HAZİNE AVCISI OYUNU

Otonom Hazine Avcısı projesinde, klavyeden hareket eden bir karakterin, içerisinde çeşitli hazineler ve engeller bulunan bir harita üzerindeki hazineleri topladığı bir oyun tasarlanması beklenmektedir. Oyunda amaç, karakterin tüm hazineleri en kısa sürede toplayarak oyunun bitirilmesini sağlamak. Bunun için nesneye yönelik programlama ve veri yapıları bilgilerinin kullanılması beklenmektedir.

Amaç: Proje gerçekleştirimi ile öğrencilerin nesneye yönelik programlama ve veri yapıları bilgisinin pekiştirilmesi, uygulanması ve problem çözme becerisinin geliştirilmesi amaçlamaktadır.

Programlama Dili: Projenin geliştirilmesinde C++, C#, Java dilleri kullanılabilir.

PROJE İSTERLERİ:

Bu projede sizden oyun karakterinin, rastgele oluşturulmuş bir harita üzerinde rastgele belirlenmiş bir başlangıç noktasından başlayarak, oluşturulmuş engellere takılmadan, en kısa sürede ve en kısa yoldan giderek ızgaradaki tüm hazine sandıklarını toplamasını sağlamanız beklenmektedir. Karakter oyun boyunca tüm ızgarayı değil, yalnızca gerekli yolları gezmelidir.

Oyunun Genel Adımları:

Haritanın Oluşturulması: Harita, her uygulama başladığında yeniden üretilmelidir. Haritanın oluşturulması için gereken algoritmanın sizler tarafından geliştirilmesi beklenmektedir. Geliştirdiğiniz algoritma, her adımda rastgele bir harita üretildiğini doğrulamalıdır. Üretilmesi ve doğrulanması için etkili algoritmaların geliştirilmesi istenmektedir. Mümkün olduğu kadar efektif çözümler geliştirilmeli, kaba kuvvet (brute force) uygulamalarından kaçınılmalıdır.

Harita oluşturulduğunda haritanın sol tarafı kış, sağ tarafı yaz temasında olmalıdır. Bir sonraki başlıkta anlatılacak sabit nesnelerin görselleri bu temalara göre değişmelidir. Temaya göre sabit nesnelerin üretilmesinde hiyerarşik nesne yapısı kullanılmalıdır.

Tüm hazine sandıklarının ulaşılabilir (engellerin ortasında kalmadığından) olduğundan emin olmanız gerekir.

Harita İçeriğinde Bulunması Gerekenler: Harita üzerinde yol (üzerinde ilerlenebilen kareler), engeller ve hazine sandıklarını içeren kareler bulunmalıdır.

Engeller, sabit ve hareketli olmak üzere iki çeşit olmalıdır. Sabit engeller; ağaç, dağ, kaya ve duvar gibi oyun boyunca hareket etmeyen nesnelerdir. Yaz ve Kış için ayrı sabit engeller tanımlanmalıdır. Hareketli engeller ise kuş ve arı gibi nesneler olup kuşlar 5 karelik, arılar ise 3 karelik alanda hareket edebilir. Bu hareket kuşlar için yukarı-aşağı, arılar için sağ-sol şeklinde olmalıdır. Hareketli nesnelerin her biri 2x2 birimlik alan kaplamalıdır.

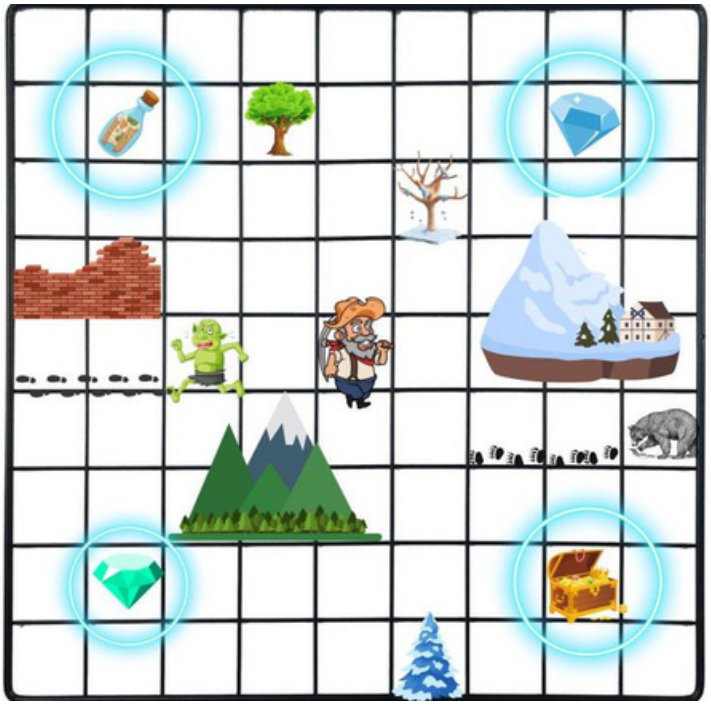
Her harita oluştuğunda en az 20 adet sabit (her sabit nesneden en az 2 adet oluşturulması zorunludur), 3 adet hareketli nesnenin üretilmiş ve harita üzerinde uygun yerlere yerleştirilmiş olması beklenmektedir.

Hazine sandıkları; altın sandık, gümüş sandık, zümrüt sandık ve bakır sandık olmak üzere dört çeşit olmalıdır. Hazine sandıkları toplanabilen nesnelerdir. Karakter bir sandığı topladığında ilgili sandık harita üzerinde kaybolmalıdır. Sandıkların toplanma sırasına göre ekranın sağ üst köşesinde sandığın türüne göre bilgi verilmelidir (Örneğin; toplanan sandık altın sandık ise "Altın sandık toplandı! (8,16-8,18) konumunda bulundu" yazılmalıdır).

Her harita oluşturulduğunda her sandık türünden en az 5 adet üretilmiş ve harita üzerinde uygun yerlere yerleştirilmiş olması beklenmektedir.

Haritada Başlangıç ve Bitiş Noktalarının Belirlenmesi: Karakterin başlangıç noktası harita üzerindeki uygun (engel ya da sandık içermeyen) karelerde rastgele olacak şekilde belirlenmelidir. Bitiş noktası, geliştirilen algoritmaya göre toplanan en son hazine sandığının konumu olmalıdır.

Beklenen Çıktılar: Tüm bu bilgiler doğrultusunda, karakterin rastgele belirlenen başlangıç noktasından başlayarak tüm hazine sandıklarını en kısa sürede toplayabileceği harita oluşturulmalıdır.



Şekil 1. Örnek ızgara ve engel görünümü

Sınıf Tanımları:

Verilen projede aşağıdaki tanımlara uygun olacak şekilde sınıfların oluşturulması ve kodlanması beklenmektedir.

Karakter:

Bu sınıfta bulunması gereken özellikler ve fonksiyonlar:

● ID, Ad bilgileri tutulmalıdır.
● Karakterlerin ilerlediği koordinatları tutacak Lokasyon değişkenleri olmalıdır.
● Constructor, Get, Set metotları yer almalıdır.

NOT: *Karakter çapraz gidemez. Sadece sağ, sol, yukarı ya da aşağı yönde hareket sağlayabilir.*

Lokasyon Sınıfı :

● x ve y koordinatlarını tutan iki farklı değişken tutulmalı. Constructor, Get ve Set

metotları yer almalıdır.

Engel Sınıfı:

● Engel sınıfında sabit ve hareketli olmak üzere 2 farklı tipteki engellerin oluşturulması gerekmektedir.

Hareketsiz Engeller:

Ağaçlar:

● Çeşitli türleri ve boyutları olabilir.
● Yoğun ormanlık alanlar veya tek başına ağaçlar olarak bulunabilirler.
● Oyuncu üzerinden geçemez.
● 2x2, 3x3, 4x4 veya 5x5'lik boyutta olabilirler.

Kayalar:

● Farklı boyutlarda ve şekillerde olabilirler (Örneğin; küçük çakıl taşları, büyük kayalar).
● Oyuncu üzerinden geçemez.
● 2x2 veya 3x3'lük boyutta olabilirler.

Duvarlar:

● Bir duvar başka bir duvarla yan yana olmamalıdır.
● Oyuncu üzerinden geçemez.
● 10x1'lik boyutta olabilirler.

Dağlar:

- Birden fazlası bir araya gelerek dağ kümeleri oluşturabilir.
- Oyuncu üzerinden geçemez.
- 15x15'lik boyutta olabilirler.

Dinamik Engeller:

Kuşlar:

- Yalnızca yukarı-aşağı yönde hareket edebilir.
- Oluşturulduğu kareden itibaren 5 kare yukarı ve aşağı gidip gelebilir.
- Oyuncu üzerinden geçemez.
- 2x2 birim boyutunda olmalıdır.

Arılar:

- Yalnızca sağ-sol yönde hareket edebilir.
- Oluşturulduğu kareden itibaren 3 kare sağa ve sola gidip gelebilir.
- Oyuncu üzerinden geçemez.
- 2x2 birim boyutunda olmalıdır.

Uygulama Sınıfı:

- Uygulama içerisinde karakterin hedefe kaç adımda ulaştığı, hangi nesneleri elde ettiği gibi bilgilerin tutulması ve ekranda gösterilmesi sağlamalıdır.

NOT: *Yukarıdaki sınıflar, işlevsel bakımdan genel olarak tanımlanmış olup her sınıf için kullanılacak özellik (property) ve metotların tanımlaması sizden beklenmektedir. Ayrıca, gerektiği durumlarda neden kullanıldığının açıklanması koşuluyla yukarıdaki sınıflardan farklı sınıf tanımlamaları yapılabilir.*

Projede ağaç, kuyruk gibi veri yapılarının ve Encapsulation, Inheritance, Polymorphism, Abstraction yapılarından gerekli olanların kullanılması gerekmektedir. Projede her yapıyı kullanmamış olsanız bile, proje sunumu esnasında bu yapıların ne olduğunu bilip bilmediğinizi ölçecek sorular sorulacaktır.

Arayüz ve Görsellik:

- Arayüzdeki tüm görseller net ve açık olmalıdır. Oluşturulan yol, hareketli nesnelerin hareket güzergahı net bir şekilde görülmelidir.
- Proje çalıştırıldığında rastgele üretilmiş harita tasarımı ekrana getirilmelidir. Başlangıç noktası belirlenmiş olmalıdır.

NOT: *Verilen projede arayüz tasarımından puan alacaksınız. Bu nedenle arayüze özen göstermeniz beklenmektedir. Arayüz tasarımı için hazır tasarımlar kullanılmamalıdır.*