Programlama Laboratuvarı 1. Proje Ödevi

1. Safiyye Berra Çevik
Bilgisayar Mühendisliği
Koaceli Üniversitesi
Koaceli, Türkiye
berracevik642@gmail.com

2. Esra Kurt
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
Koaceli, Türkiye
esrakurt221@gmail.com

***Özetçe*—Bu rapor, Kocaeli Üniversitesi Programlama Laboratuvarı 2 dersinin 1. Proje ödevi için hazırlanmıştır.**

Anahtar Kelimeler —Java, Oyun, En KısaYol Algoritması, Engeller, Sis

I. ÖZET

Bu proje; Java Programlama dili kullanılarak karakterin en kısa yol algoritmasından faydalanıp birçok türdeki engeli aşıp sisten dolayı göremediği farklı çeşitlerdeki hazinelere olabildiğince az sürede ulaşıp oyunu bitirmesi amacına uygun yapılmıştır.

II. GİRİŞ

Otonom hareket eden bir karakterin rastgele oluşturulmuş bir harita üzerinde çeşitli hazineleri topladığı bir oyun olan bu projede haritanın oluşturulması, harita içeriğinde bulunan çeşitli engellerin oluşturulması ve çakışmaması aynı şekilde hazine türlerinin oluşturulması ve çakışmaması, en kısa yol algoritmalarının kullanılması, kullanıcıdan harita boyut bilgisinin alınması ve her yeni girişte yeni bir harita oluşturulması, haritanın ve engellerin temalarını yaz ve kış olarak 2 türde olması istenmiştir.

Bunlarla birlikte; karakterin başlangıç noktasının rastgele belirlenmesi ve bitiş noktası ise toplanacak son hazine sandığının konumu olmalıdır. Engeller ve hazineler de haritaya rastgele yerleştirilmelidir. Hareketli engellerin hareket alanlarını kırmızı ile işaretlenmeli ve karakter bu alanlara girememelidir. Aynı zamanda karakter hareketsiz engellerden de geçememelidir.

III. YÖNTEM

İlk olarak hangi programlama dilini ve geliştirme ortamını kullanacağımıza karar verdik. Java'ya hakim olduğumuzu düşünerek ve derslerde de Netbeans'i sık sık kullanmış olduğumuzdan bunlarda karar kıldık. Projemizde kullanacağımız en kısa yol algoritması ve görseller için araştırmalara başlayıp classlarımızı proje raporundaki isterlere göre oluşturduk. Önce Haritapanel classımızı oluşturup haritada görmek istediğimiz görsellere uygun nesneleri oluşturup kodlamaya başladık. Haritada görmemiz gereken engeller ,hazineler ve karakter için ayrı ayrı classlar açıp içerilerini doldurmaya başladık. Polymorphism kullanarak engeller classımızı "abstract class" olarak oluşturup ata sınıftan çocuk sınıflar oluşturarak engel sınıfının genel özelliklerini alt sınıfların da taşımasını sağladık. Ve abstract classlarda metotlar gövdesiz olarak kullanıldığından metotlarımızı gövdesiz kullandık. Polymorphism'in üst sınıfın referansı ile alt sınıftaki nesnelerin kullanılmasına olanak sağladığını göz önüne alarak alt sınıflar olan hareketli engeller için "kus , ari" hareketsiz engeller için de "snowkaya, sunkaya ,snowduvar, sunduvar, snowagac, sunagac, snowdag, sundag" classlarını ve nesnelerini oluşturduk. Ve engellerin üzerinden geçilmemesi için "getGeçilemezKonumlar" metodunu oluşturduk.

Hazineler classımızda " Hazine" üst sınıfında "AltinSandik, GumusSandik, ZumrutSandik, BakirSandik" alt classlarını açtık ve gerekli constructor'ları oluşturduk. Ve yine gerekli metotlarımızı tanımladık. Karakter classımızda karakterin ID'sini, adını ve lokasyon değişkenlerini tanımlayıp constructor'larını oluşturduk. Rastgele başlaması için "rastgeleBaslangicLokasyonu" metodunu tanımladık. Lokasyon sınıfında x ve y özel değişkenlerini tanımlayıp yine kurucu

metotlarını oluşturduk. Ve x ve y değerlerini ayarlayıp geri döndüren “getter, setter” metotlarını kullandık. Uygulama classında toplanan hazineler ve konumları listelerini tanımladık. “hazineTopla” metodu oluşturduk. Toplanan hazinelerin ve konumlarının listelerini “getter” metoduyla geri döndürdük. Karakterin oyunu kaç adımda bitirdiği, hangi sandıkları topladığı konumlarıyla birlikte ekrana yazdırılması, bunların istenilen sırada sırada sıralanıp (altin,gumus,zumrut,bakir) yazdırılmasını yine uygulama classında yazdık.

“EnKisaYolBulucu” classımızda oyunlarda en kısa yolu bulmak için genelde kullanılan “A Star“ algoritmasını kendi projemize uygun olarak kodladık. Bunun için videolardan ve github’tan fazlaca yaralandık.Ve bu class için gerekli olan “Node, Path, SortedList” classlarını açtık. Bu algoritmanın mantığına göre karakter sezgisel olarak hedefe ilerliyor. En kısa yol içingerekli maliyetleri hesaplıyor ve elde ettiği düğüm olarak tuttuğu hazineleri açılan kuyrukta tutarken hala uğramadığı hazineleri kapalı kuyrukta tutuyor. Hedefe ilerlerken komşu kareleri ziyaret edip etmediğini kontrol edip uğradığı yere tekrar uğramadan hızlıca hedefe yönelip olabilecek en kısa sürede sandıkları topluyor.

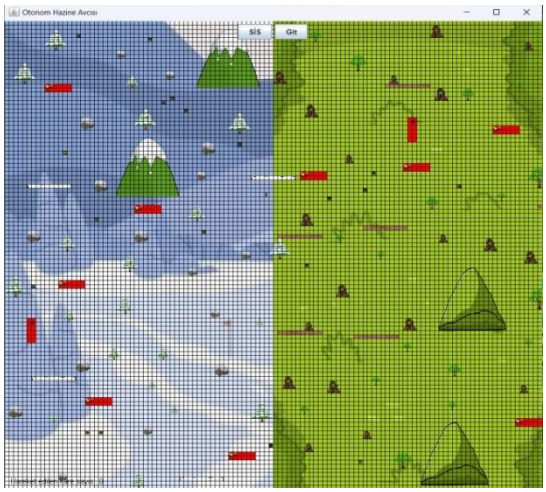
Main classımız olan “OtonomHazineAvcisi” classında giriş arayüzü oluşturup karakter id’si ve adını sorguladığımız butonlar ve sonrasında kullanıcının harita boyutunu girmesi için giriş paneli oluşturduk. Harita oluşturma ve oyuna başlama işlemlerinin de bu classta yapılmasını sağladık.

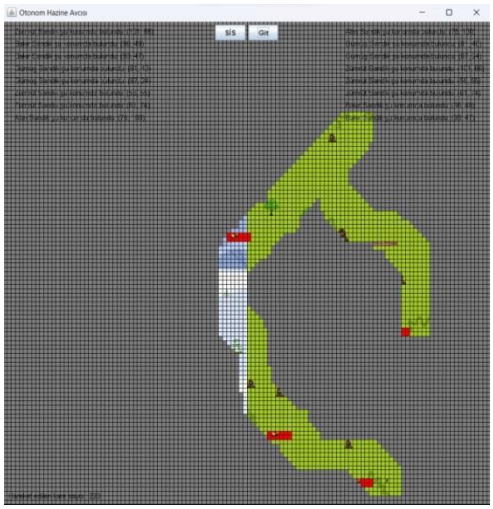
Harita Panel sınıfı oyunun haritasını oluşturur ve arka plan resmini yükler. Engelleri (kuşlar, arılar) ve hazineleri rastgele konumlara yerleştirir.Oyun alanını ve öğeleri grafiksel olarak görselleştirir. "SİS" ve "Git" adında iki buton ekler."SİS" butonuna basıldığında, haritadaki belirli bir alanı gri tonlamayı sağlar."Git" butonuna basıldığında, hazine avcısının otomatik olarak hareket etmesini ve hazineleri toplamasını başlatır veya durdurur.Rastgele engeller (kuşlar, arılar) ve farklı türde hazineler (altın, gümüş, zümrüt, bakır) oluşturur ve bunları haritaya yerleştirir.Engel ve hazine yerleştirme işlemleri sırasında, çakışmaları ve uygun konumları kontrol eder.Hazine avcısının otomatik hareketini sağlayan zamanlayıcıyı oluşturur.Hazine avcısının hedefe doğru hareket etmesini sağlar.Tüm hazineler toplandığında, oyunun bitiş zamanını ve toplam hareket sayısını kullanıcıya bildiren bir iletişim kutusu görüntüler.Harita üzerinde bulunan çeşitli öğelerin (engeller, hazineler) hareketlerini ve etkileşimlerini yönetir.Kullanıcı etkileşimlerini (buton tıklamaları) dinler ve buna göre oyun akışını kontrol eder. rastgeleHaritaOlustur(): Oyun haritasını oluşturan bir metod. Harita üzerine engeller, hazineler ve karakter yerleştirilir. Her türden belirli sayıda hazine yerleştirilir ve harita boyutuna göre engeller, ağaçlar, dağlar ve duvarlar rastgele yerleştirilir. hareketEt(): Karakterin hareket etmesini sağlayan bir metod. Karakter, en yakındaki hazineye doğru hareket eder. Eğer hedefe bir birimden daha yakınsa, doğrudan hedefe gitmek için hareket eder. Aksi takdirde, yatay veya dikey olarak hareket eder.rastgeleHareketEt(): Karakterin rastgele hareket etmesini sağlayan bir metod. Karakter, rastgele bir yöne doğru bir kare hareket eder.enYakinHazineyiBul(): Karakterin en yakındaki hazineyi bulmasını sağlayan bir metod. Karakterin bulunduğu konuma en yakın olan hazineyi belirler.gecerliKonum(int x, int y): Belirtilen konumun geçerli olup olmadığını kontrol eden bir metod. Konumun harita sınırları içinde olması ve engellerle çakışmaması kontrol edilir.paintComponent(Graphics g): Haritayı ve karakterin hareketini görselleştiren bir metod. Harita üzerindeki engeller, hazineler ve karakter çizilir. Ayrıca, toplanan hazinelerin bilgileri ve hareket edilen kare sayısı da ekrana yazdırılır.

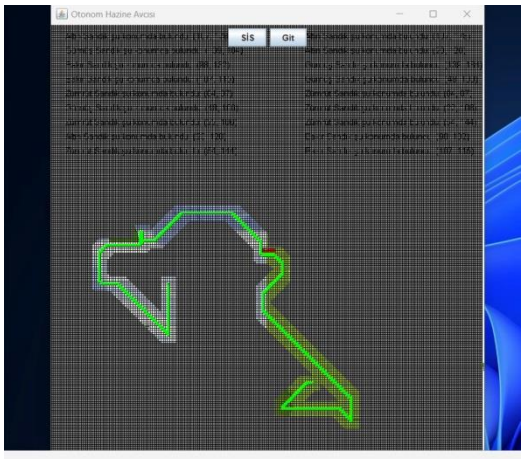
IV. SONUÇ

Bu projede Java ile oyun yapıp nesneye yönelik programlama bilgilerimizi iyiden iyiye pekiştirdik. Daha önce bilmediğimiz en kısa yol algoritması olan “Astar” algoritmasının nasıl çalıştığını öğrendik. İki boyutlu oyun yapımına ve yapay zekada sıkça kullanılan otonom mekanizmasına aşina olduk.

DENEYSEL SONUÇLAR



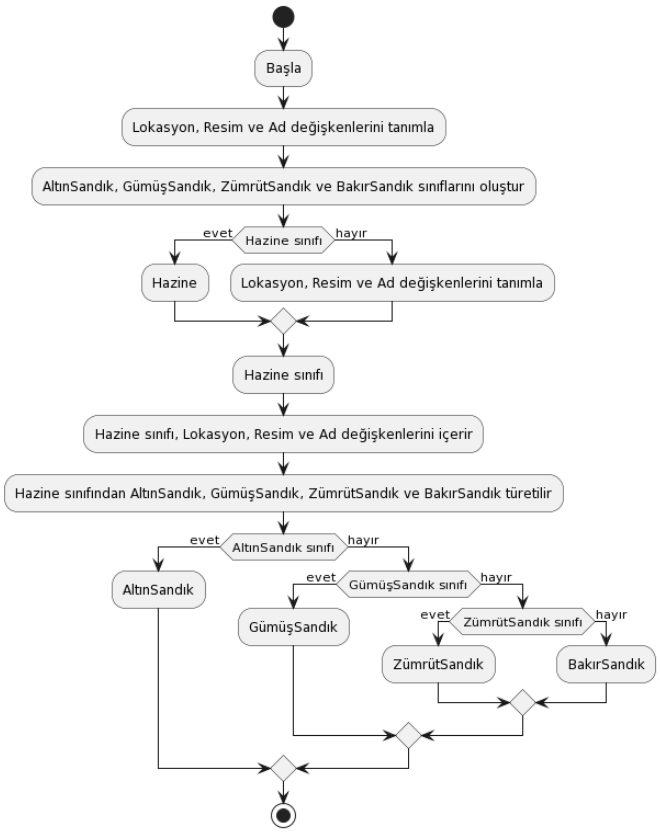


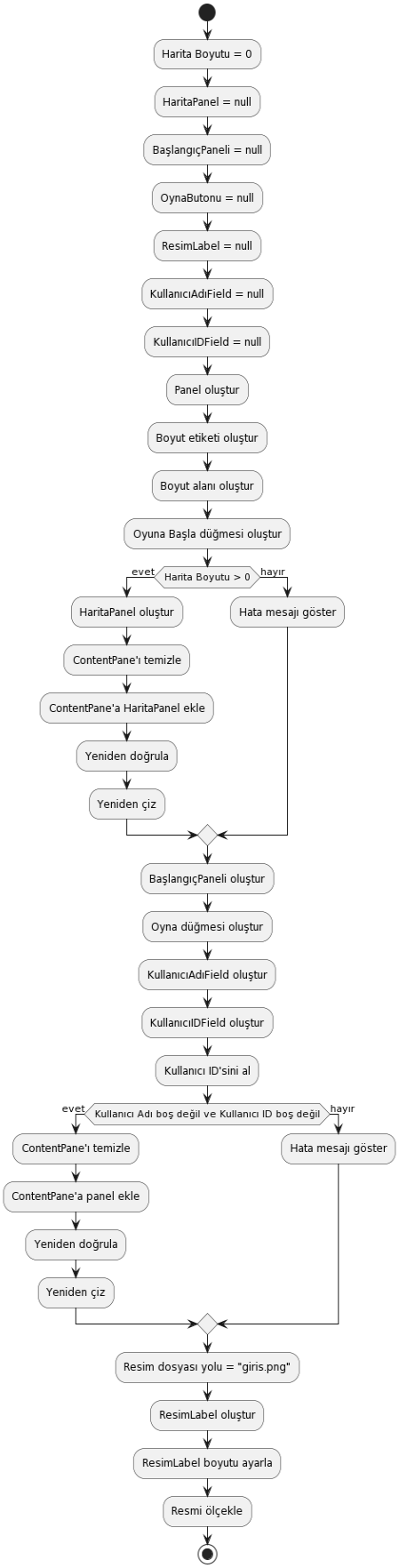


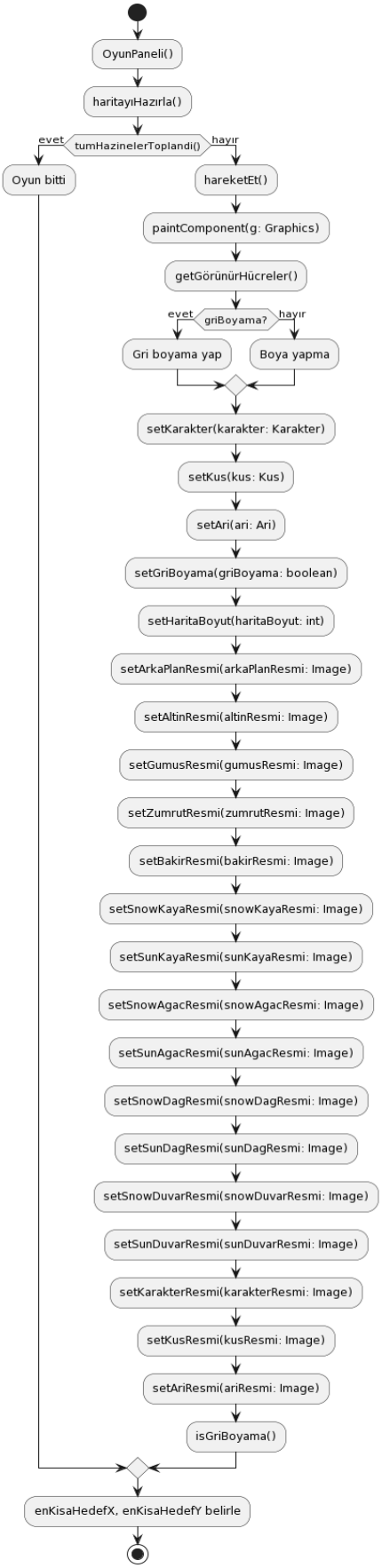
Altın Sandık şu konumda bulundu: (107, 178), 26 adımda bulundu
Gümüş Sandık şu konumda bulundu: (139, 184), 63 adımda bulundu
Bakır Sandık şu konumda bulundu: (98, 132), 155 adımda bulundu
Bakır Sandık şu konumda bulundu: (107, 115), 180 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (84, 87), 257 adımda bulundu
Gümüş Sandık şu konumda bulundu: (48, 100), 305 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (22, 106), 353 adımda bulundu
Altın Sandık şu konumda bulundu: (23, 120), 367 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (54, 144), 421 adımda bulundu

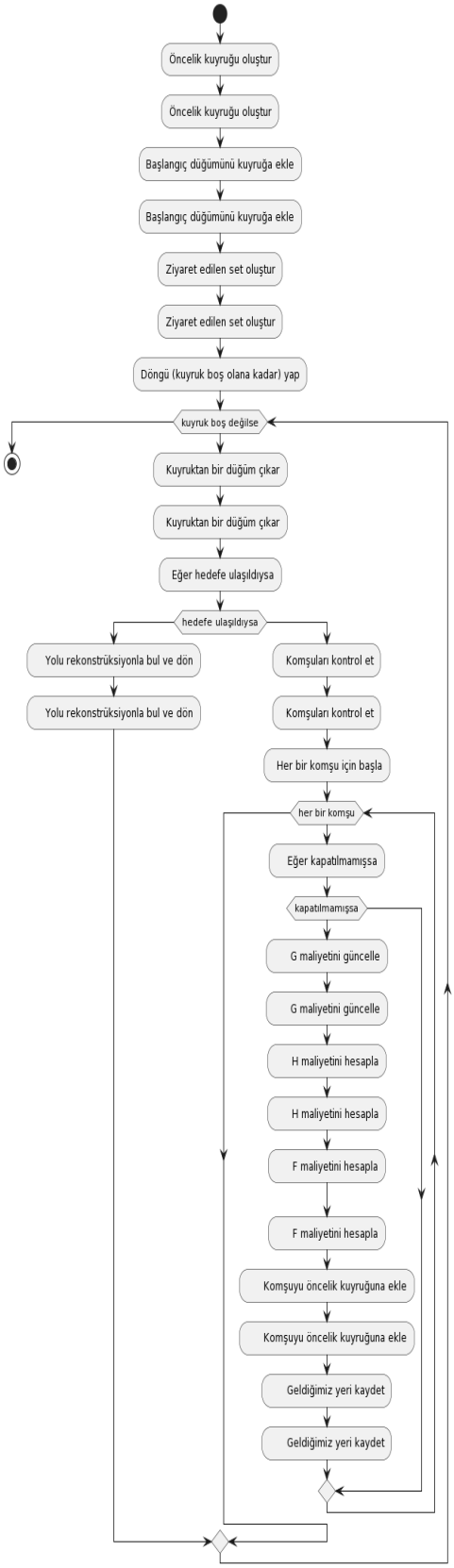
Altın Sandık şu konumda bulundu: (107, 178), 26 adımda bulundu
Altın Sandık şu konumda bulundu: (23, 120), 63 adımda bulundu
Gümüş Sandık şu konumda bulundu: (139, 184), 155 adımda bulundu
Gümüş Sandık şu konumda bulundu: (48, 100), 180 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (84, 87), 257 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (22, 106), 305 adımda bulundu
Zümrüt Sandık şu konumda bulundu: (54, 144), 353 adımda bulundu
Bakır Sandık şu konumda bulundu: (98, 132), 367 adımda bulundu
Bakır Sandık şu konumda bulundu: (107, 115), 421 adımda bulundu

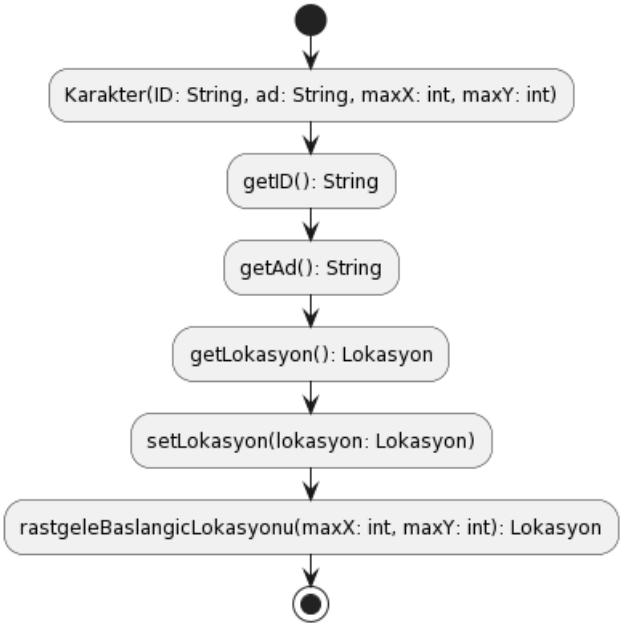
AKIŞ DİYAGRAMLARI



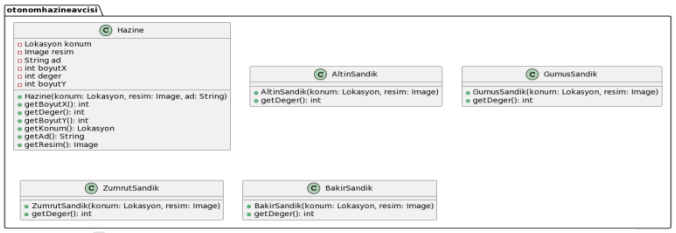

Harita Boyutu = 0
HaritaPanel = null
BaşlangıçPaneli = null
OynaButonu = null
ResimLabel = null
KullanıcıAdıField = null
KullanıcıIDField = null
Panel oluştur
Boyut etiketi oluştur
Boyut alanı oluştur
Oyuna Başla düğmesi oluştur
evet
Harita Boyutu > 0
hayır
HaritaPanel oluştur
Hata mesajı göster
ContentPane'ı temizle
ContentPane'a HaritaPanel ekle
Yeniden doğrula
Yeniden çiz
BaşlangıçPaneli oluştur
Oyna düğmesi oluştur
KullanıcıAdıField oluştur
KullanıcıIDField oluştur
Kullanıcı ID'sini al
evet
Kullanıcı Adı boş değil ve Kullanıcı ID boş değil
hayır
ContentPane'ı temizle
Hata mesajı göster
ContentPane'a panel ekle
Yeniden doğrula
Yeniden çiz
Resim dosyası yolu = "giris.png"
ResimLabel oluştur
ResimLabel boyutu ayarla
Resmi ölçekle

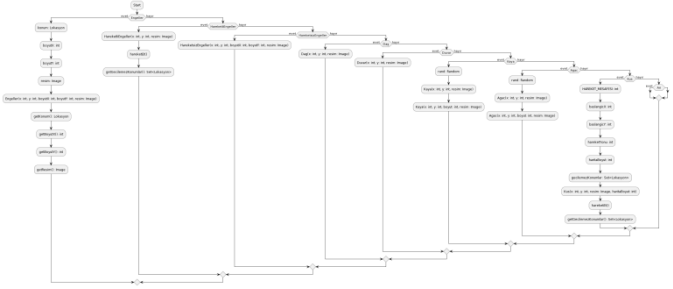

OyunPaneli()
haritayıHazırla()
evet
tumHazinelerToplandı()
hayır
Oyun bitti
hareketEt()
paintComponent(g: Graphics)
getGörünürHücreler()
evet
griBoyama?
hayır
Gri boyama yap
Boya yapma
setKarakter(karakter: Karakter)
setKus(kus: Kus)
setAri(ari: Ari)
setGriBoyama(griBoyama: boolean)
setHaritaBoyut(haritaBoyut: int)
setArkaPlanResmi(arkaPlanResmi: Image)
setAltinResmi(altinResmi: Image)
setGumusResmi(gumusResmi: Image)
setZumrutResmi(zumrutResmi: Image)
setBakirResmi(bakirResmi: Image)
setSnowKayaResmi(snowKayaResmi: Image)
setSunKayaResmi(sunKayaResmi: Image)
setSnowAgacResmi(snowAgacResmi: Image)
setSunAgacResmi(sunAgacResmi: Image)
setSnowDagResmi(snowDagResmi: Image)
setSunDagResmi(sunDagResmi: Image)
setSnowDuvarResmi(snowDuvarResmi: Image)
setSunDuvarResmi(sunDuvarResmi: Image)
setKarakterResmi(karakterResmi: Image)
setKusResmi(kusResmi: Image)
setAriResmi(ariResmi: Image)
isGriBoyama()
enKisaHedefX, enKisaHedefY belirle

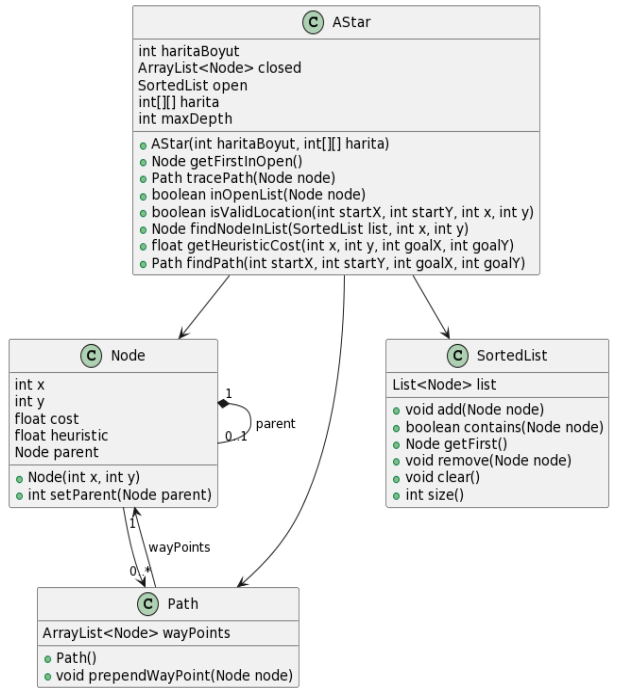


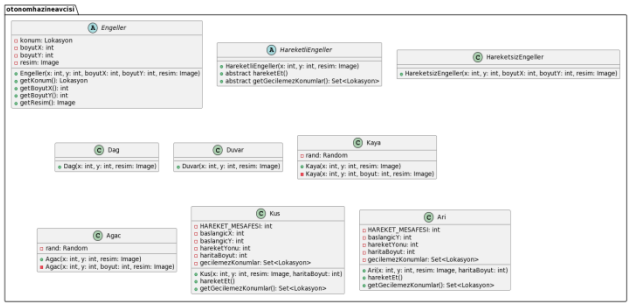


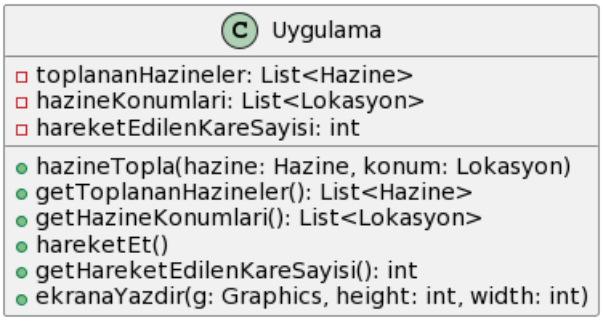
UML DİYAGRAMLARI

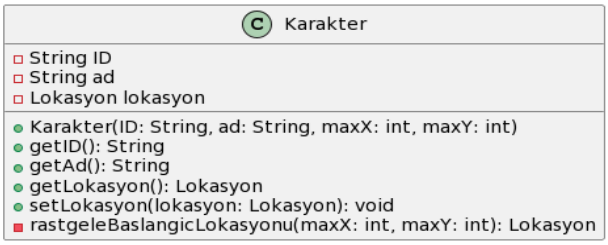












KAYNAKLAR

[1] *Bevor Sie zu YouTube weitergehen*. (n.d.-a). https://youtube.com/playlist?list=PL4rzdwizLaxYmltJQRjq18a9gsSyEQQ-0&feature=shared

[2] *Bevor Sie zu YouTube weitergehen*. (n.d.-b). https://youtube.com/playlist?list=PL_QPQmz5C6WUF-pOQDsbsKbaBZqXj4qSq&feature=shared

[3] *GitHub - marcelo-s/A-Star-Java-Implementation: A*or A Star algorithm java implementation*. (n.d.). GitHub. https://github.com/marcelo-s/A-Star-Java-Implementation

[4] *GitHub - SparshBohra/Pathfinding-Visualizer: GUI Application for A* Search Pathfinding Visualization built using Java Swing*. (n.d.). GitHub. https://github.com/SparshBohra/Pathfinding-Visualizer

[5] *GitHub - vmsaif/ant-path-finding-using-A-Star-algorithm: The world of ants and their incredible pathfinding abilities. This program, written in Java, uses the A* algorithm to determine the shortest path for an ant to reach its food. Witness the efficiency and precision of the A* algorithm in action, as the ant navigates through various terrains and obstacles.* (n.d.). GitHub. https://github.com/vmsaif/ant-path-finding-using-A-Star-algorithm

[6] İzgi, F. (2021, October 21). *Polymorphism Nedir ?* Medium. https://medium.com/kodcular/polymorphism-nedir-a10070eafe67

[7] *Java Constructor Nedir? - Emre Çelen*. (n.d.). Emre Çelen. https://emrecelen.com.tr/java-constructor-nedir/

[8] Selawsky, J. (2022, December 1). *A* Search Algorithm in Java*. CodeGym. https://codegym.cc/groups/posts/a-search-algorithm-in-java

[9] (n.d.). GitHub: Let's build from here · GitHub. https://github.com/martacanirome4/AStarAlgorithm/blob/master/src/aplicacion/Main.java

[10] Tetik, F. (2019, August 30). *Abstract Class Nedir?* Medium. https://medium.com/@denizf.b/abstract-class-nedir-d6529d5f1e9e