/Downloads/Farm script pic (1).png ile yapabilirler.  /Users/tadejanemanic/Downloads/Farm script pic (2).png  **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*  */Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.44.51.png*  *Arka plan*  **/Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.46.18.png** |
| **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 7 - Prense ve prensese evcil hayvanlarını bulmada yardım etme

|  |  |
| --- | --- |
| **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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/Screen Shot 2019-12-11 at 12.48.37.png  [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.  /Users/mateja.bevcic/Downloads/C4G7_Pomagaj_princu_in_princeski - Cela script pic (2).png  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.  /Users/mateja.bevcic/Desktop/slike LS/7/Screen Shot 2020-03-27 at 11.49.41.png    [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/8.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/9.png  “D” tuşuna basıldığında ve kız hareket ettiğinde bir çizgi çizer. "E" tuşuna basıldığında çizim durur.  Benzer şekilde, tuşa basarak kalem rengini ayarlarlar.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/10.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/11.png  [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.  /Users/mateja.bevcic/Desktop/slike LS/7/Helping Prince and Princess to find their animals script pic (5).png  [Nihai Kod]  Kız |
|  | Örneğin Prenses  /Users/mateja.bevcic/Desktop/slike LS/7 Screen Shot 2020-03-27 at 11.21.09.png  [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]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardS.png  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).  */Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/1.png*  *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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/2.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/3.png  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/rotation.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/4.png  [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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back1.png  Arka planlara tıkladığınızda, bu etkinlik için önceden hazırlanmış olan üç gerekli arka planı (tahta Kare, tahta Dikdörtgen, tahtaT (*boardSquare, boardRectangle, boardT*)) görebiliriz.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back2.png  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*).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back3.png  b) Tebeşir üzerine tekrar tıklıyoruz.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back4.png  [Adım 2] kodunun altına, öğrenciler bir oyuncuya arka planı değiştirmek için ne yapmaları gerektiğini söyleyecekleri bir blok eklerler. Yani “R” tuşuna basın.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/12.png  [Adım 4]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardR.png  “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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/8.png  [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.   1. 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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/9.png   1. 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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/13.png  [Adım 6]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardT.png  "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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/11.png  [Adım 7]  Arka planı değiştirdiğimize göre, ilk arka plana dönüp kare çizemiyoruz. O yüzden öğrenciler son bir kod ekleyecekler. [Adım 3/5] tekrarlayacaklar.   1. 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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/15.png   1. 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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/14.png  [Nihai Kod]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/final code.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/ozadje.png  [Adım 1]  Arka plan verilir. Ayrıca kız karakterini tuşlarla hareket ettirebilmek için kod verilir. Bu kod kahverengi çizgiye dokunma koşulunu da içerir.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/3.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/1.png/Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/4.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/2.png  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:**  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/5.png  [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?  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/6.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/7.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/8.png  Eğer kutucuk tıklanarak seçildi ise değişken ve üzerindeki değer ekranda görünür.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/9.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/10.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/11.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/12.png  [Adım 4]  Artık öğrenciler oyunlarında daha fazla şişe olmasını istiyor. Böylece karakterlerini kolayca kopyalayabilirler. Karaktere sağ tıklayıp *kopyala (duplicate*) seçerler.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/13.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/14.png  Kağıt karakter üzerine fareyle tıklayarak onu kâğıt karakterin içine bırakın.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/15.png  *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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/16.png  [Nihai Kod]  Kız  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code girl.png  Şişe/Kâğıt  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code bottle - paper.png  Çöp Kutusu  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code trash can.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/2.png  Elmalarının baş aşağı olmasını istemiyorlarsa, üçüncü seçeneği yani *döndürme (don’t rotate)* seçeneğini seçebilirler.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/6.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/4.png/Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/ozadje1.png  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)./Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/ozadje2.png  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/5.png  [Adım 3]  Öğrenciler artık saymak için kullanacakları değişken *puanlar (points)* oluşturabilirler. Puanlar başlangıçta 0 olarak ayarlanmalıdır (kız karakterinin üzerinde).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/11.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/7.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/9.png  [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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/10.png  [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;    /Users/mateja.bevcic/Downloads/Catching healthy food script pic (1).png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/16.png  [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.C:\Users\leo1.Portege\Pictures\coding4girls\Alice\4a298ba60c5345b8208fb7e2fbe8191a.jpg  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 söyler misiniz, buradan hangi yöne gitmeliyim? | 9 | 21 | 12 | | 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 | |  |  | |  |  | |  |  |  1. 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. |
| 1. **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) |
| 1. **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).  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.22.42.png  Rastgele değer ataması için operatörü tanıtıyoruz.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (1).png  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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.43.14.png /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.43.25.png  [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.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (3).png  [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.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (6).png  [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. /Users/tadejanemanic/Downloads/Catch the mouse script pic (9).png  [İLAVE GÖREVLER]  Öğrenciler oyunlarına herhangi bir öğe ekleyebilir. Örneğin, bir fareye her dokunduğunda sıçrayan kız.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (10).png  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*  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 11.37.43.png  *Arka plan*  */Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 11.38.29.png* |
| **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 Senaryosu Adı** | Piknik için yiyecek satın almak |
| **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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/ozadje.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/Grocery 2 script pic.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/3.png  .  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/4.png   1. *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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/5.png   1. 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. 2. Önceki Büte Değeri – Karpuz Fiyatı (previous *budget value* – *watermelon\_price*)   Öğrenciler bazı metinleri de ekleyebilir.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/6.png   1. Satın alınan ürün adedinin sayılması sağlıklı gıda (*healthy\_food*) değişkeni 1 değiştirilerek gerçekleştirilecektir.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/7.png   1. Yiyecek tıklandığında gizlenir.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/8.png  Mümkün olan çözümlerden biri;  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/9.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/10.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/12.png  [Nihai Kod]  Kız    Yiyecek  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/Final Food.png  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.   1. Değişkenler, değişkenlerin tanımlanması, başlatılması ve değiştirilme şekilleri öğrenciler ile beraber yorumlanır. 2. Rastgele sayı komutları, aritmetik ve mantık işlemleri, olay (event) yayın komutları öğrenciler ile revize edilir. 3. 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.      1. 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.  C:\Users\leo1.Portege\AppData\Local\Temp\operations3 script pic.png  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.      1. 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.      1. Yeniden etikelendir (relabel)’i seçin, işlemler (operations) görünecektir.      1. 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.     1. 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.       1. 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 Tanıtımı** | **Kısa Açıklama:**  Kraliçe Mary'nin harika dünyasına girelim. Oyuncuyu müzik dinlemesi için sarayına davet ediyor. Balo salonunda küçük dinozor arkadaşı Dino 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>     1. 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.   1. Öğretmen müzik notalarının nasıl ayarlanacağını girer:   “” *(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 | Eb | E | F | F# | G | G# | A | Bb | 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.   1. Jingle Bells (yılbaşı şarkısı) melodi kodları, müzik notaları kullanılarak da tartışılır.      1. 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). 2. 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.  https://lh4.googleusercontent.com/pmiOVv3KDx9Lqme3HaZ7wqSNhkyRIguqCCiLscVKGv6A2XGFf7K9qgR3g7IVPI7LJpljJ44Pncj21EdAyMcDgOUelrI9qFW-udwahQgYLkDI5LlGsftDS51AXbGKfAQc6JZYBtc  [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.  https://lh5.googleusercontent.com/8MF8aW7s-GatgdsXipBFNS93dBM8ZNpjEUtio3FFKukBrnqvMFCSw7h5jVQigX2pb55--lJlPpkTCSURZu9l7wjfSq6oeaRdyvyUsUYW1VYV7iIN29iWt7VnedrfhrJNHkO1-rU  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.  https://lh5.googleusercontent.com/zN_OWT89pHZ3gh81ciL8OMpmClNgQ3_tpUf2E_rk6xBmtArXOqVe6NptTF8hMD2RrJNPANONbQa7XR82o6h5vJnmbC1gTLbDrBunroxrZBssFXuQ3BmwJygtM9uBnuOhQ8wtwUA  [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.  https://lh4.googleusercontent.com/qjEJtsH7EVaZ87nmK3pnogMIRQpDy12BtzojPJEhekD_zq0Ka5pocFgAXY_Ynqc0blCH7T4moiFaV5VQMxJsszvuCfB3K3hbpYabw0PngDzDZ6vyTQh9N08f0wQELwVVJH8E-eA  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.  https://lh4.googleusercontent.com/SBynFO7adNAUvFhZhsmwqt1xpsvHGXjxb9r7hoXlSN7nhGmnduDAwkS9dytMbgTSyxfPkPKO7DbCMwxoK6ZnArj0F1dOYjOoQtqxVOXMP1CgS2UQeSQ8ZwqRS_VKftofOcWjk8g  Bir mesajı yayınlamayı bitirmek için, "*değil” "not*” operatörünü kullanın ve "a *düğmesine basıldı- key a pressed" bloğuna ekleyin.*  https://lh6.googleusercontent.com/fZDsQvwq8EXpzIPClN5maHY5JbZ2fbbxy75qYmYl_4AwLjGxqLTpPTc40hzJuhNtQlh9nGHVVyrKu67ODRvfFfkyvLxdS-yYZSPD7VKcAwAiaP2xLnUhG3tMuumLPLZhxi0xHc4  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.  https://lh4.googleusercontent.com/2UJVTx-LwmwOaOYvzd6RTblRpTDwXrwdFquL8-3ZPG0b3rNO2xthejEpP5QBN3Am-Cbwxe-nT3aLTcNGF8Rq7AUnFm9UE_Lhq7RDasW_GCam2NN2dGO4ceAcBJfsWfFv0sukufs  Notaları görüntülemek için düğmeye tıklandığında "*akorlar- chords*" mesajının yayınını etkinleştirin.  https://lh4.googleusercontent.com/AyPAYK55V9RocYTL1VgTEQP7uyF781Akmn4M6343SG2u0tNtGl-bsPo-0GIaNPfHAB0Suwdh_SASKQawgoEoFqnVfZEWbwaT4DAjppcXcHl8tZ-UNy32GiqZux_jNC-MnuPzbrw  Sahne için yeni bir "*akorları- chords*” kostümünü içeri aktarın.  https://lh3.googleusercontent.com/TiJxKAHgUuc_mRR3nt0zkr_eRW_IoTWs0Bi8wg2XpYcAGHabrbj0vAqTUm0q5ATssStEv8KGOEXFti3t0kbvrhLVkSpLsWeFBSsc5s8VqSj0nZ3kZWSujWuzRTvspa2TYjmSlmU  Sahnenin bir "akor- chords" mesajı aldığında kostümü "akorlara- chords" çevirmesini sağlayan bir kod ekleyin.  https://lh6.googleusercontent.com/PyLrO94zcfY5GTcQPkXbfsbeYFnyG10-180nvElvVX-fZ7rWXIFAPjnlyRjjbJX0rBb3vPWjRgKByLi5mbhr_RDl4bnQMXp93pi08P--DzTk6-Bu0BErVqx3QYlfxon_hO0lql0  [Adım 6]  Kostüm olarak nota şeklinde bir karakter bulun. Bu, tekrarlanması gereken şarkıyı çalmak için bir düğme olacaktır.  https://lh4.googleusercontent.com/mQ1mDLoE4QTZHwGmEvQe1J2iXRr3dwHdiclDum6KvxnndTUrdkHRSPLLT4aT1Cbv04ccoiRs1iiWfrkVfpwLbJhmTGwl0ShuFDz4Lr1V7BHwJdyvuHriNBMwrOpjbgB1VgmJgeY  Kod, şarkının ilk iki dizesi için yazılmış olup diğer dizelerin kodunu yazmanız gerekmektedir. Bu kod müziklerde görünen şarkının aynısıdır.  https://lh3.googleusercontent.com/RULXuvtXbL3sbnK1yw2MBFQoMp6sm7adUDxfuUjtc7-xcplJNq1svvKCsiMUG-HIJqDjVNng71UN_Mm9MPy5CuuYsTCY6TSmqRjmVVQGwHpAV6Oh0lrMRyL2aFvSM6PHiCJpV5c  [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.  https://lh6.googleusercontent.com/womDK0-yotcDamMu2vb7LzLUgsrTcDKaRTuFUy9d_8pLX2h-M_I6tXw3I27sl_C2cLVbyobe3Nc03tdU98dYIT_is7XuPlj5-sKzvR2OgacQBfO9BWbG0mx25slzwXGH6RX7h0k https://lh3.googleusercontent.com/CFvBjyaUJhjcjKzJSVQlkQ8YL_TzW7gK5sbqqqnTvN3XaypLwluDpMGEol_bFkG7ZWt3COKrSYmv7TQgejUBOxiSjHDOk3fDyMxwRIxqADnW_pfmx5596aI0e12sSmHrI3k8MAA  "Boş-blank" mesajını aldıktan sonra kostümü "boş-blank" olarak değiştirmek için sahneye bir "Aldığımda - When I receive" şapka bloğu ekleyin.  https://lh5.googleusercontent.com/H6wopDzfRFhH6oXrSg1gI7_tSokGoA2rRQVrsa7h8dIOdyBWSmYHd-f04BulpAzJIReoUOnfZXDHPDMQp9Uh-1S26ukh-LDbNB9L0nbwAyq5u6I3f-G_M7oQX7cIltrGusyFrtU  [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  https://lh6.googleusercontent.com/j95Bwr-0EEFBMc54FYUuwRbuaKp-_BIvvx085LSfMJ5bgYsGJCuOT363BsvclhFA-TYsTrkR8ZlO3GR_GlzLiHaFQGiVcy79Q9bI9g5KoQ3OxcGb3HE7rA7q2cIdvVXBV8oKMtA  Keman anahtarı  https://lh4.googleusercontent.com/6FM7puQhFDEXzQvMqFVQ-0olZqIVZFmW5eVgmaF79IyqdcMFUgESUvRa0qZF6v2RPPAmLYLREeBayXhd-X_ZOgwe9TBigowIg0aoDDCxzTy7HZcjnaKO0kQo7kL_gNc6bw6YYIM  Nota  https://lh3.googleusercontent.com/RULXuvtXbL3sbnK1yw2MBFQoMp6sm7adUDxfuUjtc7-xcplJNq1svvKCsiMUG-HIJqDjVNng71UN_Mm9MPy5CuuYsTCY6TSmqRjmVVQGwHpAV6Oh0lrMRyL2aFvSM6PHiCJpV5c  X  https://lh4.googleusercontent.com/vQQ44DONsWjHrj2_pmjqqMD6XVpYJgmTqoxtKfeteSOKQSM4PUpG4B74UI_3hknXea4IWruZ-XMzHa9xIjZ6gU0Nr-hidxH2fstpA5hx-ONZ-2tYMeCeGk4olAs_C-eEWM4voFw  Sahne  https://lh4.googleusercontent.com/C1MBmMIzBdwrUpXkgYGhBn3rLB9kfGcSz0rizqLq5_bvbwYdhgOfo3EbiNR6bcl5dKb7vf9Niq2D1Sm1AuufKU2bkZkKUUOCJxLAGYPctM2ct-H9YJxCTe6vLnFbA_jw7KBW_UY |
| **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 Tanıtımı** | **Kısa açıklama:** Ana karakterin rastgele konumlandırılmış yıldızları toplayıp bir hayalet tarafından kovalandığı oyunu programlayın.  **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) |

# Referanslar

Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.

Rugelj, J. (2019). Game design based learning of programming.

Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.