OTONOM HAZİNE AVCISI

Bilgisayar Mühendisliği
Kocaeli Üniversitesi

Sude Nur Opan
220201073
sudeopann@gmail.com

Çağla Gök
220201060
caglagok369@gmail.com

I. ÖZET

Otonom Hazine Avcısı projesinde, otonom hareket eden bir karakterin, içerisinde çeşitli hazineler ve engeller bulunan bir harita üzerindeki hazineleri topladığı bir oyun tasarladık. Oyunun teması yarısı kış yarısı yaz olacak şekilde ayarlandı. Nesneler hiyerarşik bir düzende yerleştirilmektedir. Oyunda amaç, karakterin tüm hazineleri en kısa sürede toplamasını sağlamaktır. Bunun için en kısa yol algoritmaları kullandık.

II. GİRİŞ

C# dilinde yazdığımız bu projeyi Visual Studio Windows Form'da geliştirdik. Projede verilen isterlere göre karakterimiz ve diğer tüm nesneler harita her yeni oluşturulduğunda random yerlere atanmaktadır. Oyun başlatıldığında karakterimiz en kısa yolu kullanarak sandıkları toplamaktadır. Her sandık toplandığında listBox1 de konumları yazdırılmaktadır. Bütün sandıklar toplandığında oyun sonlanmaktadır.

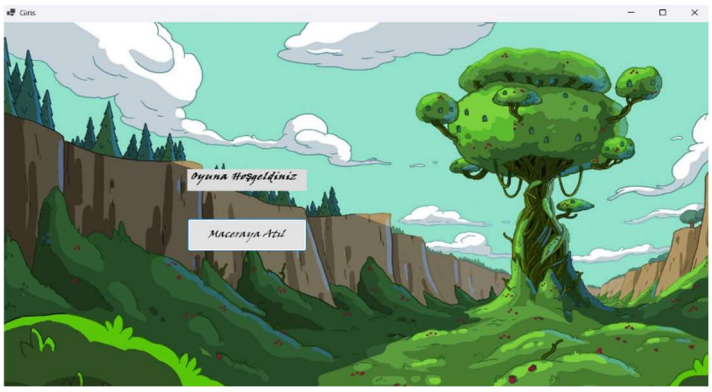
Projenin ana isterleri şunlardır:

- Oyun haritasını istenen ölçüde başlatma,
- Yeni harita oluştur ve Başlat butonlarının olması,
- Her yeniden başlatıldığında yarısı yaz yarısı kış temalı harita oluşturma,
- Bütün nesnelerin rastgele konumlarda yerleştirilmesi,
- Nesnelerin rastgele yerleşimi sırasında çakışmaması,
- Kuş ve arının hareket edeceği yerlerin kırmızı renkli olması
- Karakter hareket ettiğinde geçtiği yolların yeşil renkli olması
- Karakterin sandıkları en kısa yol algoritmasıyla bulması,
- Sandıklar toplandığında konumlarının yazdırılması.

III. YÖNTEM VE İLERLEYİŞ

Projemizi çalıştırdığımızda oyun penceresine geçebilmek için bir giriş ekranı açılıyor. Bu ekranda yer alan Maceraya Atıl butonuyla asıl oyun ekranına geçiyoruz.

Açılan yeni ekranda haritanın boyutunu kullanıcının girmesi gerekiyor. Harita boyutu girildikten sonra Yeni Harita Oluştur butonuna tıklanıp oyun için bir harita oluşturuluyor.

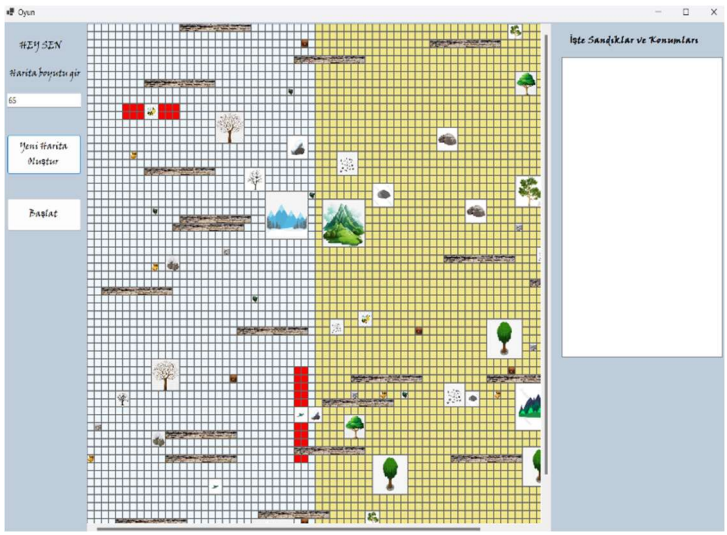


Kullanıcının girdiği boyut ölçüsünde oyun haritası oluşturulup nesnelerin tamamı çakışmayacak şekilde random atanma yapılıyor.

Ardından Başlat butonuna basıldığında oyun başlamakta olup karakter en kısa yol algoritmasıyla sandıkları bulmaya başlıyor.

Projenin içerisinde Asset ve Core adı verilen 2 klasör bulunmaktadır. Asset klasöründe projede kullandığımız nesnelerin resimleri yer almaktadır. Core klasöründe ise sınıflar yer almaktadır.

Karakter sınıfının içerisinde karakterin ismini, konumunu ayrıca KarakterAtama() metodunda karakter resminin size, location, sizemode, borderstyle ve dosya yolu gibi özelliklerini pictureBox da tutulup kullanılıyor.



Sandık sınıfında sandık resimleri bir listede tutuluyor. En az 20 adet sandık haritanın farklı yerlerine rastgele atanıyor. SandıkEngelleniyorMu() fonksiyonu sandık resimlerinin başka bir resimle çakışmasını engelliyor.

Engel sınıfında boyut ve isim tanımlanıyor.

HareketsizEngeller sınıfı Engel sınıfından türetiliyor. Constructor gerekli parametreleri alıyor ve üst sınıftan kurucu metodu çağırıyor. RastgeleKonumAl metodu, verilen resim boyutlarına göre rastgele bir konum belirliyor ve bu konumun geçerli olup olmadığını kontrol ediyor. Eğer geçerli bir konum bulunamazsa, yeni bir konum seçilene kadar döngü devam ediyor. GecerliPoz metodu, bir konumun geçerli olup olmadığını kontrol ediyor. Bu kontrol, panel içindeki diğer kontrollerle (örneğin, diğer resimlerle) çakışma olup olmadığını kontrol ederek yapılıyor. SandıkEngelleniyorMu metodu, sandık nesnelerinin etrafındaki alanın engellenip engellenmediğini kontrol ediyor. Bu, sandığın çevresindeki komşu hücrelerde başka sandık nesneleri varsa bir engel olup olmadığını kontrol ediyor.

Kayalar sınıfı HareketsizEngeller sınıfından türetiliyor. Constructor gerekli parametreleri alıyor ve üst sınıftan kurucu metodu çağırıyor. KayalarAtama metodu, kayaları panel üzerine yerleştirip belirli sayıda kaya oluşturuyor ve bunları rastgele konumlara yerleştiriyor. Bu yerleştirme işlemi sırasında, başka engellerle (örneğin, sandıklarla) çakışma olup olmadığı kontrol ediliyor. Diğer sınıflarda olduğu gibi RastgeleKonumAl metodu, GecerliPoz metodu, SandıkEngelleniyorMu metodu da aynı şekilde kullanılıyor.

Agaclar sınıfı, HareketsizEngeller sınıfından türetilmiştir. Bu sınıf, hareketsiz engeller arasında özellikle ağaçların yerleştirilmesini yönetir. Constructor gerekli parametreleri alıyor ve üst sınıftan kurucu metodu çağırıyor. AgaclarAtama metodu, ağaçları panel üzerine yerleştirip belirli sayıda ağaç oluşturuyor ve bunları rastgele konumlara yerleştiriyor. Bu yerleştirme işlemi sırasında, başka engellerle (örneğin, sandıklarla) çakışma olup olmadığı kontrol ediliyor. Diğer sınıflarda olduğu gibi RastgeleKonumAl metodu, GecerliPoz metodu, SandıkEngelleniyorMu metodu da aynı şekilde kullanılıyor.

Duvarlar sınıfı, HareketsizEngeller sınıfından türetilmiştir. Bu sınıf, hareketsiz engeller arasında özellikle duvarların yerleştirilmesini yönetiyor. Constructor gerekli parametreleri alıyor ve üst sınıftan kurucu metodu çağırıyor. DuvarAtama metodu, duvarları panel üzerine yerleştiriyor. Bu metot, belirli sayıda duvar oluşturuyor ve bunları rastgele konumlara yerleştiriyor. Bu yerleştirme işlemi sırasında, başka engellerle (örneğin, sandıklarla) çakışma olup olmadığı kontrol ediliyor. Diğer sınıflarda olduğu gibi RastgeleKonumAl metodu, GecerliPoz metodu, SandıkEngelleniyorMu metodu da aynı şekilde kullanılıyor.

Daglar sınıfı, HareketsizEngeller sınıfından türetiliyor. Bu sınıf, hareketsiz engeller arasında özellikle dağların yerleştirilmesini yönetir. Constructor gerekli parametreleri alıyor ve üst sınıftan kurucu metodu çağırıyor. DagAtama metodu, dağları panel üzerine yerleştiriyor. Bu metot, belirli sayıda dağ oluşturuyor ve bunları rastgele konumlara yerleştiriyor. Bu yerleştirme işlemi sırasında, başka engellerle (örneğin, sandıklarla) çakışma olup olmadığı kontrol ediliyor. Diğer sınıflarda olduğu gibi RastgeleKonumAl metodu, GecerliPoz metodu, SandıkEngelleniyorMu metodu da aynı şekilde kullanılıyor.

DinamikEngel sınıfı, Engel sınıfından türetiliyor ve dinamik engellerin temel özelliklerini ve davranışları tanımlanıyor. Constructor, bir boyut parametresi alıyor, bu boyutu ve engelin adını temel sınıfa ileterek temel sınıfın kurucu metodunu çağırıyor. Ayrıca random nesnesini oluşturuyor. HareketEt soyut metodu, dinamik engellerin hareket etme davranışını tanımlamak için alt sınıflar tarafından uygulanıp dinamik engellerin konumlarını güncelliyor. RastgeleKonumAyarla soyut metodu, dinamik engellerin rastgele bir konuma yerleştirilmesini sağlamak için alt sınıflar tarafından uygulanıp dinamik engellerin konumlarını rastgele bir şekilde ayarlıyor.

Ari sınıfı, arıların özelliklerini ve davranışlarını tanımlıyor. Kurucu metot, panel boyutunu, kenar boyutunu ve arıları başlatıyor. Bu metodun içinde AriAtama metodu çağrılıyor. AriAtama metodu, arıların başlangıç konumunu ve görüntüsünü ayarlıyor. Arılar, rastgele konumlarda ve belirli bir boyutta olacak şekilde yerleştiriliyor. Timer_Tick(object sender, EventArgs e): Zamanlayıcı tetiklendiğinde çağrılan olay işleyicisidir. Arının hareketini kontrol eder. AriHareketEttir(): Arının yatay hareketini yönetir. Belirli bir yön ve belirli bir mesafe ile ileri veya geri hareket eder. GuncelleAriPictureBoxKonum(): Arının PictureBox'ının konumunu günceller.RastgeleKonumAl metodu, arının rastgele bir konum almasını sağlar. Bu metot, arının panel içindeki geçerli konumlarını kontrol eder ve çakışma olmayan bir konum seçer. GecerliPoz metodu, belirtilen bir konumun geçerli olup olmadığını kontrol eder. Bu metot, arının belirli bir konuma yerleştirilmesinin mümkün olup olmadığını kontrol eder ve çakışma olup olmadığını kontrol eder. Ayrıca, arının duvarlar, ağaçlar, dağlar ve kayalıklar gibi engellerle çakışmadığından emin olur.

Kus sınıfı, arıların özelliklerini ve davranışlarını tanımlıyor. Kurucu metot, panel boyutunu, kenar boyutunu ve arıları başlatıyor. Bu metodun içinde KusAtama metodu çağrılıyor. KusAtama metodu, kuşların başlangıç konumunu ve görüntüsünü ayarlıyor. Kuslar, rastgele konumlarda ve belirli bir boyutta olacak şekilde yerleştiriliyor. Timer_Tick(): Bir zamanlayıcı etkinleştirildiğinde, kuşun hareketini kontrol eder. KusuHareketEttir(): Kuşun yukarı ve aşağı hareketini yönetir. GuncelleKusPictureBoxKonum(): Kuşun PictureBox'ının konumunu günceller. KusAtama(): Kuşu panel içine yerleştirir. Kuşun görüntüsünü belirler ve panel üzerinde rastgele bir konuma yerleştirir.RastgeleKonumAl metodu, kuşun rastgele bir konum almasını sağlıyor. Bu metot, kuşun panel içindeki geçerli konumlarını kontrol ediyor ve çakışma olmayan bir konum seçiyor. GecerliPoz metodu, belirtilen bir konumun geçerli olup olmadığını kontrol ediyor. Bu metot, arının belirli bir konuma yerleştirilmesinin mümkün olup olmadığını kontrol ediyor ve çakışma olup olmadığını kontrol ediyor. Ayrıca, arının duvarlar, ağaçlar, dağlar ve kayalıklar gibi engellerle çakışmadığından emin oluyor.

Lokasyon sınıfı x ve y değişkenlerini tutup kodda konum ile ilgili yerlerde kullanılıyor.

Uygulama sınıfı, Windows Forms uygulamasının ana formunu temsil ediyor. Bu form, oyun alanını (grid), karakteri, sandıkları ve diğer engelleri içeriyor.

Form, birçok bileşen ve işlev içerir:

panel1: Oyun alanını temsil eden bir Panel kontrolüdür.

textBox1: Kenar boyutunu girmek için bir metin kutusu kontrolüdür.

button1: Grid'i oluşturmak için bir düğme kontrolüdür.

button2: En yakın sandığı bulmak için bir düğme kontrolüdür.

listBox1: Çeşitli bilgileri görüntülemek için bir liste kutusu kontrolüdür.

Uygulama sınıfının constructor'ı (public Uygulama()), formun ilk oluşturulduğunda çalışıyor. Bu constructor, bileşenlerin başlatılmasını sağlıyor. OluşturGrid metodu, grid'in oluşturulmasını ve oyunun başlatılmasını sağlıyor. Bu metod, kenar boyutunu ve oyun alanındaki engellerin yerlerini belirliyor. Engelleme metodu, belirli bir hücrenin engellerle çakışıp çakışmadığını kontrol ediyor. Bu metot, çakışma durumunda true, aksi halde false döndürüyor. RastgeleKonumAl metodu, rastgele bir hücre konumu almak için kullanılıyor. Bu metod, oyun alanındaki geçerli konumları kontrol ederek rastgele bir konum seçiyor. CakısmayanPoz metodu, resimlerin konumunu belirlerken çakışmayı kontrol ediyor. Bu metot, daha önce kullanılan konumları kontrol ederek çakışmayı önlüyor. KısaYolBulma metodu, karakterin sandığa en kısa yolu bulmasını sağlıyor. Bu metod, karakterin mevcut konumundan hedef konuma en kısa yolun bulunmasını sağlayan bir algoritma uyguluyor. KarakterYolBoyuHareket metodu, karakterin belirlenen yolu takip ederek hareket etmesini sağlıyor. Bu metot, karakterin her adımda belirli bir zaman aralığında hareket etmesini sağlıyor. YeniYonBelirle metodu, karakterin rastgele bir yöne dönmesini sağlar. SandıkEksikMı metodu, oyun alanında sandık olup olmadığını kontrol ediyor. Eğer sandık varsa true, yoksa false döndürüyor. button1_Click metodu, "Grid Oluştur" düğmesine tıklandığında tetikleniyor. Bu metot, girilen kenar boyutuna göre grid'i oluşturuyor. button2_Click_1 metodu, "En Yakın Sandık Bul" düğmesine tıklandığında tetikleniyor. Bu metot, karakterin en yakın sandığı bulup ona doğru hareket etmesini sağlıyor.

IV. Deneysel Sonuçlar

Panelin boyut ölçeklendirmesinde, ilk önce panel.Boyu / hucreboyutu ve panel.Genislik / hucreboyutu ölçüsüyle dinamik bir ölçeklendirme yaptık. Ancak bu kodumuzda rastgele atama yapmada ve çakışmada problem yarattı. Bizde hücre boyutuna belirli bir sayı atayıp bu sorunu çözdük.

V. Sonuç

Projenin isterlerine uygun bir şekilde kullanıcıdan harita boyutu alınıyor, tüm nesneler random atanıyor, karakterimiz yazdığımız algoritma sayesinde sandıkları bulup toplayarak otonom hareket ediyor.

VI. Kaynakça

https://learn.microsoft.com/tr-tr/dotnet/csharp/

https://stackoverflow.blog/

https://teknikakil.com/c-sharp/c-sharp-form-ornekleri/c-ornekleri-yilan-oyunu/

https://www.kemalkefeli.com.tr/csharp-ile-yilan-oyunu.html

https://www.turkhackteam.org/konular/yilan-oyunu-c-kodlari.2040322/

https://www.youtube.com/watch?v=Lsew1nJD_9A

https://www.youtube.com/watch?v=HIoqBT0-P1g

https://www.youtube.com/watch?v=HIoqBT0-P1g

https://www.youtube.com/watch?v=Nc8TUL4_RCU

https://medium.com/tapu-com-bak%C4%B1%C5%9F-a%C3%A7%C4%B1s%C4%B1/bfs-breath-first-search-geni%C5%9F-%C3%B6ncelikli-arama-algoritmas%C4%B1n%C4%B1-tan%C4%B1yal%C4%B1m-ec7050a41af

https://bilgisayarkavramlari.com/2009/03/02/a-yildiz-arama-algoritmasi-a-star-search-algorithm-a/

VII. Pseudo Kod

```
Başla
 Sınıf Uygulama
  Oluştur
    karakterPictureBox = Yeni PictureBox()
    button1_Click
     Eğer kenarBoyutu bir sayıya dönüştürülebilirse
      OluşturGrid(kenarBoyutu)
     Değilse
     "Lütfen geçerli bir sayı girin." uyarısı göster
     OluşturGrid(kenarBoyutu)
       panel1.Controls'i temizle
       sandıklar.SandıkAtama()
       dağlar.DagAtama()
       kayalar.KayalarAtama()
       ağaçlar.AgaclarAtama()
       duvarlar.DuvarAtama()
       kuşlar.KusAtama()
       Arılar arılar = Yeni Arılar(hucreBoyutu, panel1, kenarBoyutu)
       arılar.AriAtama()
       kullanılanPoz listesine karakterPictureBox'in konumunu ekle
       KenarBoyutu boyutunda bir döngü başlat
       KenarBoyutu boyutunda bir döngü başlat
       Eğer j < kenarBoyutu / 2 ise
         panel1'in arka plan rengini Alice Mavisi yap
       Değilse
         renkliPanel'in arka plan rengini Khaki yap
         panel1'e renkliPanel'i ekle
           Döngü Sonu
    Engelleme(hedefPoz)
     RastgeleKonumAl(resimGenişlik, resimYükseklik)
      YesilBoyamal(x, y)
    KısaYolBulma(başla, hedef)
      Önceki = Yeni Sözlük<Point, Point>()
      ZiyaretEdildi = Yeni Küme<Point>()
      Kuyruk = Yeni Kuyruk<Point>()
      Kuyruğa başla'yı ekle
```

```
        ZiyaretEdildi'ye başla'yı ekle
        Kuyruk boş olana kadar devam et
            Mevcut = Kuyruk'tan al
            Eğer Mevcut, hedef'e eşitse
              Yol = Yeni Liste<Point>()
              Mevcut, başla olana kadar devam et
              Yol'e ekle
              Mevcut'i Önceki'den al
            Döngü Sonu
            Yol döndür
        Sonuç
        YeniYonBelirle
        KarakterYolBoyuHareket(yol)
            Döngü Sonu
        Sonuç
            KomsuHucreleri'nden geçerli olanları seç ve döndür
        Sonuç
        KarakterÇarpışmasıKontrolü(hedefPozisyon)
        Sonuç
        SandıkEksikMı
    Sonuç
Bitir
```

```
```