KOCAELİ ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMLAMA LAB II PROJE 2

190201074 FATİH GÜNER

190201076 KEREM KARATAŞ

Summary: Projede bizden istenen OOP programlama paradigmasını kullanarak belirlenen veri yapıları algoritmalarıyla tasarladığımız bir arayüzle seçtiğimiz bir şirinin(tembel, gözlüklü) şirineye ulaşmasıdır. Ulaşırken karşısında birtakım zorluklar vardır bunlar azman ve gargamel düşman karakterleridir. Aynı zamanda karşısına çıkan zorluklar dışında şirineye ulaşmaya çalışırken belirli frekanslarda üretilen altın ve mantar objeleri beklemektedir. Bu iki obje türünü alabildiği taktirde puanı yükselecektir. Seçilen şirin default olarak 20 puanda başlayacaktır.Objelerin özelliklerine göre puanları farklıdır.Aynı zamanda düşman karakterlere rastladığı taktirde puanı düşmanına göre belli bir sayıda azalacaktır.Eğer oyuncunun puanı 0'ın altına düşerse oyun sona erecektir aksi taktirde devam edecektir.Oyuncunun oyuna başladığı nokta sabit olup keyboard komutlarıyla hareket sağlanacaktır.Verilen txt dosyasından düşman karakterlerin girebileceği kapılar ve mapi oluşturmamız istenmektedir.

Introduction: Projeyi yaparken ide olarak Netbeans idesini kullandık.Arayüz tasarımı için ise swing kütüphanesinden yararlandık. Java Swing, zengin bir widget kümesi içeren hafif bir Grafik Kullanıcı Arabirimi (GUI) araç takımıdır. Paket, Java uygulamalarımız için GUI bileşenleri yapmamızı sağlar ve platformdan bağımsızdır.

Method(Classes and descriptions used):

- Main.java
- SmurfGame.java
- Fonksiyonlar.java
- Karakterler.java
- Oyuncu.java
- GozlukluSirin.java
- TembelSirin.java
- Dusman.java
- Gargamel.java
- Azman.java
- Obje.java
- Altin.java
- Mantar.java
- DijkstraAlgorithm.java

- ✓ Main.java: SmurfGame classı için bir nesne üretip boş constructor şeklinde oluşturur. Oyunumuz başlar.
- ✓ SmurfGame.java: Bu classımız oyunumuzun oynandığı classtır. Bu classta diğer classlarımız için oluşturduğumuz nesneler vardır. Jframe extends edilip KeyListener ve ActionListener implements edilmiştir.Mapimizi Fonksiyonlar.java'da okuyan bir method vardır.Onun için bir obje oluşturup mapimizi oluşturuyoruz.Gerekli if conditionlarıyla düşmanları ve kapıları belirliyoruz.KeyListener implement ettiğimiz için onunla birlikte gelen keyPressed methodu override edildi ve bizde içini doldurduk.Örnek verecek olursak eğer klavyeden sol yönüne basıldıysa gerekli if koşullarıyla renklere göre duvar olup olmadığını kontrol ettirip ona göre oyuncunun(Tembel şirin, Gözlüklü şirin) kendi özelliklerine göre harereketi sağlanacaktır(bu diğer key hareketleri için de geçerlidir).Ve tüm bunlar olurken tüm actionları dinleyen actionPerformed methodu ActionListener implement ettiğimizden override olarak geldi.Bu method bizim acitonlarımızı dinlediği için oyun başlar başlamaz bu method otomatik bir şekilde çalışacaktır. Bu methodda önceden üstte global olarak belirlediğimiz delayaltin,delaymantar özelliklerini kullandık. Gerekli if koşullarıyla delayaltin%.... ==0 (aynı olay mantar içinde) yaparak saniyemizi ayarlayıp bu conditionların içinde duruma göre rastgele 5 atlın oluşturup devamında silip sonra mantar oluşturup sildik.Bu oluşma ve silme methodları Altın.java Mantar.java classlarında bulunmaktadır.Üstte global olarak bu classlar için nesne üretmiştik.
- ✓ Fonksiyonlar.java: Bu classımız da verilen harita.txt dosyası için var olan txt okuma fonksiyonları ve buna bağlı olarak multi-dimensiol button mapini SmurfGame.javaya return eden bir method vardır.Buna ilaveten dijkstraya ait bir map tasarlayan method da vardır.
- ✓ Karakterler.java: Bu classımızın diğer classlarımızın aksine abstract class olma özelliği vardır.Ortak özellikleri olan nesneleri bir çatı altında toplamak temel amaçtır. Oyuncuadı oyuncuId, Img, x,y gibi özellikleri vardır.Bu sınıftan inherit alan diğer sınıflar bu özelliklere ve bu sınıfın constructor'ına sahip olacaklardır.

- ✓ Oyuncu.java: Bu classımız Karakterler.javayı extends etmektedir.super keywordü constructoru'nun içinde bulunmaktadır bunlara ek olarak x ve y nin değerleri burada atılmıştır.Oyun başlarken default gelen skor özelliği de burada int skor=20 olarak tutulmaktadır.

- ✓ TembelSirin.java: Bu classımızda Oyuncu.java classını extends etmektedir.Constructorında süper keywordü ile birlikte Karakterler.java da oluşturulan Img özelliği burada newlenerek içine Tembel şirinimizin resmini koyduğumuz ve boyutunu butonlara göre ayarladığımız Apı'leri çağırdık.Ek olarak setOyuncuId'yi bu şirin için 1 yaptık.

- ✓ GozlukluSirin.java: Bu classımızda TembelSirin.javadan farklı olarak setOyuncuId'si 2 olara tuttuk ve ımage nesnesinde Gözlüklü şirinimizin resmini getirdik.

- ✓ Dusman.java: Bu classımızda tıpkı karakter sınıfımız gibi abstract classtır aynı zamanda Karakterler sınıfını extends etmektedir.İçerisinde damage, x1,y1,x2,y2,x3,y3,x4,y4 özellikleri vardır. Constructorunda süper keywordüyle bu koordinatlara değer atanmıştır. Bu değerler haritayı okurken ki düşman karakterlerin kapıları belirlerken gideceği yer/yerler olacaktır.

- ✓ Gargamel.java: Bu classımız Dusman.java classını extends etmektedir.Constructorda süper keywordünün dışında tıpkı şirinlerimiz için özelleştirdiğim resimler gibi buradada img nesnesine gargamel resmini çağırıyoruz.Dusman classında bulunan damage burada gargamel için projede verilen değeri eşitliyoruz. Kimlik farklılığı olsun diye oyuncuıd'sine 1'i atıyoruz.

- ✓ Azman.java: Bu classımız Gargamel.javadan farklı olarak constructorunun içinde img olarak azman resmini getiriyor ve oyuncuıd'sine 2'yi atıyoruz.

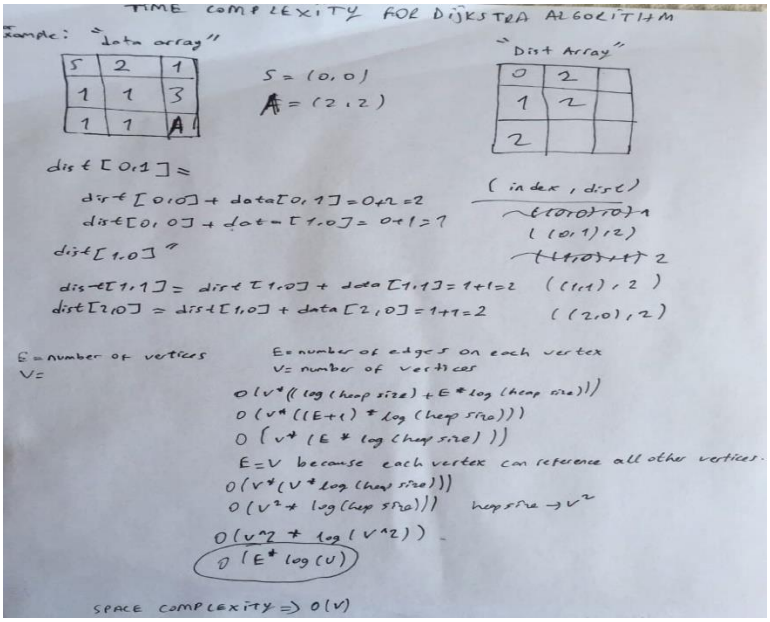
- ✓ Obje.java: Bu classımız abstract classtır.İçinde gömülü olan Random sınıfından, Image sınıfından obje oluşturduk.Bunları oluşturduğumuz Altin.java ve Mantar.java classlarında kullanmak için oluşturduk.Buna ilaveten bu iki classta kullanmak için objeskor özelliğini tanımladık.

- ✓ Altin.java: Bu classımız Obje sınıfından extend etmektedir. Constructorunun içinde img nesnesine altın resmini oluşturduk ve objeskor özelliğine 5'i atadık. Bu classta altinOlustur ve altinSil methodları mevcuttur;her ikisi de parametre olarak Jbutton[][] type'sinde bir nesne almaktadır.

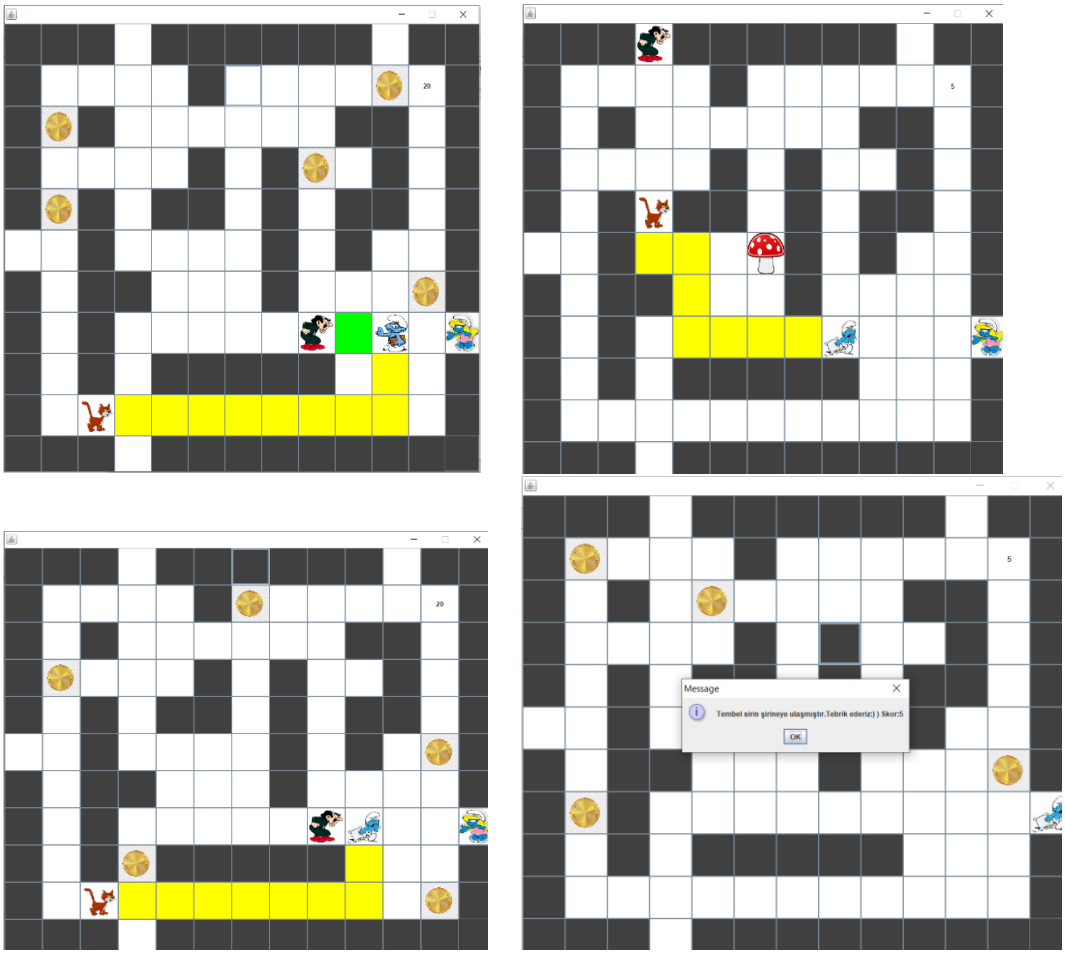
- ✓ Mantar.java: Bu classımız Altin.java'dan farklı olarak constructorunda kendi img nesnesine mantar resmini atadık ve objeskor özelliğine 50 değerini atadık.mantarolustur ve mantarSil methodları mevcuttur ve ikisi de Jbutton[][] type'sinde bir nesne almaktadır.

- ✓ **Lokasyon.java**: Bu classımızda gargamelin ve azmanın koordinatlarının tutan özellikler tuttuk ve getter setterleri yazdık.

- ✓ **DijkstraAlgorithm.java:** Bu classımızda veri yapısı algoritması olan(Greedy Algorithm) Dijkstra Algoritması kullanılmıştır.Burada özellik oalrak vertex sayısı belirlenmiştir.Butonların dijkstra yolundaki butonlara attığımız ,sourceden bütün vertexlere olan uzaklığı hesapladığımız geçilen bir yol var ise true false değerlerde uğraştığımız boolean dizimiz,minimum yolu hesapladığımız gibi methodlar vardır.Karmaşıklığı aşağıdadır.



Experimental Results:



Result: Yazdığımız kodu arayüze yansıtma konusunda deneyimler kazandık.Kalıtım,abstraction gibi konseptleri daha iyi deneyimle şansı bulduk. Projeyi hazırlayan hocamıza teşekkürlerimizi sunarız.

References:

WebSite:

- (1) https://www.javatpoint.com/java-jbutton
- (2) https://www.programiz.com/dsa/dijkstra-algorithm
- (3) https://tugrulbayrak.medium.com/uml-class-diyagramlari-4c3bb7e9cc4c

UML CLASS DİAGRAM