Smurfs Maze Game

Ferhat Açıkalın 180202065@kocaeli.edu.tr

Berkay Yasin Çiftçi 190202056@kocaeli.edu.tr

İ. GİRİŞ

Programlama Laboratuvarı-2 I. Projesi olarak bizden "Şirinler Oyunu" adında bir labirent oyunu olan ve bir masaüstü ara yüzü tasarımı içeren bir uygulama yapmamız beklenmektedir.

İİ. PROJENİN ÖZETİ

Biz proje için Java Programlama dilini ve Netbeans geliştirme ortamını seçtik. Java Programlama dili; açık kodlu, nesneye yönelik, zeminden bağımsız, yüksek verimli, çok işlevli, yüksek seviye, adım adım işletilen (yorumlanan-interpreted) bir dildir.

Netbeans platformu; Oracle tarafından geliştirilen bir Java geliştirme ortamıdır(IDE) ve ücretsiz olarak dağıtılmaktadır. Özellikle kullanıcı ara yüzü tasarımında sağladığı kolaylıklardan dolayı tercih edilmektedir. Bizim de Netbeans'i tercih etme sebeplerimizin başında kullanıcıya sağladığı kolaylıklar yatmaktadır.

Proje dokümanında sınıflara göre değişen isterleri bulunmaktadır. Tüm sınıflar için ise 3 adet ortak özellik kullanmamızı istediler.

Projede biz Java programlama dilinde bulunan "JavaSwing" adlı tasarım kütüphanesinden yararlandık. Bu kütüphane sayesinde labirent oluşturmada ve oyuncuların düşmanların takip edilebileceği bir ara yüz tasarımı gerçekleştirilmiştir. İlk olarak classları tanımladık. Classlar için gerekli olan özellikleri, parametreli ve parametresiz olmak üzere her class için en az iki adet constructor (parametreli/parametresiz) ve set get metotlarını tanımladık. Bazı classlar için gerekli olan kalıtım yapıldı. Oyuncu ve Düşman Sınıfı için Karakter Sınıfından, Tembel Şirin, Puan ve Gözlüklü şirin için Oyuncu sınıfından, Azman ve Gargamel için Düşman sınıfından, Altın ve Mantar için Obje sınıfından kalıtım yapılmıştır. Bazı gerekli metotlarda istenen ve gerekli classlara override edilmiştir.

Projemizde Encapsulation, Intheritance, Polymorphism, Abstraction yapılarından gerekli olanlar uygun görülen classlarda kullanılmıştır.

İİİ. TEMEL BİLGİLER

A. YAPILAN ARAŞTIRMALAR VE KARŞILAŞILAN SORUNLAR

Proje geliştirilirken karşılaşılan sorunlardan birkaçı aşağıda belirtilmiştir ve bu sorunları nasıl aştığımız ve yapılan araştırmalardan aşağıda detaylı olarak bahsedilmiştir.

Birinci olarak karşılaşılan sorun; Java Swing kütüphanesini kullanarak haritayı görsel olarak ekrana göstermek oldu. Bunun için gerekli kütüphanenin syntax öğrenildikten sonra haritamızı hazırlamış olduk.

İkinci olarak karşılaşılan sorun; harita.txt dosyasında bulunan 1 ve 0 oluşan harita matrisinin okunması oldu. Bu sorunu çözmek için kısa bir araştırma yaptıktan sonra ilk iki satırı atlayarak matrisi projemize aktarmış olduk.

Üçüncü olarak karşılaşılan sorun; en kısa yol algoritması olarak bizden istenen yani Dijisktra algoritmasını proje entegre etmekte zorlandık. Birkaç araştırma ve deneme yanılma yöntemi ile harita matrisinin komşuluk ilişkilerini bulduktan sonra düğümler halinde komşuluk ilişkileri yapılarak Dijisktra algoritması projemize entegre etmiş bulunduk.

Dördüncü olarak karşılaşılan sorun ise; Komşuluk matrisini nasıl oluşturacağımız hakkındaydı. Bu konu üzerinde birçok yol denedik bu yollardan birkaçını Dijisktra algoritmasına uyarlayamadığımızdan dolayı vazgeçtik. Sonra for döngüsü ile matrisin i ve j değerlerinin bir fazlasına göre 1 olan değerleri bir arrayliste, oluşturulan arraylist de bir hashmap key value olacaka şekilde tutulmuştur.

B. YAZILIM MİMARİSİ

Projede gerekli olan classlar tanımlandı. Classlar oluşturulduktan sonra işlevleri hakkında bilgi sahibi olmak istersek;

Dijkstra.java:

Bu sınıfta harita.txt den okuduğumuz harita matrisinin komşuluk matrisleri bulunduktan sonra Dijkstra algoritmasına uyarlanması için yazılmıştır.

Graph türünde ve static olan enKisayYolHesapla fonksiyonu, kaynak düğümünden en kısa yol bulan bir fonksiyondur. İki adet parametre alan bir fonksiyondur. Bu parametreler node ve graph tipindendir.

minUzaklik, 3 adet parametre alan ve static void tipli bir fonksiyondur. Bu fonksiyon minimum mesafeyi hesaplamaktadır.

Bir diğer fonksiyonumuz ise enAzAralik, 1 adet parametre almaktadır ve static Node tipindedir. Bu fonksiyonumuz düğüme olan en kısa uzaklığı bulmak için yazılmıştır.

LokasyonSinifi.java:

Bu Classımızda x ve y koordinatları tutan iki farklı değişken tanımlanmıştır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

İki adet int tipinde fonksiyonumuz bulunmaktadır.Birisi xy2nodeId ve diğeri xy2nodeIndex tir. Bu fonksiyonlarımızda harita matrisinin seçilen x ve y koordinatlarına göre index değerini döndürmektedir.

KarakterSinifi.java:

KarakterSinifi JButton kalıtım almaktadır. Bu classta ID, Ad, Lokasyon, currentNode ve enum türünde karakterturu adlı değişkenlerden oluşmaktadır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir. Parametreli olan constructor (int ID, String Ad, int x, int y) şeklinde parametre almaktadır.

Graph türünde enkısayol adlı bir parametreli metodumuz bulunmaktadır. Bu metodumuz Dijkstra.enKisayYolHesapla Dijkstra.java adlı classımızın , kaynak düğümünden en kısa yol bulan fonksiyona return etmektedir.

Classımızın başında tanımladığımız değişkenlerin get ve set metotları classımızın en alt kısmında tanımlanmıştır.

DusmanSinifi.java:

DüsmanSinifi classımız KarakterSinifi classından kalıtım almaktadır.

Bu classta DusmanID, DusmanAdi, ve enum türünde DusmanTuru adlı değişkenlerden oluşmaktadır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

OyuncuSinifi.java:

OyuncuSinifi classımız KarakterSinifi classından kalıtım almaktadır.

Bu classta OyuncuID, Skor, OyuncuAdi ve enum türünde OyuncuTur adlı değişkenlerden oluşmaktadır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

PıanıGoster() adlı void bir fonksiyonumuz bulunmaktadır. Bu fonksiyon oyuncuların elde ettikleri puanları göstermeye yarıyor.

moveLocation, moveLeft, moveRigth, moveUp, moveDown, updateGraph adlı fonksiyonlarımı OyuncuSinifindan kalıtım alan classlarda override edilmek üzere yazılmıştır.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

GSirin.java:

GSirin classımız OyuncuSinifi classından kalıtım almaktadır.

Gözlüklü şirin 2 birim ilerleyebildiği için int tipinde birim adında bir değişken tanımlanıp bu değişkenin değeri 2 olarak atanmıştır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

PıanıGoster() adlı void bir fonksiyonumuz bulunmaktadır. Bu fonksiyon oyuncuların elde ettikleri puanları göstermeye yarıyor. Bu fonksiyonumuzu OyuncuSinifi adlı classımızdan override edilerek GSirin classımızda kullanılmıştır.

Void tipli moveLocation adlı int x ve int y olmak üzere iki adet parametre alan bir fonksiyonumuzdur. Bu fonksiyonumz ile karakterimizin hareket edeceği lokasyonu yani x ve y değerlerini alıp lokasyon sınıfına gönderiyor. İf yapısını kullanarak çarpışma olup olmadığı kontrol ediliyor. moveLocation adlı fonksiyonumuz OyuncuSinifi adlı classımızdan override edilerek kullanılmıştır.

Void tipli updateGraph adlı parametresiz bir fonksiyondur. Bu fonksiyonumuz OyuncuSinifi adlı classımızdan override edilerek kullanılmıştır. Dijkstra classımızdaki enKisayYolHesapla adlı fonksiyonumuza gerekli olan parametreleri iletir.

Void tipli moveLeft, moveRight, moveUp, moveDown adlı parametresiz fonksiyonlarımızdır. Bu fonksiyonlarımız OyuncuSinifi adlı classımızdan override edilerek kullanılmıştır. Adından da anlaşılacağı üzere karakterimizin hangi yöne gittiğine dair gerekli hesaplamalar yapan bir fonksiyonumuzdur.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

Gargamel.java:

Gargamel adlı classımız DusmanSinifi adlı classımızdan kalıtım almaktadır.

Gargamel 2 birim ilerleyebildiği için int tipinde birim adında bir değişken tanımlanıp bu değişkenin değeri 2 olarak atanmıştır. Node tipinde currentNode adında bir değişken tanımlanmıştır. Bulunduğu konumdaki düğümü belirtmek için kullanılacaktır.sPath adında LinkedList<Node> tipinde bulunan değişkenimiz ise düğümler için oluşturulan bağlı listedir.spathElementList adında ArrayList<JButton> tipinde değişkenimiz ise Java Swing kütüphanesinin bir metodu olan JButton için ayrılmış bir arraylisttir.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

Void tipli moveLocation adlı int x ve int y olmak üzere iki adet parametre alan bir fonksiyonumuzdur. Bu fonksiyonumz ile karakterimizin hareket edeceği lokasyonu yani x ve y değerlerini alıp lokasyon sınıfına gönderiyor. İf yapısını kullanarak çarpışma olup olmadığı kontrol ediliyor. Diğer fonksiyonumuz ise void tipli moveOn adlı fonksiyonumudur. Bu fonksiyonumuz ise oyunucyu yakalayıncaya kadar devam etmesi için

yazılmış bir fonksiyondur.Bir diğer fonksiyonumuz olan spath_Draw ise karakterimizin konumunu ve görüntüsünü ekranda kullanıcıya göstermek amaçlı yazılmıştır.spath_Clear ise haritadan sıfırlamak amaçlı kullanılan bir fonksiyondur.resetlevel ise en kısa yolu temizler.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

Graph.java:

Node tipinde set yapısında oluşturduğumuz hashset bulunmaktadır.

Void tipli nodeEkle adlı parametreli bir fonksiyonumuz vardır. Bu fonksiyonumuz oluşturduğumuz node tipindeki kümeye düğüm eklemek için kullanılır.

Void tipli clear adlı parametresiz bir fonksiyonumuz bulunmaktadır. Oluşturduğumuz node tipinde set yapısındaki NodeSet içeriklerini silmemiz için kullanılır.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır

Map2Node.java:

Bu Map2Node.java classımız harita.txt dosyasından okunan matrisin komşuluk matrisini bulmamızda kullandığımız bir classtır.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğer parametresizdir. Parametreli olan fonksiyon ile harita.txt den okunan matrisin 2 boyutlu diziye atıldıktan sonra classımızda bulunan iki boyutlu dizi adlı değişkene atmak için kullandık.

İnt tipinde sayaca adında ve Node tipinde ArrayList değişkenleri tanımlanmıştır. Sayaca adındaki değişkenimiz iki boyutlu dizide soldan sağa olacak şekilde düğümleri numaralandırmak için kullandığımız bir değişkenimizdir.

Void tipli parseMap adlı parametresiz bir metodumuz bulunmaktadır. Bu metodumuz ile Node.java adlı classımızın parametreli constructorına ıd,x,y değerlerine karşılık olarak sayaca,j,i değerlerini gönderir.Ve bu oluşan düğümü NodeList adlı ArrayListe ekleme yapar.Bu metotdumuzda komşuluk ilişkileri bulunup NodeListe eklenir.Bunu sadece iki boyutlu dizideki 1 değerleri için yapar.

Void printMap adlı fonksiyonumuz ise bulduğumuz komşılık değerelerini konsola bastırmamıza yaramaktadır.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

Node.java:

Bu Node.java classımız harita.txt dosyasından okunan matrisin komşuluk matrisini bulunduktan sonra düğümlerin komşuluk bağlantıların yapılmasında kullandığımız bir classtır.

Classımızda LokasyonSinifi tipinde lokasyon adında, int tipinde id adında, KarakterSinifi tipinde placeholder adında, objeholder adında ObjeSinifi tipinde , Node tipinde LinkedList en Kisa yol adında , uzaklık adında int tipinde , komsular adında hashmap adlı değişkenlerimiz bulunmaktadır.

İki adet constructor bulunmaktadır. Bunlardan biri parametreli diğeri parametresizdir.

Void tipli komsuEkle adındaki metodumuz bulunmaktadır. Bu metodumuz gelen düğüm ile uzaklığın komsular adlı hashmap e eklenmesi için kullanılır.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

TSirin.java:

Void tipli puaniGoster adlı metotdumuz OyuncuSinifi adlı classımızdan overrde edilmiştir.

Tembel şirin 1 birim ilerleyebildiği için birim adında int tipinde bir değişken oluşturularak bu değişkene bir değeri atanmıştır.

Parametreli constructımız bulunmaktadır. Tsirin adlı classımız OyuncuSinifi adlı classımızdan kalıtım almaktadır.moveLocation adında parametreli bir fonksiyonumuz bulunmaktadır. Bu fonksiyonumuz karakterimizin hareket lokasyonu için kullanılmaktadır..Karakterimizin sağa sola yukarı ve aşağı hareket etmesi için oluşturulmuş 4 adet fonksiyonumuz bulunmaktadır. Karakterimizin yaptığı hareketler sonucu arayüzü güncellemek için updateGraph adlı fonksiyonumuz kullanılmıştır.

DosyaOkuma.java:

DosyaOkuma adlı classımız bizim harita.txt de tutulan harita matrixi ve kötü karakter olan gargamelin hangi kapıdan gireceğini belirtilen string dosyalarının hafızada tutularak projemize entegre edilmiştir.

Harita_kaynak adlı iki boyutlu değişkenimiz bulunmaktadır. Bu değişkenimiz harita.txt dosyasındaki matristeki kadar satır ve sütün sayısı içermektedir ve harit.txt dosyasındaki matrisi bu değişkenimizde saklanacaktır.

Okuma adlı void tipli bir fonksiyonumuz bulunmaktadır. Bu fonksiyonumuz ile ilk iki satırı ayrı ayrı diğer kalan satırları ise tamamını bir diziye olacak şekilde atma işlemi gerçekeleştirmektedir.

Yazdirma adlı void tipli ve parametreli olan bir fonksiyonumuz bulunaktadır. Bu fonksiyonumuz ise harita.txt dosyasından okuma işlemi yapılıp bu okuma işlemi sonucu ortaya çıkan dizileri terminalde yazdırmak amaçlı kullanılmıştır.

Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

Main.java:

Main classımız bizim projemizin çalışmasınıa sağlayan classımızdır. Bu classımızda diğer classlarda yazılan metotların kullanarak projemiz çalışır hale getirilmiştir.

Java Swing kütüphanesinden olan JFrame , JPanel gibi değikenlerimiz bulunmaktadır.Diğer değişkenlerimizin işlevleri de adlarından anlaşılmaktadır. Örnek verecek olursak oyun bitti adlı boolean tipli değişkenimiz oyunun bitip bitmediğini kontrol etmek için kullanılmaktadır.

Diger bir örnek ise düşmanlar adında DusmanSinifi oluşan ArrayList bulunmaktadır. Dusman karakterler için gerekli fonksiyonların çağrılıp kullanılması için yazılmıştır.

Void tipli main adlı programımızı çalıştırmamıza yarayan fonksiyonumuz bulunmakradır. Bu fonksiyonumuzda gerçekleşen işlemler detaylı olarak kaba kod (yalancı kod) kısmında anlatılacaktır.

setPuandurumu adlı void tipli fonksiyonumuz ise oyunumuzda sol üst köşede bize puanımızı göstermeye yarayan ve bu puanı hesaplayan gerekli fonksiyonları içeren bir metodumuzdur.

Keypressed ve key Released adlı fonksiyonlarımız bulunmaktadır bunlar void tipli fonksiyonlarımızdır. Bu fonksiyon tuşlara basıldığında çalımaktadır bir diğer deyişle klavyeden girilen değerleri okumaya yarıyor.

AZMAN.java:

Azman adlı classımız DusmanSinifi adlı classımızdan kalıtım almaktadır.

Azman 1 birim ilerleyebildiği için int tipinde birim adında bir değişken tanımlanıp bu değişkenin değeri 1 olarak atanmıştır. Node tipinde currentNode adında bir değişken tanımlanmıştır. Bulunduğu konumdaki düğümü belirtmek için kullanılacaktır.sPath adında LinkedList<Node> tipinde bulunan değişkenimiz ise düğümler için oluşturulan bağlı listedir.spathElementList adında ArrayList<JButton> tipinde değişkenimiz ise Java Swing kütüphanesinin bir metodu olan JButton için ayrılmış bir arraylisttir.

İki adet constructor bulunmaktadır bunlardan biri parametreli diğeri parametresizdir.

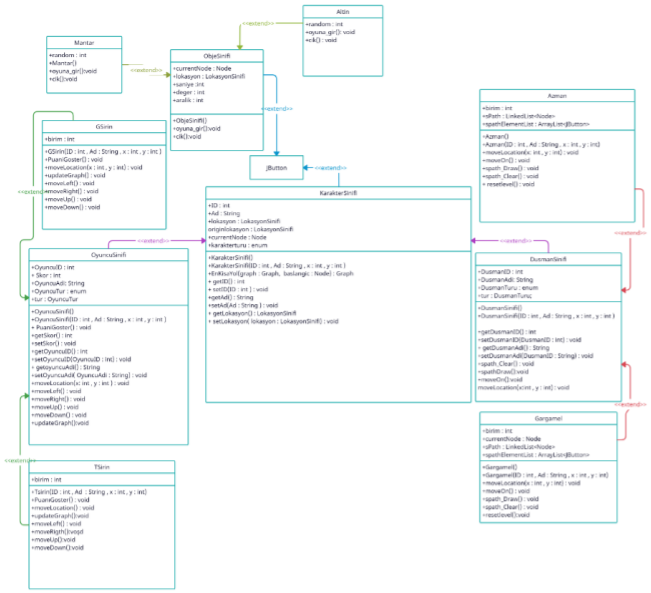
Void tipli moveLocation adlı int x ve int y olmak üzere iki adet parametre alan bir fonksiyonumuzdur. Bu fonksiyonumz ile karakterimizin hareket edeceği lokasyonu yani x ve y değerlerini alıp lokasyon sınıfına gönderiyor. İf yapısını kullanarak çarpışma olup olmadığı kontrol ediliyor. Diğer fonksiyonumuz ise void tipli moveOn adlı fonksiyonumudur. Bu fonksiyonumuz ise oyunucyu yakalayıncaya kadar devam etmesi için yazılmış bir fonksiyondur.Bir diğer fonksiyonumuz olan spath_Draw ise karakterimizin konumunu ve görüntüsünü ekranda kullanıcıya göstermek amaçlı yazılmıştır. spath_Clear ise haritadan sıfırlamak amaçlı kullanılan bir fonksiyondur.resetlevel ise en kısa yolu temizler.

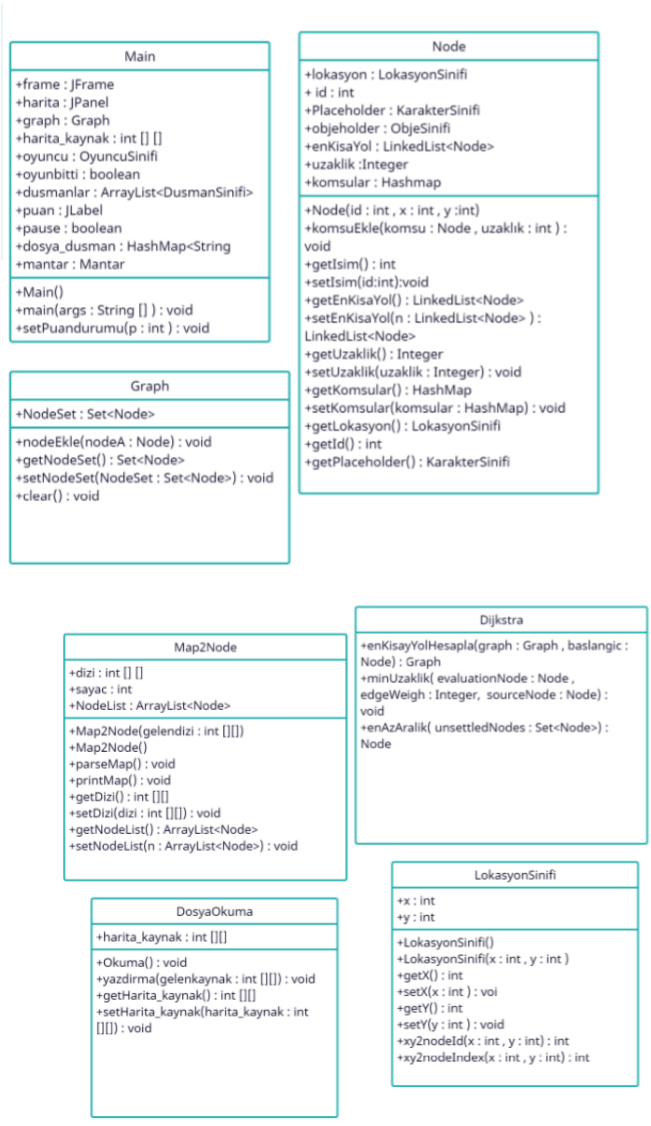
Tanımladığımız değişkenlerin gerekli set ve get metotları yazılmıştır.

C. YALANCI KOD

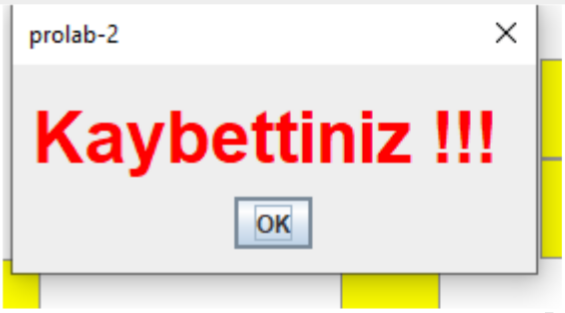
1. Başla
2. Harita.txt dosyasını oku
3. Düşmanlari ve kapılarını al
4. dosya_dusman'a ekle
5. harita bilgisini çözümle
6. Harita bilgisini harita_kaynak dizisine ekle
7. Arayüz bileşenlerini oluştur
8. Map2Node sınıfı ile harita bilgisine göre nodeları oluştur
9. Komşusu olmayan nodeları ekrana duvar olarak ekle
10. Şirine karaketerini ekrana ekle
11. Oyuncu seçimi yap
12. Oyuncuyu başlangıç düğümüne koy
13. Oyuncu başlangıç konumuna getir
14. Oyuncuyu ekrana getir
15. Puan bilgisini göster
16. Kapi konumlarını tanımla
17. Düşman karakterleri ilgili kapılara koy
18. Düşman karakterleri konumlarına göre ait oldukları düğüme ekle
19. Düşman karakterleri ekrana getir
20. Kullanıcın klavye komutlarını alan KeyListener oluştur.
21. Tuşların hareket işlevlerini tanımla
22. Mantar objesini oyuna eklemek için zamanlayıcı oluştur
23. Altın objesini oyuna eklemek için zamanlayıcı oluştur
24. Oyun döngüsünü başlat
25. En kısa yol hesapla
26. Düşmanları oyuncuya olan en kısa yolda hareket ettir.
27. Oyuncu düşmalara değerse puan düşür
28. Oyuncu obje toplarsa puan arttır
29. Oyuncu hedef konuma gelirse oyunu bitir
30. Oyuncu.skor>0 ve oyunbitti==false ise 24e git
31. Oyun bitti diyaloğunu göster
32. Bitir

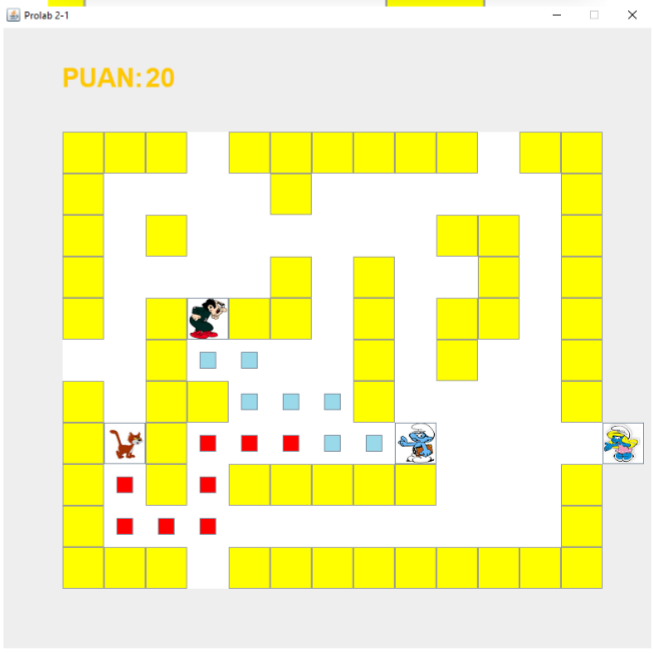
D. UML SINIF DİYAGRAMI

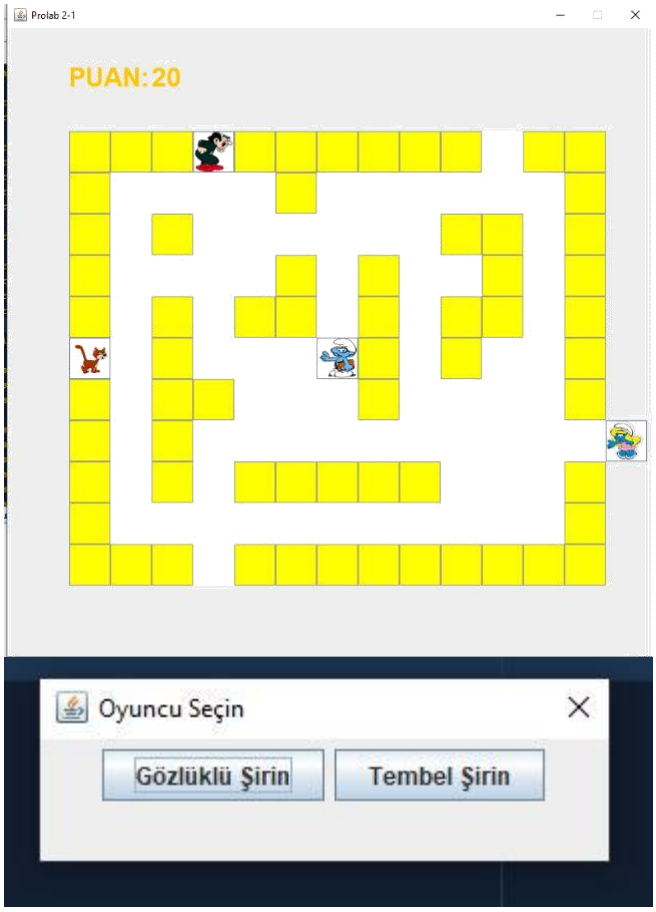




E. DENEYSEL SONUÇLAR







F. ZAMAN VE BELLEK KARMAŞIKLIĞI ANALİZİ

Projede kullandığımız dijkstra algoritması greedy bir algoritmadır. En iyi çözümü seçerek ilerler. Başlangıçta tüm düğümlerin uzaklığını sonsuz olarak belirledik. Başlangıç düğümünün uzaklığını 0 olarak belirledik. Başlangıç düğümünden son düğüme kadar graph içindeki her düğüm için en kısa mesafe hesaplandı. Graftaki herbir düğümün komşularının listesi o düğüm içerisinde oluşturuldu. Böylece algoritma her seferinde komşuları aramak zorunda kalmadı yada bir komuşuluk matrisinde komşularını aramadı. E kenar sayısı olsun, V de düğüm sayısı olsun. En kötü senaryoda n tane düğüm ve n tane kenar olsun. Bu durumda komşuluk listesi olmasaydı karmaşıklık O(n^2) olurdu. Fakat komuşuluk listesi kullanarak bunu O(ElogV) ye düşürdük. Algoritmayı bellek karmaşıklığı yönünden incelersek, Düğümleri ve düğümlerin komşularını bellekte tuttuğumuz için en kötü senaryoda bellek karmaşıklığı O(V)+O(V) =O(V) şeklinde ifade edilebilir.

G. SONUÇ

Bu proje sayesinde java programlama dilini kullanarak nesneye yönelik programlama mantığını (kalıtım, classlar, nesneler, methodlar) ve grafik kütüphanesi kullanımı konusunda bol bol bol bilgi sahibi olduk. Dijkstra algoritması ile en kısa yol hesaplanmasını ve projemize nasıl entegre edeceğimizi öğrendik. En sonundada bir oyunun ne kadar zorluklarla yapıldığını bu projede benimsemiş olduk. Teşekkür Ederiz.

H. KAYNAKÇA

https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/#:~:text=Dijkstra's%20shortest%20path%20algorithm%20%7C%20Greedy%20Algo%2D7&text=Given%20a%20graph%20and%20a,algorithm%20for%20minimum%20spanning%20tree.

https://www.softwaretestinghelp.com/dijkstras-algorithm-in-java/

https://www.javatpoint.com/java-swing

https://www.baeldung.com/java-dijkstra

https://www.geeksforgeeks.org/java-program-for-dijkstras-shortest-path-algorithm-greedy-algo-7/

https://www.geeksforgeeks.org/implementing-generic-graph-in-java/