KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA LAB. I- I. Proje

Sude Yağmur Göçgün / 190202046 / sude_572000@hotmail.com

Canberk Tezcan / 190202033 / can-berk2000@hotmail.com

Sporcu Kart Oyunu

1-ÖZET

Projemizin amacı bir oyuncunun (bu kullanıcı olacak) otomatik oyuncuyla (bilgisayara karşı) rekabet edebileceği basit bir kart oyunu tasarlamaktır. Oyunda, toplamda 16 sporcu kartı bulunacaktır. Bu kartlardan 8'i futbolcu, 8'i basketbolcu olmalıdır. Her bir kullanıcıya rastgele 4 basketbolcu, 4 futbolcu kartı dağıtılacaktır. Futbolcuların penaltı, serbest vuruş ve kaleciyle karşı karşıya özellikleri bulunmaktadır. Basketbolcuların üçlük, ikilik ve serbest atış özellikleri bulunmaktadır. Kendilerine dağıtılan 8 karttan kullanıcı kendi seçtiği, bilgisayar her turda random seçilen kartı her turda ortaya atacaktır. Her hamle ardışık sırayla bir futbolcu bir basketbolcu kartlarıyla oynanacak şeklinde olmalıdır.Her iki oyuncunun ortaya koyduğu iki kart her hamlenin sonunda karşılaştırılacaktır. Her iki oyuncunun kartı pozisyon bilgisine göre yüksek değerli karta sahip kullanıcıya 10 puan kazandıracaktır. Değerler aynıysa kartlar geri alınmalıdır. Bu durumda hamle sırası diğer sporcu özelliğinden (futbolcuysa basketbolcuya, basketbolcuysa futbolcuya) devam etmelidir. Eldeki kartlar bitene kadar oyun devam edecektir.

2-Giriş

Proje için Java programlama dili ve Netbeans geliştirme ortamını kullandık. Java programlama dili; açık kodlu, nesneye yönelik, zeminden bağımsız, yüksek verimli, çok işlevli, yüksek seviye, adım adım işletilen bir dildir. Netbeans platformu; Oracle tarafından geliştirilen bir java geliştirme ortamıdır (IDE). Özellikle kullanıcı arayüzü tasarımında sağladığı kolaylıklardan dolayı tercih edilmektedir. Bizim de Netbeans'i tercih etme sebeplerimizin başında GUI tarafında sağladığı kolaylıklardır.

3- TEMEL BİLGİLER

Projemiz Sporcu.java , Futbolcu.java , Basketbolcu.java , Oyuncu.java , Kullanici.java, Bilgisayar.java , GirisResmi.java , Kart.java , OyunResmi.java ,BitisResmi,java, Test.java , Main.java olmak üzere 12 sınıftan oluşmaktadır.

Sporcu. java =

Sporcu classımızda prive tipinde sporculsim , sporcuTakim ve public tipinde acikfoto , kapalifoto değişkenlerimizi tanımladık. Ve

bunları oluşturduğumuz public Sporcu() constuctorına parametre olarak gönderdik. Bunun yanısıra birde parametresiz bir Sporcu constructorı oluşturduk. Private değişkenlerimizi ise özel olarak ulaşmamızı sağlayabilmek için getter ve setter methodları ile tekrar tanımladık. Classımıza bir de sporcuPuaniGoster() isimli bir method açtık fakat override edileceği içi, içini henüz doldurmadık.

Basketbolcu.java =

Bu classımız özelliklerini Sporcu.java classından kalıtım alacaktı. Bu yüzden classımızı

public class Basketbolcu extends Sporcu{}

şeklinde açtık. Sporcu sınıfından kalıtım aldığımız özellikler haricindeki özelliklerimizi ekledik.

private int ikilik

private int ucluk;

private int serbestAtis;

private boolean kullanim;

private boolean kartOrtayaAtildiMi;

kartOrtayaAtildiMi özelliği sayesinde oluşturduğumuz basketbolcu kartının oyun esnasında kullanıp kullanılmadığını kontrol etmeyi amaçladık. Basketbolcu() constructorına bu özelliklerimizi parametre gönderdik, bir de parametresiz constructor açtık.

Sporcu sınıfındaki sporcuPuaniGöster methodunu override ederek, basketbolcu kartımızın burada tanımlı özelliklerini ekrana yazdırmayı amaçladık.

```
@Override
public void sporcuPuaniGoster() {
    super.sporcuPuaniGoster();
    System.out.println("ikilik atış: "+this.ikilik);
    System.out.println("üçlük atış: "+this.ucluk);
    System.out.println("serbest atış: "+this.serbestAtis);
}
```

Bu classa ait private özelliklerimizi de getter ve setter methodarı ile tanımladık.

Futbolcu.java =

Bu classta özelliklerini kalıtımla Sporcu classından almalıydı. Futbolcu classımızı extends ile açtık. Buraya da futbolcuya özel özellikleri ekledik ve bunları parametre alan ve almayan şeklinde 2 constructor yazdık.

```
public class Futbolcu extends Sporcu {

   private int penalti;

   private int serbestAtis;

   private int kaleciKarsiKarsiya;

   private boolean kullanim;

private boolean kartOrtayaAtildiMi;

   public Futbolcu(String sporcuIsim, String
sporcuTakim,ImageIcon acikfoto,ImageIcon
kapalifoto,int penalti, int serbestAtis, int
kaleciKarsiKarsiya ) {

    super(sporcuIsim, sporcuTakim,acikfoto,kapalifoto);

    this.penalti=penalti;

    this.serbestAtis=serbestAtis;

    this.kaleciKarsiKarsiya=kaleciKarsiKarsiya;

   }

   public Futbolcu() {

   }
```

Tekrar sporcu sınıfındaki sporcuPuaniGoster methodunu override ederek Futbolcuda kullanmayı amaçladık.

```
@Override
public void sporcuPuaniGoster() {
    super.sporcuPuaniGoster();
    System.out.println("penaltı vuruş: "+this.penalti);
    System.out.println("serbest atış: "+this.serbestAtis);
    System.out.println("kaleci ile karşı karşıya: "+this.kaleciKarsiKarsiya);
}
```

Private özelliklerimizi de getter ve setter ile ulaşılabilir hale getirdik.

```
public int getPenalti() {   return penalti;  }
```

public void setKullanim(int kullanım){
this.kullanim=true ; }

public boolean kartKullanildiMi(){
return kullanim ; }

public void setPenalti(int penalti) { this.penalti = penalti; }

public int getSerbestAtis() { return serbestAtis; }

public void setSerbestAtis(int serbestAtis) {
this.serbestAtis = serbestAtis; }

public int getKaleciKarsiKarsiya() {
return kaleciKarsiKarsiya; }

public void setKaleciKarsiKarsiya(
int kaleciKarsiKarsiya) { this.kaleciKarsiKarsiya = kaleciKarsiKarsiya; }

Oyuncu.java =

Bu class oyunu oynayacak 2 kişinin (yani kullanıcı ve bilgisayar) özeliklerini tanımlamak amacıyla açıldı. Bu sınıfımızda öncelikle

private int oyuncuID;

private String oyuncuAdi;

private int skor;

olarak özelliklerimi belirledik. Private özelliklere başka classlarda ulaşabilmek için getter ve setter methodları tanımladık.

```
public int getOyuncuID() {
    return oyuncuID;
}

public void setOyuncuID(int oyuncuID) {
    this.oyuncuID = oyuncuID;
}

public String getOyuncuAdi() {
    return oyuncuAdi;
}

public void setOyuncuAdi(String oyuncuAdi) {
    this.oyuncuAdi = oyuncuAdi;
}

public int getSkor() {
    return skor;
}

public void setSkor(int skor) {
    this.skor = skor;
```

Parametreli ve parametresiz olmak üzere iki adet Oyuncu(){} constructorı açtık.

Oyuncunun skorunu gösterebilmek için public void skorGoster(){} şeklinde bir method açtık bu classımızda. Bu sınıfımızdan kalıtım alacak başka bir sınıfımızda override edilerek kullanılacak olan ve bilgisayarın ve kullanıcının desteden random şekilde 8'er tane kart almasını sağlamak amacıyla bir dizi method açtık. Bunlardan biri

```
public ArrayList<Futbolcu> futbolcukart(Futbolcu[] desteFutbolcu, Basketbolcu[] desteBasketbolcu) {
    int a, b = 0;

    ArrayList<Futbolcu> futbolcukart = new ArrayList<Futbolcu>();
    for (int i = 0; i < 8; i += 2) {
        a = (int) (Math.random() * 8);

        if (desteFutbolcu[a].kartKullanildiMi()) {
            i -= 2;

        } else {
            desteFutbolcu[a].setKullanim(1);

            futbolcukart.add(desteFutbolcu[a]);

        }

    }
    return futbolcukart;
```

idi. Burada test sınfında oluşturduğumuz ve içine sporcularımızı attığımız deste[]futbolcu ve deste[] basketbolcu 'yu gönderdik.Bu standart bi işlemdi. For döngümüzün içinde random bir sayı üretilmesini istedik. Eğer futbolcu destesinde bu süretilen random sayıya tekabül eden indeks kullanıldıysa tekrar sayı ürettik. Kullanılmadıysa kullanıldı bilgisini tuttuk ve bu indeksi yeni oluşturduğumuz futbolcukart arraylistine attık.Döngümüz 4 kez döndü.Ve böylece elimizde her seferinde random ve farklı gelecek 4 adet futbolcu kartı bilgisini tuttuk. Test sınıfında bu methodu 2 kez bilgisayar ve kullanıcı için çağırarak 8 futbolcu kartını iki oyuncuyya random bir şekilde dağıtabildik. Sonrasında basketbolcu kartları için de aynı formatta bir method yazarak aynı işlemleri tekrarladık.

```
public ArrayList<Basketbolcu> basketbolcukart(Futbolcu[] desteFutbolcu, Basketbolcu[] desteBasketbolcu)
    int a, b = 0;

    ArrayList<Basketbolcu> basketbolcukart = new ArrayList<Basketbolcu>();
    for (int i = 0; i < 8; i += 2) {
        a = (int) (Math.random() * 8);

        if (desteBasketbolcu[a].kartKullanildiMi()) {
            i -= 2;

        } else {
            desteBasketbolcu[a].setKullanim(1);
            basketbolcukart.add(desteBasketbolcu[a]);

        }

    }
    return basketbolcukart;
```

İsterlerden biri de kullanıcı ve bilgisayarın elindeki tüm kartları bir kartListesi methodyla tutmamız istendiği için kullanıcı ve bilgisayarda override edilerek kullanılacak şu methodu yazdık.

```
public Sporcu[] kartListesi(ArrayList<Futbolcu>futbolculiste,ArrayList<Basketbolcu>basketbolculiste){

  Sporcu[] kullanicilistesi=new Sporcu[8];

  int i,j=0;

  for (i = 0; i <8; i+=2) {

    kullanicilistesi[i]=futbolculiste.get(j);

    kullanicilistesi[i+1]=basketbolculiste.get(j);

    j++;

  }

  for (i = 0;  i< kullanicilistesi.length; i++) {

       kullanicilistesi[i].sporcuPuaniGoster();

  }

  return kullanicilistesi;  }
```

şeklinde bir method yazarak kullanıcıdaki futbol ve basketbol kartlarını sporcu tipinde bir dizide tuttuk. Aynı işlemler bilgisayar için de geçerli.

Burda önemli olan bilgisayarın oyun sırasında ortaya atacağı kartı seçmesi ve bunu doğru sıraya uygun yazmasıydı. Bunun için kartSec ismindeki methodumuzu açtık. Öncelikle içine bir sıra değişkeni gönderdik ki örneğin sıra basketbolcu kartlarında iken futbolcu kartı ortaya atamasın. Sonrasında attığı kartı tekrar atamasın diye methoda bir de içi boş bir arraylist (sayikontrol) gönderdik. Bir de bilgisayarın tüm kartlarını. Sonrasında şöyle bir kontrol yaptık:

İf(sıra futbolcuda mı? Evet ise şunarı yap)

1 ile 4 arasında random bir sayı oluştur.

While(Sayı kontrolde oluşan bu sayı var mı?)

Var ise tekrar bi sayı oluştur

(sayı kontrolde bulunmayana kadar tekrar eder.)

Oluşan random sayı, sayı kontrol dizisinde yoksa sayıkontrol dizisine add yap.

Else(sıra basketbolcuda mı? Evet ise)

5 ile 8 arasında random sayılar uretmeye başla.

Ve devamında aynı işlemleri sürdür.

Üretilen sayıya göre örn 1 ise

Bkart1 i return et

2 ise bkart2 yi return et.

Bu şekilde kart seç her çağırıldığında bilgisayar ortaya bir kart atabildi.

Bilgisayar.java =

Bilgisayar sınıfı tüm özelliklerini Oyuncu.java dan extends etti.

```
public class Bilgisayar extends Oyuncu {

    public Bilgisayar(int oyuncuID, String oyuncuAdi, int skor) {
        super(oyuncuID, oyuncuAdi, skor);

    }
public Bilgisayar(){
}
    @Override
    public Sporcu[] kartListesi(ArrayList<Futbolcu> futbolculiste, ArrayList<Basketbo
        return super.kartListesi(futbolculiste, basketbolculiste);
    }

    @Override
    public Kart kartSec(Kart bkart1, Kart bkart2, Kart bkart3, Kart bkart4, Kart bkar
        return super.kartSec(bkart1, bkart2, bkart3, bkart4, bkart5, bkart6, bkart7,
    }
```

Kullanıcı.java=

Bilgisayar.java gibi tüm özelliklerini Oyuncu.javadan extends etti.

Kart.java =

Başta her kart için bir buton(16 adet) açıp bu butonları futbolcu ve basketbolcu sınıflarına eşitleyebilmek için çok uğraştık. Fakat bu yoldan bir yere varamayacağımızı anladığımızda Jbutton classından özellikleri extends alan Kart isminde bir sınıf açtık. Bu sınıfa

```
String isim;

int penalti;

int serbestVurus;
```

```
int serbestAtis;
int kaleciKarsiKarsiya;
int ikilik;
int ucluk;
boolean kartOrtayaAtildiMi;
ImageIcon acikresim= new ImageIcon();
ImageIcon kapaliresim= new ImageIcon();
int x,y;
```

Sonrasında

```
public Kart (String isim,int penalti,int serbestVurus,int
kaleciKarsiKarsiya,ImageIcon acikresim,ImageIcon
kapaliresim,boolean kartOrtayaAtildiMi,int x,int y){
    this.acikresim=acikresim;
    this.kapaliresim=kapaliresim;
 this .isim=isim;
 this.penalti=penalti;
 this.serbestVurus= serbestVurus;
this.kaleciKarsiKarsiya=kaleciKarsiKarsiya;
 this.kartOrtayaAtildiMi=kartOrtayaAtildiMi;
 this.x=x;
 this.y=y;
```

şeklinde futbolcu için bir constructor yazdık ve basketbolcu için bunu overload ettik.

```
public  Kart(int ikilik, int ucluk,int serbestAtis,String
isim,ImageIcon acikresim,ImageIcon kapaliresim,boolean
kartOrtayaAtildiMi,int x,int y){
    this.acikresim=acikresim;
    this.kapaliresim=kapaliresim;
    this.serbestAtis=serbestAtis;
    this.ikilik=ikilik;
    this. ucluk=ucluk;
    this.kartOrtayaAtildiMi=kartOrtayaAtildiMi;
    this.isim=isim;
    this.x=x;
    this.y=y;
```

Bu class sayesinde tüm kartlarımızın tüm bilgilerine oyun işleyişi sırasında sorunsuz erişebildik.

GirisResmi.java =

JPanel classından extends alan bu sınıfı oyuna giriş esnasındaki ekrana fotoğraf ekleyebilmek için açtık.

```
public class GirisResmi extends JPanel {
    @Override
   protected void paintComponent(Graphics g)
   {
   super.paintComponent(g);
   Image arkaplan=new ImageIcon("C:\\Users\\sude yağmu

   g.drawImage(arkaplan, 0, 0, 1500, 1000, null);

   }
}
```

OyunResmi.java =

Oyunun üzerinde oynanacağı panele fotoğraf ekleyebilmek için açtık.

```
public class OyunResmi extends JPanel {
   @Override
   protected void paintComponent(Graphics g)
   {
   super.paintComponent(g);
   Image arkaplan=new ImageIcon("C:\\Users\\sude ya
   g.drawImage(arkaplan,0,0, null);
   g.drawImage(arkaplan, 0, 0, 1500, 1000, null);


   }
```

BitisResmi.java=

Oyun bitiminde skorları gösteren panele fotoğraf ekleyebilmek için açtık.

```
public class BitisResmi extends JPanel {
   protected void paintComponent(Graphics g)
   {
   super.paintComponent(g);
   Image arkaplan=new ImageIcon("C:\\Users\\sude yağmu
   g.drawImage(arkaplan, 0, 0, 1500, 1000, null);
```

Test.java =

public Test():

Bu class bizim ana clasımız yani tüm oyun

bu classımızda oynanıyor ve tüm işlemler

burada yapılıyor.Öncelikle gerekli kütüphanelerle

birlikte global değişkenlerimizi belirledik.

```
Kart kart1;
Kart kart2;
Kart kart3;
Kart kart4;
Kart kart5;
Kart kart6;
Kart kart7;
Kart kart8;
Kart bkart1;
Kart bkart2;
Kart bkart3;
Kart bkart4;
Kart bkart5;
Kart bkart6;
Kart bkart7;
Kart bkart8;
int kullaniciskor, bilgisayarskor;
Kart ortadakikart1;
Kart ortadakikart2;
ArrayList<Integer> sayideneme = new ArrayList<>();
ArrayList<Integer> ozellikDeneme = new ArrayList<>();
JButton sonrakitur = new JButton("Sonraki Tur");
boolean berabereGeldi = false;
JButton oyunubitir = new JButton();
JLabel kullanicininskoru = new JLabel();
JLabel bilgisayarinskoru = new JLabel();

ArrayList<Futbolcu> kullanicifutbolculiste = new ArrayList<Futbolcu>();
ArrayList<Futbolcu> bilgisayarfutbolculiste = new ArrayList<Futbolcu>();
ArrayList<Basketbolcu> kullanicibasketbolculiste = new ArrayList<Basketbolcu>();
ArrayList<Basketbolcu> bilgisayarbasketbolculiste = new ArrayList<Basketbolcu>();
ArrayList<Kart> kullanicifutbolcukartlari = new ArrayList<Kart>();
ArrayList<Kart> kullanicibasketbolcukartlari = new ArrayList<Kart>();
boolean tumfutbolcukullanildimi, tumbasketbolcukullanildimi = false;
```

kullanıcıfutbolculiste=kullanici.futbolcukart(desteFutbolcu.deste Basketbolcu)

Daha sonra burada futbolcularımızı ve basketbolcularımızı tek tek tanımlayıp bunları oluşturmuş olduğumuz desteFutbolcu ve desteBasketbolcu ArrayListlerimize atadık.Ve bunlarla birlikte her sporcu için bir ImageIcon oluşturarak bunların resimlerini elimizde tutmuş olduk.

```
ImageIcon acikfotof1 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\cengizzzz.p
ImageIcon acikfotof2 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\KARIUS.png"
ImageIcon acikfotof3 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\MARADONA.pn
ImageIcon acikfotof4 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\VOLKAN.png"
ImageIcon acikfotof5 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\MARCO.png")
ImageIcon acikfotof6 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\MESSI.png")
ImageIcon acikfotof7 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\NEYMAR.png"
ImageIcon acikfotof8 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\RONALDO.png
ImageIcon acikfotob1 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\birdddd.png
ImageIcon acikfotob2 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\cediii.png"
ImageIcon acikfotob3 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\ERSAN.png")
ImageIcon acikfotob4 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\HARDEN.png"
ImageIcon acikfotob5 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\JORDAN.png"
ImageIcon acikfotob6 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\KOBE .png")
ImageIcon acikfotob7 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\KYRIE.png")
ImageIcon acikfotob8 = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\LEBRON.png"

ImageIcon kapalifoto = new ImageIcon("C:\\Users\\Canberk\\Documents\\NetBeansProjects\\JavaApplication1\\src\\Image\\kapalifotoo
```

Futbolcu ve basketbolcularımızı oluşturduktan sonra oluşturduğumuz kullanıcı ve bilgisayarın listerinde futbolcukart ve basketbolcukart methotlarını kullanrak bu listelerin içini doldurmuş olduk.

kullanicifutbolculiste = kullanici.futbolcukart(desteFutbolcu, desteBasketbolcu);

kullanicibasketbolculiste = kullanici.basketbolcukart(desteFutbolcu, desteBasketbolcu);

bilgisayarfutbolculiste = bilgisayar.futbolcukart(desteFutbolcu, desteBasketbolcu);

bilgisayarbasketbolculiste = bilgisayar.basketbolcukart(desteFutbolcu, desteBasketbolcu);

Ve oyuncu sınıfındaki kartlistesi methodunu kullanarak kullanıcının ve bilgisayarın kartlarını konsola yazdırdık

kullanici.kartListesi(kullanicifutbolculiste, kullanicibasketbolculiste);

bilgisayar.kartListesi(bilgisayarfutbolculiste, bilgisayarbasketbolculiste);

Proje oyuna giriş ekranının olduğu ve oyunun oynandığı pencereleri burada tanmlayarak sonrasında oluşturduğumuz buton gibi nesneleri bu pencerelere ekledik.

JFrame jf = new JFrame();

JFrame jf1 = new JFrame();

Oyuna basla butonu olusturup bunu penceremize ekledik ve sonrasında tüm sporcularımızı burada constructorların formatına uygun bir şekilde oluşturup bunlara oyunun oynandığı pencerimize ekledik.

kart1 = new Kart(kullanicifutbolculiste.get(0).getSporcuIsim(), kullanicifutbolculiste.get(0).getPenalti(), kullanicifutbolculiste.get(0).getSerbestAtis(), kullanicifutbolculiste.get(0).getKaleciKarsiKarsiya(), kullanicifutbolculiste.get(0).acikresim,kullanicifutbolculiste.get(0).kapaliresim ,false, 50, 750);
kart1.setIcon(kullanicifutbolculiste.get(0).acikresim);

kullanicifutbolcukartlari.add(kart1);

kart1.setBounds(x, y, wt, lt);

jf1.add(kart1);

kart1.setVisible(false);

kart1.addActionListener(this);

kart1.setText("1");

Bunun gibi tek tek tüm sporcularımızı oluşturduktan sonra oyunuBitir,kullanicininskoru,bilgisyarınskoru, ve sonrakitur butonlarımızın setBounds() fonksiyonunu kullanarak konumlarını belirleyip setVisible() fonksiyonuyla da görünürlüklerini kontrol ettik ve butonların kendine has özelliklerini burada tanımladık

GirisResmi ve OyunResmi sınıflarından birer nesne oluşturduk

GirisResmi girisresmi = new GirisResmi();

OyunResmi oyunresmi = new OyunResmi();

GirisResmini jf adlı pencerimize ekledik

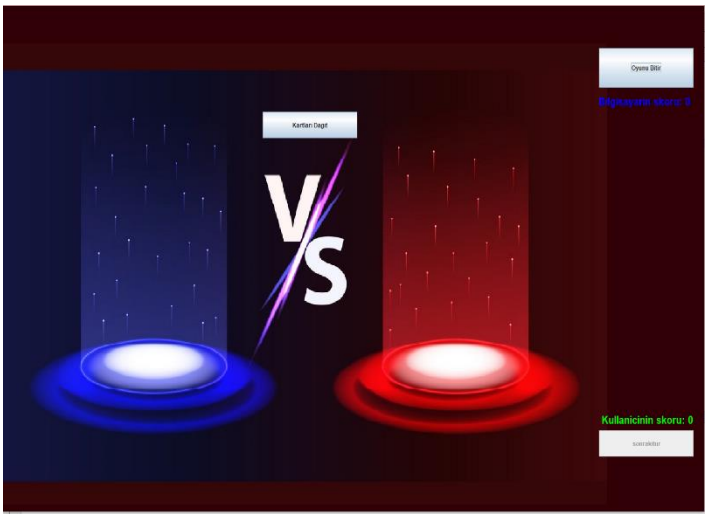
```
jf.add(girisresmi);
        jf.setTitle("SPORCU KART OYUNU");
        jf.setSize(1500, 1000);
        jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jf.setVisible(true);
```

oyunabasla.addActionListener(new ActionListener() :

oyunabasla butonuna tıklanıldığında ise jf1 pencerimizi aktif kıldık ve oyunresmi adında oluşturduğumuz nesnemizi bu pencerimize ekledik.

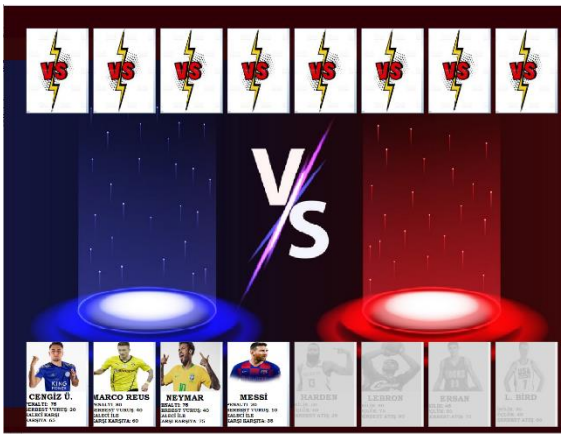
```
  oyunabasla.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
           jf.setVisible(false);

           jf1.add(oyunresmi);
           jf1.setTitle("OYUN BASLADİ");
           jf1.setSize(1500, 1000);
           jf1.setVisible(true);
           jf1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
      });
```

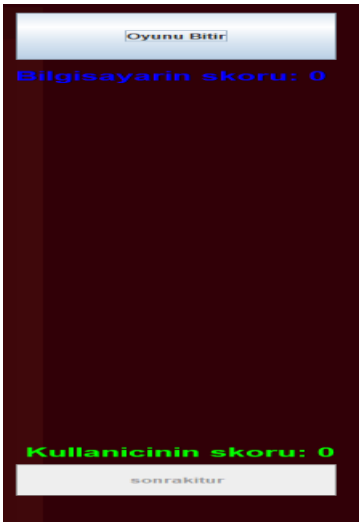


kartlariDagit.addActionListener(new ActionListener():

KartlarıDagıt butonuna tıklandığında ise daha önceden oluşturduğumuz kartları setVisible(true) fonksiyonuyla görünür kıldık.



Test methodumuzu bitirmeden kullaniciskor ve bilgisayarskor değişkenlerimizi sıfıra eşitleyerek Test methodumuzla işlemlerimiz bitirdik.



public int ozelliksec(Kart kart1, Kart bkart, ArrayList<Integer> sayideneme, String sira):

Bu methottaki asıl amacımız sporcuların kendine ait özelliklerini rastgele bir seçim sonrasında birbiriyle kıyaslamaktı. Rastgele özellik kıyaslamak içinse int a değerini 1 ile 3 arasında rastgele değer atayıp buna göre işlemler yaptık ve son kart ortaya atıldığında beraber gelmesi durumunda farklı bir işlem yapılacağı için öncelikle bilgisayarın ürettiği rastgele sayıların tutulduğu sayideneme arraylistlimizin uzunluğunu kontrol edip a değişkenini gibi değişkenleri tanımlayıp gerekli atamaları yaparak işlemlere başladık.

int a = (int) (1 + Math.random() * 3);

int wt = 140, lt = 190, kp = 0, bp = 0, b = 0;

İf(sayidenemenin uzunluğu sekiz ise)s

{ozellikdeneme methoduna rasgele ürelien a değerini ata}

else if(ozellikdenemenin içinde a sayisi varsa)

{ 1 ile 3 arasında yeni bir a değeri üret}

Bu işlemlerle öncelikle son kart olup olmamasını kontrol ettik.Son kart ise kıyaslanan özellikteki a sayisini ozellikDeneme arraylistinde tutarak daha öncesinde içinde bulunabilmesi durumunda farklı bir özellik kıyaslaması yapabildik.

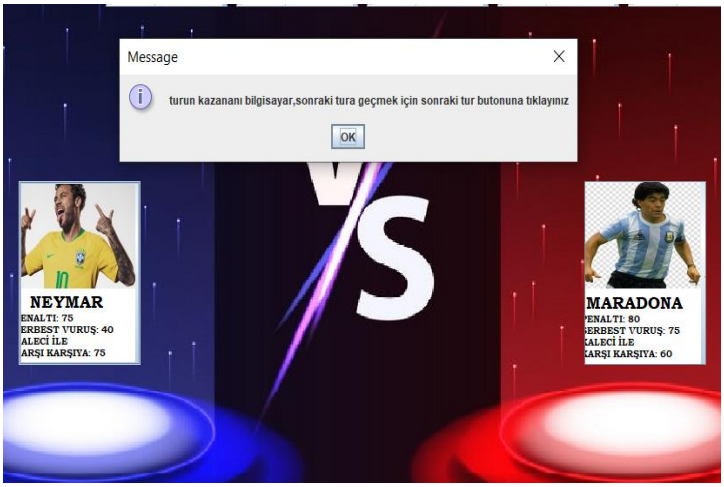
Son kart olmaması durumunda ise a değişkeninin alacağı değerlere göre bir switch case yapısı kurarak oluşan durumları yazıp oluşması halinde nelerin olacağını belirledik.

Rastgele atanan a değeri sonucunda kıyaslanan özelliğe göre kullanıcının puanı yüksekse kullanıcının ve bilgisayarın kartlarının kartOrtayaAtildiMi boolean değerini true ,berabereGeldi boolean değerini false yaptık ve kp değişkenini bir arttırdık.Bilgisayarınki yüksekse de yine bilgisayarın ve kullanıcının kartlarının kartOrtayaAtildiMi boolean değerini true yapıp berabereGeldi boolean değerini false yaptık ve bp değişkenini bir arttırdık. Berabe gelmesi durumundaysa bir kullanıcının ve bilgisayarın kartlarını eski konumlarına göndererek kartOrtayaAtildiMi boolean değerini false yapıp berabereGeldi boolean değerini true yaptık ve aynı zamanda kullanıcın futbolcu ve basketbolcu kartlarına erişmlerini setEnabled() fonksiyonuyla erişime kapayıp sonraki tur butonunu kullana bilinir kıldık.

```
switch (a) {
    case 1:
        if (kart1.penalti > bkart.penalti) {
            berabereGeldi = false;
            kp++;
            JOptionPane.showMessageDialog(null, "Turun kazananı kullanici,sonraki tura geçmek için sonraki
            kart1.kartOrtayaAtildiMi = true;
            bkart1.kartOrtayaAtildiMi = true;
        } else if (kart1.penalti < bkart.penalti) {
            berabereGeldi = false;
            bp++;
            JOptionPane.showMessageDialog(null, "Turun kazananı bilgisayar,sonraki tura geçmek için sonra
            kart1.kartOrtayaAtildiMi = true;
            bkart1.kartOrtayaAtildiMi = true;
        } else {
            JOptionPane.showMessageDialog(null, "Berabere,sonraki tura geçmek için sonraki tur butonuna t
            berabereGeldi = true;
            kart1.setBounds(kart1.x, kart1.y, wt, lt);
            bkart.setBounds(bkart.x, bkart.y, wt, lt);
            bkart.setIcon(bkart.kapaliresim);
            enableDisableFutbolcu(false);
            enableDisableBasketbolcu(false);
            sonrakitur.setEnabled(true);

            kart1.kartOrtayaAtildiMi = false;
            bkart1.kartOrtayaAtildiMi = false;

        }
        break;
```



Methodumuzun int reurn değeri olduğu için işlemlerimizin sonunda kp ve bp değişkenlerini kıyaslarak bir return işlemi uyguladık.

İf(kp değeri bp değerinden büyükse)

{ b değerini 1 yap}

else if (bp değeri kp değerinden büyükse)

{ b değerini 2 yap}

Else hiçbir şey yapma be değeri 0 değerine eşit olsun

b değişkenini return et.

public void sifirla(Kart kart1, Kart kart2):

Kart tipinde gelen kart1 in görünürlüğünü kapa;

Kart tipinde gelen kart1 in görünürlüğünü kapa;

kart1 in kartOrtayaAtildiMi bilgisini true yap;

kart2 nin kartOrtayaAtildiMi bilgisini true yap;

public void actionPerformed(ActionEvent e):

Oyuncu oyuncuu = new Oyuncu(1, "can", 0);

Bilgisayar bilgisayar = new Bilgisayar(2, "sude", 0);

Kullanacagımız methotlar bir oyuncunun ve kullanıcının methotları olduğundan burda işlemleri yapabilmek adına birer nesne oluşturduk.

Burdaysa tüm tıklama işlemlerini bir switch case yapısı kurarak kontrol ettik. Daha önceden kart tipindeki butonlarımızı setText() fonksiyonunu kullanarak üsütne yazı yazdırıp burda da tıklanılan şeyin o olması durumunda yapılması gerekenleri yazdık.

```
switch (e.getActionCommand()) {
    case "1":
        Kart kartdeneme = null;

        kart1.setBounds(250, 420, 140, 190);
        kartdeneme = oyuncuu.kartSec(bkart1, bkart2, bkart3, bkart4, bkart5, bkart6, bkart7, bkart8, "fu
        kartdeneme.setIcon(kartdeneme.acikresim);
        kartdeneme.setBounds(900, 420, 140, 190);
        kart1.kartOrtayaAtildiMi = true;
        kartdeneme.kartOrtayaAtildiMi = true;
        ortadakikart1 = kart1;
        ortadakikart2 = kartdeneme;
        x = ozelliksec(kart1, kartdeneme, sayideneme, "futbolcu");
        if (x == 1) {

            kullaniciskor += 10;
            enableDisableFutbolcu(false);
            enableDisableBasketbolcu(false);
            ortadakikart1.setEnabled(true);
            sonrakitur.setEnabled(true);
            kullanicininskoru.setText(" Kullanicinin skoru: " + kullaniciskor);

        } else if (x == 2) {
            bilgisayarskor += 10;
            enableDisableFutbolcu(false);
            enableDisableBasketbolcu(false);
            ortadakikart1.setEnabled(true);
            sonrakitur.setEnabled(true);
            bilgisayarinskoru.setText("Bilgisayarin skoru: " + bilgisayarskor);
        } else {
            if (sayideneme.size() == 8) {
                if (ozellikDeneme.size() != 3) {
                    x = ozelliksec(kart1, kartdeneme, sayideneme, "futbolcu");
                } else {

                    sonrakitur.setEnabled(false);
                    JOptionPane.showMessageDialog(null, "Oyunu bitirmek için lütfen oyunu bitir butonuna
                }
            } else {
                enableDisableFutbolcu(false);
                enableDisableBasketbolcu(false);
                sonrakitur.setEnabled(true);
            }

        }
        break;
```

Örneğin burda case 1 olması kullanıcının kart1'e tıklaması anlamına gelmektedir yani kullanıcı kartını seçmiştir.Biz de öncelikle bu kartı setBounds() fonksiyonuyla istediğimiz konuma yerleştirdik.

Kart kartdeneme = oyuncuu.kartSec(bkart1, bkart2, bkart3, bkart4, bkart5, bkart6, bkart7, bkart8, "futbolcu", sayideneme);

Diyerek oyuncu sınıfının kartSec methoduyla bilgisayara daha önce kullanılmamış bir kart seçtik ve onun da konumunu ayarladık.

ortadakikart1 = kart1;

ortadakikart2 = kartdeneme;

kullanıcının ve bilgisayarın kartlarını ortadaki kart1 ve ortadaki kart2 değişkenlerine sonrasında başka bir methotda kullanmak üzere atama yaptık.

Kartları bildiğimize göre sıra geldi bu kartların rastegele bir özelliğini kıyaslamaya.Bunu da yazmış olduğumuz ozellik sec methodunu kullanarak yaptık.

Öncesinde oluşturduğumuz int x değerine

x = ozelliksec(kart1, kartdeneme, sayideneme, "futbolcu");

Ataması yaptık ve x in alacağı değere göre de bir kıyaslama yaptık.

İf(x değeri 1 değerine eşitse)

{ kullanıcının skorunu 10 arttır.

enableDisableFutbolcu(false) ; yaparak futbolcu kartlarına erişimi kapa. enableDisableBasketbolcu(false); yaparak basketbolcu kartlarına erişimi kapa.

ortadakikart1.setEnabled(true); yaparak ortadaki kullanıcı kartının erişebelirliğini aç.

sonrakitur.setEnabled(true); yaparak sonrakitur butonuna erişimi aç.

Kullanicininkoru labeline (Kullanicinin skoru: kullaniciskor değerini ata.}

Else if (x değeri 2 değerine eşitse)

{ bilgisayarın skorunu 10 arttır.

enableDisableFutbolcu(boolean deger) methoduna false değerini gönder.

enableDisableBasketbolcu(boolean deger) methoduna false değeri gönder. ortadakikart1.setEnabled(true) yaparak ortadaki kullanıcı kartının erişebelirliğini aç

sonrakitur.setEnabled(true); yaparak sonrakitur butonuna erişimi aç.

bilgisayarinskoru labeline (Bilgisayarın skoru: bilgisayarskor değerini ata.}

else

{ if (sayideneme arraylistinin boyutu 8 mi)

{ (ozellikDeneme arraylistinin boyutu 3 mü)

{ x değerine ozelliksec methodunu kullanarak yeni bir değer ata.}

else{

sonrakitur butonunun erişimini kapa.

Ekrana “Oyunu bitirmek için lütfen oyunu bitir butonuna tiklayiniz.” Mesajnı ver.

else{

enableDisableFutbolcu(boolean deger) methoduna false değeri gönder.

enableDisableBasketbolcu(boolean deger) methoduna false değerini gönder.

Çık.

Kaba kod yardımıyla göstermiş olduğumuz ilk casedeki işlemleri diğer caselere uygun bir biçimde yaparak

Herhangi bir karta tıklanıldığı taktirde bilgisayara rastgele bir kart seçip bu kartların sporcu tipine göre herhangi bir özeliğini kıyaslayarak gücü yüksek olan oyuncunun skorunu arttırdık.

case "Oyunu Bitir":

BitisResmi bitisresmi = new BitisResmi();

JFrame jf2 = new JFrame();

BtisResmi sınıfından bitisresmi nesnesi oluşturduk ve jf2 adına yeni bir JFrame oluşturduk.

kullanicininskoru.setVisible(true);

bilgisayarinskoru.setVisible(true);

Kullanininskoru ve bilgisayarinskoru butonlarının görünürlüğünü açtık.

Jf2 JFraemimizin konumunu belirleyip görünürlüğünü açtık ve oluşturduğumuz objeleri jframememizn üstüne ekledik ve jframe’in kapatılması halinde programı durdurmasını sağladık.

Daha sonrasında kullanicinin ve bilgisayarın skorunu kıyaslayarak kullanıcının skorunun yüksek olması, bilgisayarın skorunun yüksek olması ve iki oyuncunun da skorunun eşit olması halinde kullanicininskoru ve bilgisayarınskoru butonlarının konumlarını ,içindeki yazıları ve ekrana verilmesi gereken mesajları yazdık.

case "Sonraki Tur":

Burada da işe ilk başta sayideneme arraylistimizin son indeksinin değerinin 4den büyük olup olmamasını kontrol ederek başladık.4 den büyükse yani kullanıcı en son basketbolcu kartı oynamışsa bir for döngüsü açarak tüm futbolcu kartları kullandı mı bilgisine ulaşmayı hedefledik eğer tüm futbolcu kartları kullanıldıysa basketbolcu kartlarını erişime açıp kullanılmadıysa da yalnızca futbolcu kartlarını açıp basketbolcu kartlarının erişimini kapadık.Aynı işlemlerin tersini de en son basketbolcu kullanılmışsa diyerek yaptık.Bu koşullarda berabere gelmediği için öncesinde atamış olduğumuz ortadaki kart1 ve ortadakikart2 değişkenlerini sifirla methoduna gönderdik.Berabere gelmesi durumundaysa sayideneme arraylisitmizin en son indexteki değerini sildik böylece bilgisayara berabere gelen kartı tekrar seçebilme özelliğini vermiş olduk.

public void enableDisableFutbolcu(boolean choice) :

Bu methotta amacımız işlem kolaylığıydı methodumuza gelen boolean değere göre tüm futbolcuların erişimini açtık yada kapattık.

```
public void enableDisableFutbolcu(boolean choice) {
    kart1.setEnabled(choice);
    kart2.setEnabled(choice);
    kart3.setEnabled(choice);
    kart4.setEnabled(choice);
}
```

public void enableDisableBasketbolcu(boolean choice):

Bir önceki methotta olduğu gibi burda da işlem kolaylığı olması amacıyla gelen boolean değerine göre basketbolcu kartlarının erişimini açıp kapamayı hedefledik.

```
public void enableDisableBasketbolcu(boolean choice) {
    kart5.setEnabled(choice);
    kart6.setEnabled(choice);
    kart7.setEnabled(choice);
    kart8.setEnabled(choice);
}
```

Main.java=

Public static void main methodunun içinde test.java classını çağırarak kodu çalıştırdık.

4-DENEYSEL SONUÇ

Projeyi gerçekleştirmeden önce,daha önce hiç kullanmadığımız java swing arayüzü hakkında pek çok araştırma yaparak bilgiler edindik. Java dilinde henüz yeni başlamamızdan ötürü kendimizi çok fazla geliştirme fırsatı bulduk. Oyun içindeki algoritmaları kurabilmek için çok farklı fikirler ve yöntemler denememiz çözüm üretme kabiliyetimizi arttırdı.

5-SONUÇ

Bize verilen projeyi isterleri eksiksiz kullanarak gerçekleştirebildik. Hiç bilmediğimiz arayüz hakkında çok fazla sorunla karşılaştıkça onları çözmek için çok fazla araştırma yaptık ve sporcu kart oyununu başarıyla tamamladığımızı düşünüyoruz.

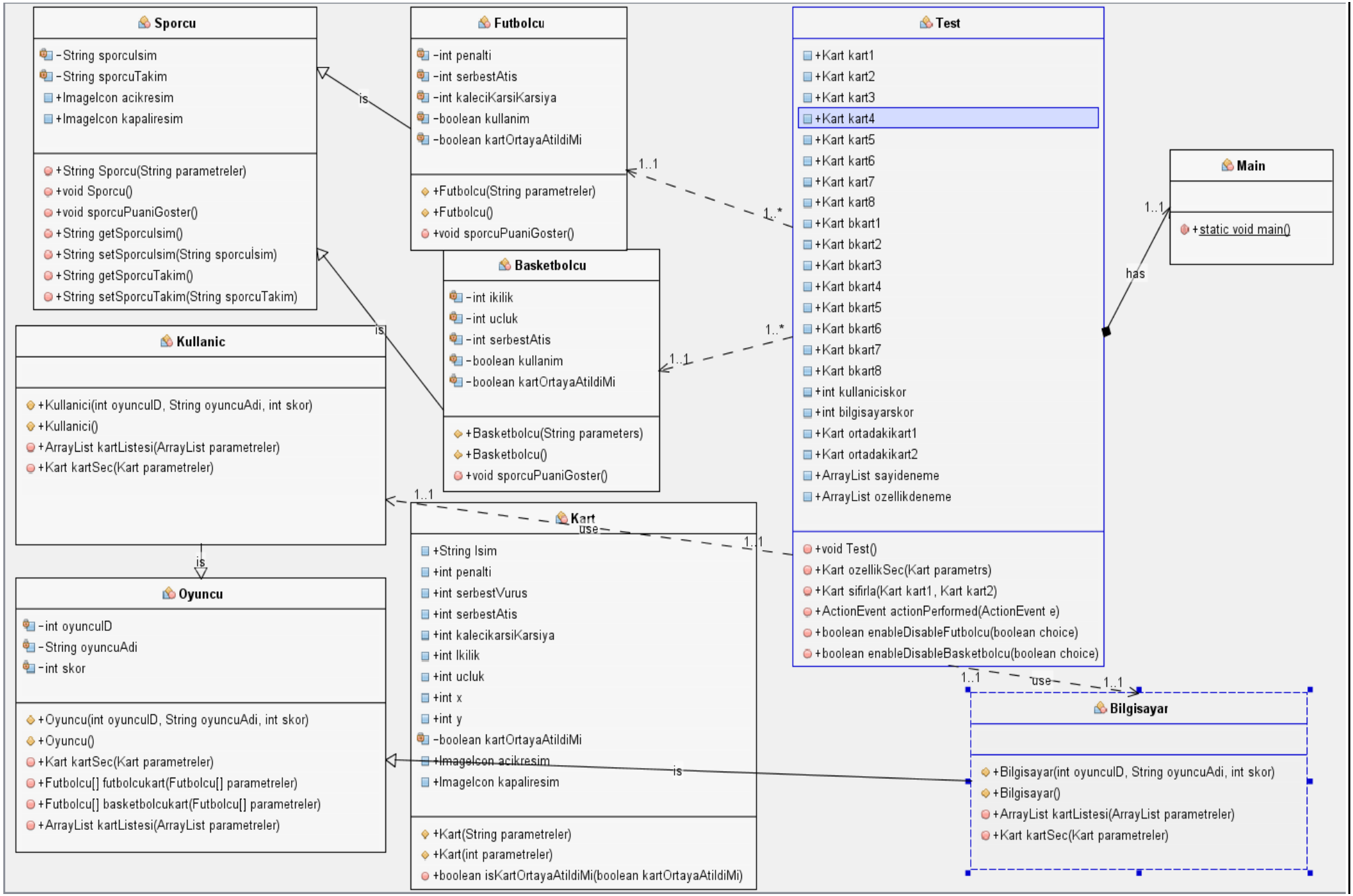
6-KAYNAKÇA

https://www.geeksforgeeks.org/creating-frames-using-swings-java/

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

https://www.youtube.com/watch?v=d_DOlqax0qk

https://www.geeksforgeeks.org/java/


Sporcu
-String sporculsim
-String sporcuTakim
+ImageIcon acikresim
+ImageIcon kapaliresim
+String Sporcu(String parametreler)
+void Sporcu()
+void sporcuPuaniGoster()
+String getSporculsim()
+String setSporculsim(String sporculsim)
+String getSporcuTakim()
+String setSporcuTakim(String sporcuTakim)
Futbolcu
-int penalti
-int serbestAtis
-int kaleciKarsiKarsiya
-boolean kullanim
-boolean kartOrtayaAtildiMi
+Futbolcu(String parametreler)
+Futbolcu()
+void sporcuPuaniGoster()
Basketbolcu
-int ikilik
-int ucluk
-int serbestAtis
-boolean kullanim
-boolean kartOrtayaAtildiMi
+Basketbolcu(String parameters)
+Basketbolcu()
+void sporcuPuaniGoster()
Test
+Kart kart1
+Kart kart2
+Kart kart3
+Kart kart4
+Kart kart5
+Kart kart6
+Kart kart7
+Kart kart8
+Kart bkart1
+Kart bkart2
+Kart bkart3
+Kart bkart4
+Kart bkart5
+Kart bkart6
+Kart bkart7
+Kart bkart8
+int kullaniciskor
+int bilgisayarskor
+Kart ortadakikart1
+Kart ortadakikart2
+ArrayList sayideneme
+ArrayList ozellikdeneme
+void Test()
+Kart ozellikSec(Kart parametrs)
+Kart sifirla(Kart kart1, Kart kart2)
+ActionEvent actionPerformed(ActionEvent e)
+boolean enableDisableFutbolcu(boolean choice)
+boolean enableDisableBasketbolcu(boolean choice)
Main
+static void main()
Kullanic
+Kullanici(int oyuncuID, String oyuncuAdi, int skor)
+Kullanici()
+ArrayList kartListesi(ArrayList parametreler)
+Kart kartSec(Kart parametreler)
Oyuncu
-int oyuncuID
-String oyuncuAdi
-int skor
+Oyuncu(int oyuncuID, String oyuncuAdi, int skor)
+Oyuncu()
+Kart kartSec(Kart parametreler)
+Futbolcu[] futbolcukart(Futbolcu[] parametreler)
+Futbolcu[] basketbolcukart(Futbolcu[] parametreler)
+ArrayList kartListesi(ArrayList parametreler)
Kart
+String Isim
+int penalti
+int serbestVurus
+int serbestAtis
+int kalecikarsiKarsiya
+int Ikilik
+int ucluk
+int x
+int y
-boolean kartOrtayaAtildiMi
+ImageIcon acikresim
+ImageIcon kapaliresim
+Kart(String parametreler)
+Kart(int parametreler)
+boolean isKartOrtayaAtildiMi(boolean kartOrtayaAtildiMi)
Bilgisayar
+Bilgisayar(int oyuncuID, String oyuncuAdi, int skor)
+Bilgisayar()
+ArrayList kartListesi(ArrayList parametreler)
+Kart kartSec(Kart parametreler)
is
is
is
is
1..1
1..*
1..*
1..1
1..1
1..1
has
1..1
use
use
1..1
1..1