**Kocaeli Üniversitesi**

**Bilgisayar Mühendisliği Bölümü**

**Programlama Laboratuvarı II**

*Tunay Baştürk – Umut Kılıç*

[*190201032@kocaeli.edu.tr-190201028@kocaeli.edu.tr*](mailto:190201032@kocaeli.edu.tr-190201028@kocaeli.edu.tr)

**Projenin Özeti:**

*Bu proje ile bizden labirent içerisinde seçilen oyuncunun puanları bitmeden önce şirine ulaşmasını isteyen nesneye yönelik programlama ve veri yapıları algoritmalarını kullanarak oyun tasarlanmamız istenmiştir. Kullanıcı oyuna başlamadan önce seçim ekranından bir şirin seçmesi gerekmektedir .Seçilen şirin 20 puanla oyuna başlayacaktır ve şirinin puanı 0 ya da 0 ‘ ın altına düştüğünde oyun kaybetme ile bitececiktir. Seçilen şirin ile klavyenin w – a – s – d tuşları ile kontrol ederek şirine ulaşmaya çalışacaktır ancak txt den okutularak yerleri belirlenen gargamel ve azmandan en az birinin onu durdurmaya çalışacaktır. Gargamel ve azman dört kapıdan birinden içeri girecektir. Oyuncu oyuna başladıktan sonra oyuncu ile arasındaki en kısa mesafeye göre ilerleyecektir . Bu ilerlemeyi dijisktra algoritması kullanarak en kısa yolu bulacaktır . Biz java programlama dilini kullandık . Arayüz olarak da swing kullandık .*

**1.GİRİŞ:**

Biz proje için java proglamlama dili ve netbeans geliştirme ortamını kullandık.

Java programlama dili :

**Java Basittir**; Başka dillerde çok zor yazılan işlemler Java’da çok basit ifadelerle gerçekleştirilebilmektedir

**Java Nesneye Yöneliktir;** Diğer bazı diller gibi nesneye yöneliklik sonradan dile eklenmemiştir, başından beri Java’da bulunmaktadır.

**Java Dağıtıktır;** Java ‘dağıtık ’bir dildir. ‘Dağıtık’ birden fazla bilgisayarda çalışan programların birbirleriyle uyumlu çalışabilmesidir.

**Java Güvenlidir;** Java güvenli bir dildir. Java diliyle virüs yapılamaz. Hiçbir virüs Java programına bulaşamaz.

**Java Taşınabilirdir;** Java programları her ortamda aynı ve benzer bir şekilde çalışır. Her ortam/makine için ayrı bir program yazmaya gerek kalmaz.

**Java Yorumlanır;** Java yorumlamalı bir dildir. Yani bir java programının komutları, çalışırken makinenin anlayacağı formata çevrilir.Java’da bu Java Virtual Machine(JVM) tarafından yapılır.

**Java Dinamiktir;** Java'da bir programla kullandığı birimlerin (kütüphaneler,modüller veya sınıfların) birbirine bağlanması çalıştırma anında yapılır. Buna 'sonradan bağlama' (late binding) denir. Kullanılan birimlerin iç yapısı değiştirildiğinde, bu birimleri kullanan programın değişmesi gerekmez.

**NetBeans geliştirme platformu:**

Java diliyle hazırlanmış bir IDE’dir ve Java Virtual Machine (JVM) aracılığı ile Solaris, Mac OS ve Linux gibi popüler işletim sistemi üzerinde sorunsuz çalışabilir. NetBeans’in içerisinde yer alan özellikler ve bileşenler NetBeans’i yazılım geliştirmek için daha işlevsel bir ortam haline getirir. NetBeans’in sunduğu başlıca özellikleriyse şöyle sıralayabiliriz;

Ücretsizdir ve Türkçe dilini destekler

Kullanıcı tabanlı ayarlama yapılabilir

Pencerelerin yerleşimi ve görünümleri düzenlenebilir

NetBeans Visual Library’e sahiptir

Depolama özelliği mevcuttur

Tümleşik geliştirme araçlarına sahiptir

Framework Wizard / Kütüphane Sihirbazı bulunur.

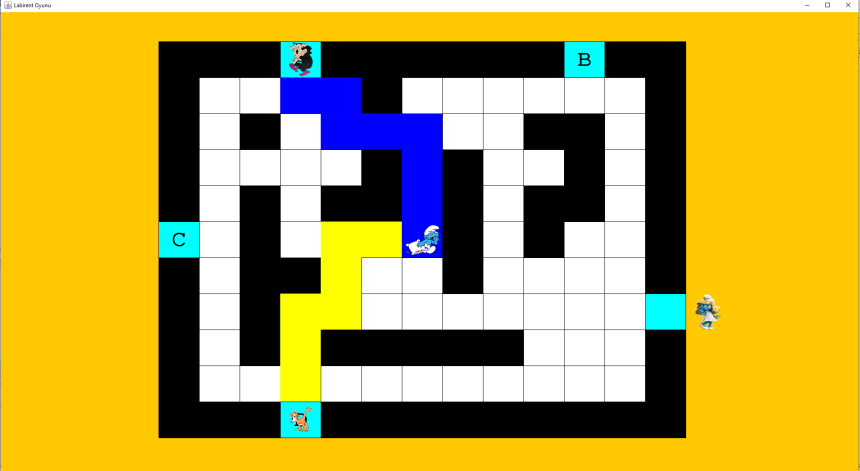
**2.YÖNTEM:**

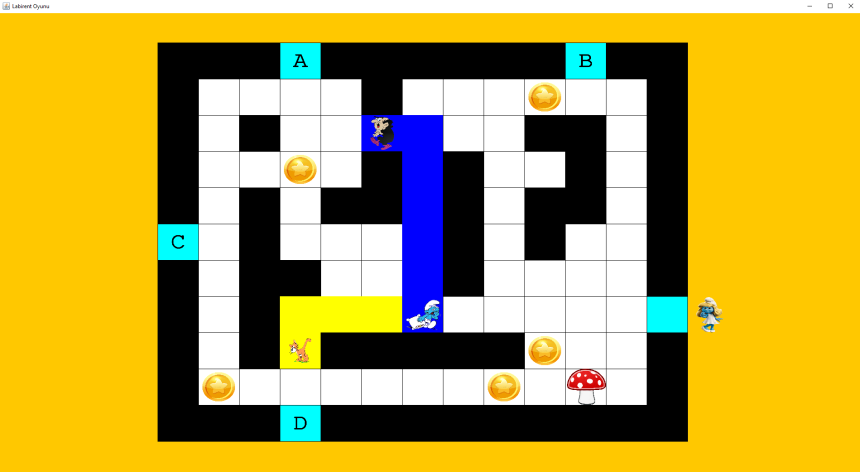
Labirent map classı’nın yapıcı metodunda try catch ile txt den okuma yaparak haritayı labirent adlı matrise atarız.Graf sınıfından nesne adlı nesne oluşturulur.Graf sınıfındaki bağlama fonksiyonunda tek tek düğümler birbirlerine bağlanır , yollar çift yönlü bağlanır , duvarlar ise hiçbir düğüm ile bağlanmaz , kapılar ise tek yönlü bağlanır.Bu oluşturduğumuz düğümler oluşturduğumuz nesne adlı nesneye atanır ve bu nesne döndürülür. Komşular fonksiyonu ile yollar sonuç dizisine eklenir. Mesafe ,önceki ve ziyaretettimi isimli liste düğüm sayısı kadar eleman tanımlanır. İlk uzaklık sıfır olarak atanır. Enküçükkose fonksiyonu çağırılır. Sonraki düğüm ziyaret edilmediyse ve uzaklık değeri a dan küçük ise b ye eşitlenir ve a değeri küçülür. Döngü de mesafe yani düğüm sayısı kadar döndürülür ve içeride if ile ziyaretettimi ve oranın mesafesinin a mesafesinden küçük olup olmadığı sorgulanır. Eğer bu şartları sağlıyorsa yeni a mesafesi o indexteki mesafemiz olur. B ‘ ye de o index atanır ve bu enkucukkose fonksiyonunda b döndürülür. Graftaki komşular dizisinin içine sonraki düğüm atılır ve bir sonraki olan elamana uzaklığı hesaplatılır . Önceki isimli dizi geri döndürülür.Karakter sınıfında bulunan en kısa yol adlı fonksiyonda bulunan kısayol adlı arraylist döndüren fonksiyon sayesinde en kısa yolu buluruz ve bu fonksiyonu labirentmap classında karakter sınıfından karakter nesnesi oluşturulur. Labirentmapta gargamelyol ve azmanyol adlı iki tane arraylist tanımlanır ve bu arraylistlere bu oluşturduğumuz karakter sınıfındaki kısa yol fonksiyonları eşitlenerek yolu kullanılabilir hale getiririz.

**3.DENEYSEL SONUÇLAR:**

***Seçim Ekranı:***

******

******

**İki adım gittikten sonra :****

**Kazanma ekranı:**



**Kaybetme ekranı:**



**4-Zaman karmaşıklığı**

public static int[] EnkisaYol(Graf G, int x) {

1 final int[] mesafe = new int[G.size()];

1 final int[] onceki = new int[G.size()];

1 final boolean[] ziyaretettimi = new boolean[G.size()];

N for (int i = 0; i < mesafe.length; i++) {

mesafe[i] = Integer.MAX\_VALUE;

}

1 mesafe[x] = 0;

N for (int i = 0; i < mesafe.length; i++) {

1 final int sonraki = enKucukKose(mesafe, ziyaretettimi);

1 ziyaretettimi[sonraki] = true;

1 final int[] n = G.Komsular(sonraki);

N for (int j = 0; j < n.length; j++) {

1 final int v = n[j];

1 final int d = mesafe[sonraki] + G.agirlik(sonraki, v);

N if (mesafe[v] > d) {

N mesafe[v] = d;

N onceki[v] = sonraki;

}

}

}

1 return onceki;

}

private static int enKucukKose(int[] mesafe, boolean[] v) {

1 int a = Integer.MAX\_VALUE;

1 int b = 0;

N for (int i = 0; i < mesafe.length; i++) {

N if (!v[i] && mesafe[i] < a) {

N b = i;

N a = mesafe[i];

}

}

1 return b;

}

public static ArrayList KisaYol(Graf G, int[] onceki, int t, int u) {

1 final java.util.ArrayList yol = new java.util.ArrayList();

1 int x = u;

N while (x != t) {

N yol.add(0, G.getLabel(x));

N x = onceki[x];

}

1 yol.add(0, G.getLabel(t));

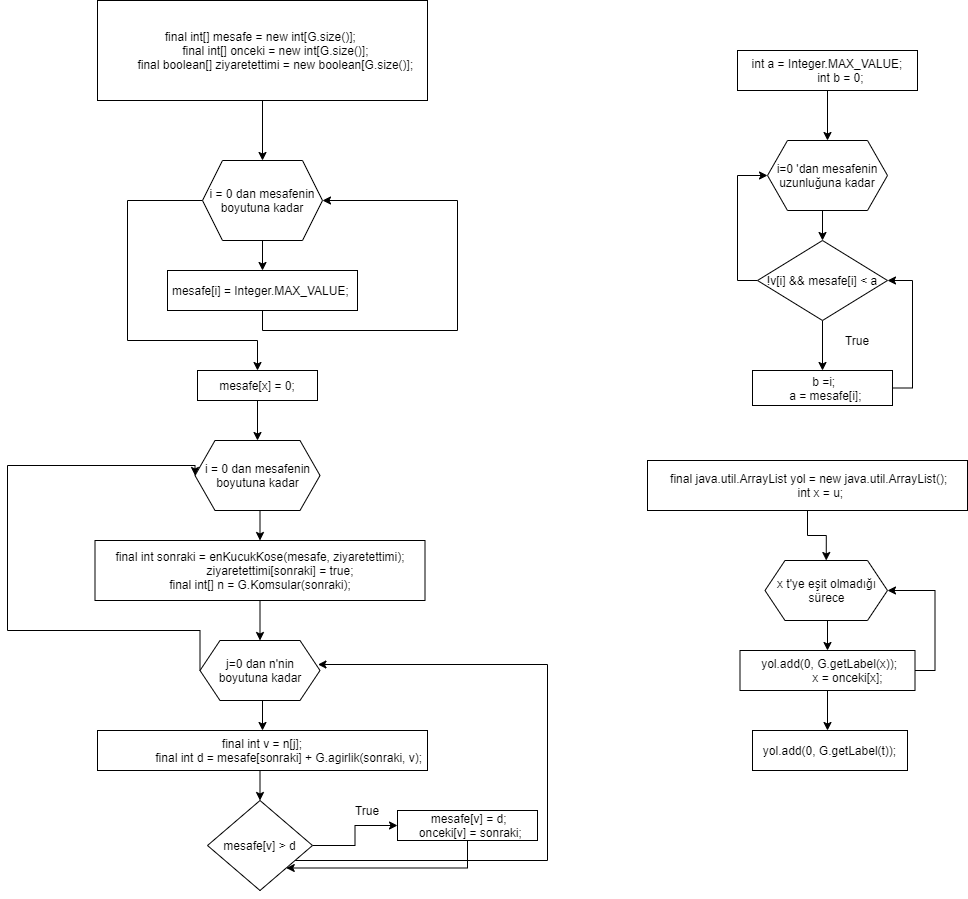
1 return yol;

}

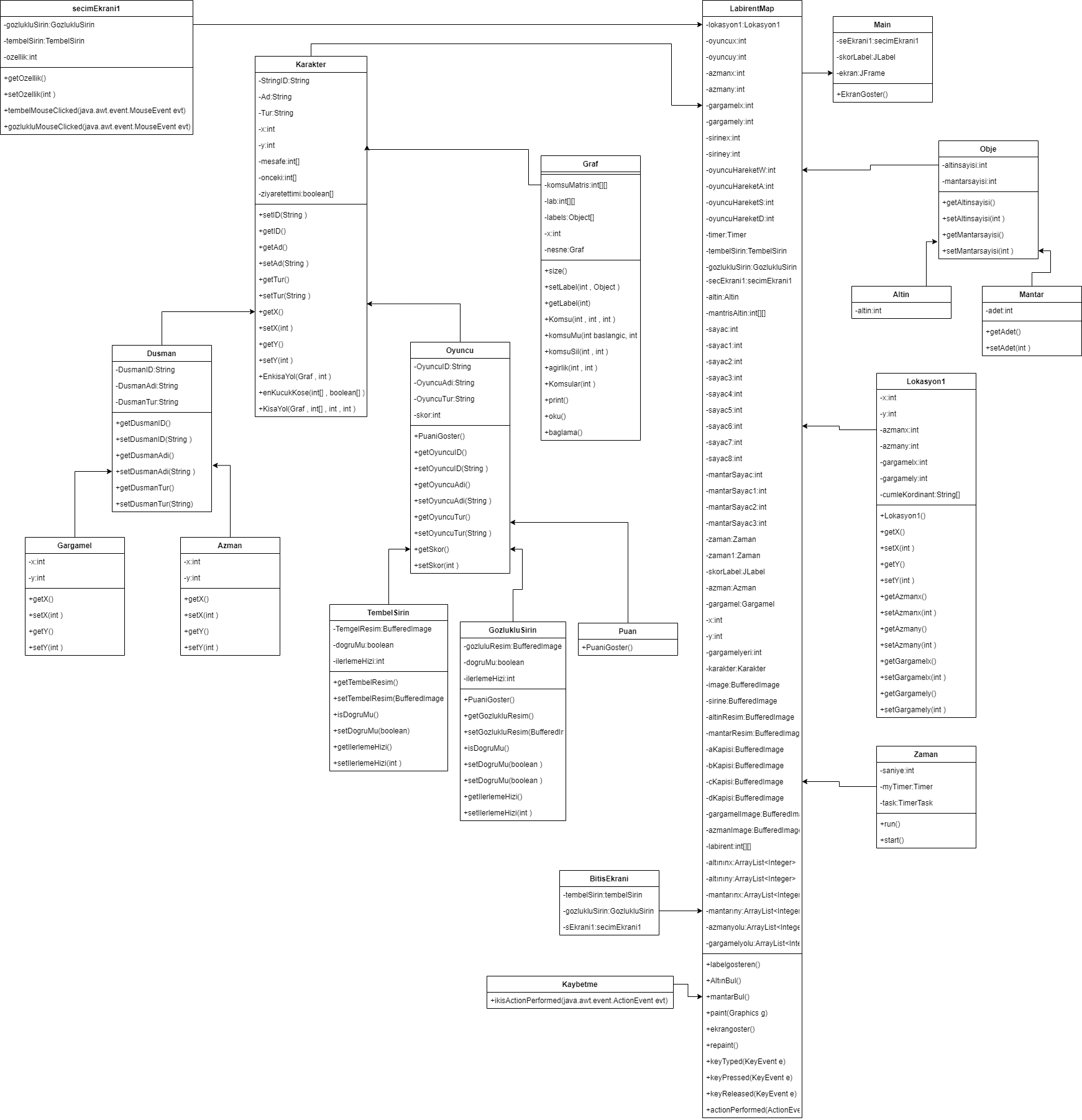
}

***Algoritmanın zaman Karmaşıklığı = O(N^2)***

**5- Algoritmanın akış diyagramı:**

****

**6- Uml Diyagramı:**

****

**7- SONUÇ:**

Bu proje sonucunda nesneye yönelik programlamayı, veri yapılarındaki düğüm mantığını , dijikstra algoritmasını anladık . Takım halinde çalışmayı , azimli olmayı , araştırma yapmayı , sabırlı olmayı öğrendik . Bu proje sayesinde javanın arayüzü olan swinge hakim olduk .

**KAYNAKÇA:**

[1] https://www.youtube.com/watch?v=bImvsFNym9I&t=1s

[2] https://www.youtube.com/watch?v=SZXXnB7vSm4&t=844s

[3] https://www.youtube.com/watch?v=LpJceTbzJFo

[4]https://www.youtube.com/watch?v=QShu1tzoGpU&list=PLzIWkToFwqHRjHdDsrJhQSWGaOJkAHIU4

[5] https://www.kodlamamerkezi.com/java/java-timer-zamanlayici-kullanimi/

[6] https://www.youtube.com/watch?v=S5XryP3IOGA&t=359s

[7] https://qastack.info.tr/programming/299495/how-to-add-an-image-to-a-jpanel

[8] https://mbilgil0.medium.com/abstract-class-vs-interface-bf98133bfadf

[9] https://www.youtube.com/watch?v=Og3ZKXP3wpE