# KOCAELİ ÜNİVERSİTESİ
# BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
# PROGRAMLAMA LABORATUVARI

Tuğba ÇEVİKER
210202056

Samethan KAZANBAŞ
210202043

https://www.overleaf.com/project/638cb51ae7c1d9c79729557b

## I. ÖZET

Bu döküman Programlama Laboratuvarı I dersinin 2. Projesi olan Taş Kağıt Makas oyununun çözümünü açıklamaya yönelik oluşturmuluştur. Dökümanda Giriş, Yöntem, Tablo , Katkılar, Sonuç ve Kaynakça kısımlarına yer verilmiştir.

## II. GİRİŞ

Projede bizden istenen C++ veya Java dilini kullanarak bir oyuncunun diğer bir oyuncuyla rekabet edebileceği taş kağıt makas oyununu tasarlamaktır.

Oyun kullanıcı-bilgisayar ya da bilgisayar-bilgisayar olmak üzere iki şekilde oynanabilmelidir.

Oyunda her oyuncunun elinde baslangıçta tas, kagıt ve makas olmak uzere 3 farklı tipte olan nesnelerden 5 adet bulunacaktır.

Kullanıcı-bilgisayar oyununda kullanıcı kendi nesnelerini oyunun basında kendisi seçecek, bilgisayar ise bunu rastgele yapacaktır.

Bilgisayar-bilgisayar oyununda bilgisayarın seçimleri otomatik olarak yapılmalı ve oyun adımları hızlı bir şekilde ilerlemelidir.

Kullanıcı bilgisayar oyunu sırasında kullanıcı bilgisayarın elindeki nesnelerini görmemelidir. Ancak her seçim yapıldığında seçilen nesneler arayüzde kullanıcı tarafından görülebilmelidir. Kullanıcı elindeki kalan nesnelerden herhangi birini her hamle basında kendi seçebilmelidir.

Kullanıcı-bilgisayar oyununda oyun;Önce kullanıcı elindeki nesneler arasından secim yapar daha sonra bilgisayar rastgele secim yapar en son nesneler karşılaştırılarak skorlar hesaplanır ve yeni seçimle oyun devam eder. Bilgisayar-bilgisayar oyununda oyun; her iki oyuncu için de nesne seçimi rastgele seçimlerle ilerler.

Oyun iki durumda bitebilir.Bunlar ;İki taraftan birinin elindeki nesneler tükendiginde nesnesi tükenen oyunu kaybeder ya da başlangıçta belirlenen hamle sayısı tamamlandığında oyuncuların elinde kalan nesnelerin dayanıklılıkları toplamı karşılaştırılır ve değeri fazla olan oyunu kazanır ve oyun sona erer.

Bütün nesneler seçilmeden aynı nesne seçimi tekrarlanmamalıdır. 5 nesnenin tamamı seçildiginde aynı nesne tekrar seçilebilmelidir.

örneğin;
nesneler listesi =(tas,kagit,makas,tas1,makas1)
hamleler listesi =(tas,makas,tas1,makas1,kagit,tas,...)(doğru seçim)
hamleler listesi =(tas,tas,makas,kagit,tas1,kagit,...)(yanlış seçim)

Kullanıcı ve bilgisayarın secmis oldugu nesnelerin karsılastırılması ve skorların hesaplanması belirlenen(şekil 1) formullere gore yapılacaktır. Buna gore herhangi bir hamlede karsı karsıya gelen nesnelerin birbirine karsı etkileri ayrı ayrı hesaplanmalıdır. Sonrasında bir nesnenin etki değeri kadar rakip nesnenin dayanıklılık degeri azaltılacaktır. Dayanıklılık degeri 0 veya 0'ın altına dusen nesneler oyundan silinecektir. Nesneyi yenen rakip nesnenin seviye puanını 20 artırılacaktır.Eğer seviye puanı 30 veya 30 un üstüne çıkarsa o nesne terfi edecektir (tas-ağır tas,makas-usta makas,kagıt-ozel kagit)

$$makas\_etkisi = \frac{keskinlik\{*direnc\}}{(a * nufuz_r\{*kalinlik_r\} + (1-a) * katilik_r\{*sicaklik_r\}}$$
$$kagit\_etkisi = \frac{nufuz\{*kalinlik\}}{(a * katilik_r\{*sicaklik_r\} + (1-a) * keskinlik_r\{*direnc_r\})}$$
$$tas\_etkisi = \frac{katilik\{*sicaklik\}}{(a * keskinlik_r\{*direnc_r\} + (1-a) * nufuz_r\{*kalinlik_r\}}$$

Fig. 1. Belirtilen etki formülleri

## III. YÖNTEM

Oyun classımızın içinde farklı methodlarda kullanabilmek için Scanner scan , Random r, boolean sonuc=false,File file ve oyunarayuz arayuz değişkenlerini static olarak tanımladık. Oyun classımızın main methodunda arayüz classımızı çağırdık.Eclipsede WindowBuilder yardımıyla tasarladığımız oyunarayuz classımızın yapıcı methoduna kullanıcı-bilgisayar ve bilgisayar-bilgisayar olmak üzere iki buton ekledik.Kullanıcı-bilgisayar butonunun actionPerformed methodunda oyun2 adlı yeni bir classla frame açtık.Oyun2nin yapıcı methodunda zamanımızın yetmediğini belirten bir label , oyunu terminalden oynatabilmek için başlat butonu ekledik.Başlat butonunun actionPerformed methodunda oyun2 classında oluşturduğumuz "calistir" methodu başlatılır . "calistir" methodunun içinde oyun.kpc() methodu çağırılmaktadır.Bilgisayar-bilgisayar butonunun actionPerformed methodunda oyun1 adlı yeni bir classla frame açtık.Oyun1in yapıcı methoduna başlat butonu ekledik.Başlat butonunun actionPerformed methodunda oyun1 classında oluşturduğumuz "calistir" methodu başlatılır . "calistir" methodunun içinde oyun.pcpc() methodu çağırılmaktadır.

kpc():Kullanıcı-Bilgisayar oyunu

İlk önce kullanıcı isimi ve idsi istenir boş değişkenlere atanır.ArrayList¡nesne¿ k1N ve ArrayList¡String¿ k1NA olmak üzere iki boş ArrayList oluşturulur.Kurulan for döngüsü sayesinde ((0)-TAS (1)-KAĞIT (2)-MAKAS) kullanıcının seçimlerine göre nesneler oluşturulur , k1N ve k1NA ArrayListleri "add" methodu yardımıyla doldurulur.

```
int no=scan.nextInt();
if(no==0) {

    nesne tas=new tas();
    k1N.add(tas);
    String adi="tas";
    k1NA.add(adi);
```

Fig. 2. Örnek kod parçacığı

Alınan bilgilerle "oyuncu k1=new kullanici(ad,id,k1N,k1NA)" ilk oyuncu oluşturulur.

İlk önce bilgisayar isimi ve idsi istenir boş değişkenlere atanır.ArrayList¡nesne¿ b1N ve ArrayList¡String¿ b1NA olmak üzere iki boş ArrayList oluşturulur.Kurulan for döngüsü sayesinde random methoduyla ((0)-TAS (1)-KAĞIT (2)-MAKAS) üçe kadar değerler atanır.Atanan değerlere göre nesneler oluşturulur , b1N ve b1NA ArrayListleri "add" methodu yardımıyla doldurulur. Alınan bilgilerle "oyuncu b1=new bilgisayar(ad,id,b1N,b1NA)" ikinci oyuncu oluşturulur. Terminalden hamle sayısı istenir.Hamle sayısı kadar dönen for döngüsü kurulur.Her bir hamlede k1 ve b1 nesnelerinin "nesneSec" methodu çağırılır ve return edilen değere Karşılık gelen nesne " getNesneler().get(return edilen değer)" methoduyla ekrana yazdırılır ve her iki nesne için de "getNesneAdlari().get(return edilen değer).nesnePuaniGoster" Methoduyla değerler ekrana yazdırılır. İki oyuncu için de " getNesneAdlari().get(return edilen değer) " methodu çağırılır ve nesneler karşılaştırılır.Eğer nesneler aynıysa veya aynı tür ise nesneler birbirine zarar veremez.Farklı türde iseler " double etki1=k1.getNesneler.get(return edilen).etkihesapla(b1.getNesneAdlari(),return edilen) " sayesinde kullanıcı nesnesinin bilgisayar nesnesine verdiği etki etki1 değişkenine atanır. " double etki2=b1.getNesneler.get(return edilen).etkihesapla(k1.getNesneAdlari(),return edilen) " sayesinde bilgisayar nesnesinin kullanıcı nesnesine verdiği etki etki2 değişkenine atanır. Bu iki değişken iki oyuncu için de durumGuncelle methoduna gönderilir.Eğer oyunculardan birinin nesnesi öldüyse nesne o oyuncunun tüm ArrayListlerinden "remove" methoduyla silinir.Ardından diğer nesnenin parametresiz durumGuncelle methodu çağırılır ve seviye puanı arttırılır.Eğer oyunculardan birinin nesnesinin seviye puani otuz ve üzeri olursa o nesne terfi eder.Tüm ArrayListlerde "set" methoduyla yerine terfi ettiği nesne yazılır.(tas-agirtas , makas- ustamakas , kagit-ozelkagit) Eğer oyunculardan birinin nesneleri tükenirse diğer oyuncu kazanır."sonuc=true" ataması yapılır ve hamle kadar dönen for döngüsünden "break" komutuyla çıkılır. Eğer oyunculardan birinin nesneleri tükenmeden hamle sayısı biterse "sonuc=false" olduğu için "if(!sonuc)" yapısının içinde oyunun hamle sayısının bittiği belirtilir.Her iki oyuncu için de ellerindeki nesne sayısı kadar dönen for döngüsü kurulur ve "getNesneler.get(i).getDayaniklilik()" methoduyla her bir nesnenin dayanikliliklari toplanır.İki oyuncu için de "setSkor" methoduyla elde edilen değerler oyuncuların skorlarına atanır.Skorlar karşılaştırılır eşit ,büyük ve küçük olma durumlarına göre sonuç belli olur ve ekrana yazdırılır. Tüm bu adımlar "oyun.txt" dosyasında ve terminalde hamle hamle belirtilir.

pcpc():Bilgisayar-Bilgisayar oyunu

Her hamlede gerçekleşen olayları ekrana yazdırabilmek için boş bir "String text " oluşturulur.Her hamle başında bu String boşaltılır. ArrayList¡nesne¿ b1N ve ArrayList¡String¿ b1NA olmak üzere iki boş ArrayList oluşturulur.Kurulan for döngüsü sayesinde ((0)-TAS (1)-KAĞIT (2)-MAKAS) random methoduyla üçe kadar değerler atanır ve nesneler oluşturulur , b1N ve b1NA ArrayListleri "add" methodu yardımıyla doldurulur. Alınan bilgilerle "oyuncu b1=new bilgisayar(ad,id,b1N,b1NA)" ilk oyuncu oluşturulur.

ArrayList¡nesne¿ b2N ve ArrayList¡String¿ b2NA olmak üzere iki boş ArrayList oluşturulur.Kurulan for döngüsü sayesinde random methoduyla ((0)-TAS (1)-KAĞIT (2)-MAKAS) üçe kadar değerler atanır.Atanan değerlere göre nesneler oluşturulur , b2N ve b2NA ArrayListleri "add" methodu yardımıyla doldurulur. Alınan bilgilerle "oyuncu b2=new bilgisayar(ad,id,b2N,b2NA)" ikinci oyuncu oluşturulur. "JOptionPane.showInputDialog " methoduyla hamle sayısı isternir.Hamle sayısı kadar dönen for döngüsü kurulur.Her bir hamlede b1 ve b2 nesnelerinin "nesneSec" methodu çağırılır ve return edilen değere Karşılık gelen nesne " getNesneAdlari().get(return edilen değer)" methoduyla ve her iki nesne için de "getNesneler().get(return edilen değer).nesnePuaniGoster" Methoduyla değerler texte atanır . Bundan sonra her hamlede gerçekleşen olaylar text Stringine atanır. İki oyuncu için de " getNesneAdlari().get(return edilen değer) " methodu çağırılır ve nesneler karşılaştırılır.Eğer nesneler aynıysa veya aynı tür ise nesneler birbirine zarar veremez.Farklı türde iseler " double etki1=b1.getNesneler.get(return edilen).etkihesapla(b2.getNesneAdlari(),return edilen) " sayesinde bilgisayar1 nesnesinin bilgisayar2 nesnesine verdiği etki etki1 değişkenine atanır. " double etki2=b2.getNesneler.get(return edilen).etkihesapla(b1.getNesneAdlari(),return edilen) " sayesinde bilgisayar2 nesnesinin bilgsayar1 nesnesine verdiği etki etki2 değişkenine atanır. Bu iki değişken iki oyuncu için de durumGuncelle methoduna gönderilir.Eğer oyunculardan birinin nesnesi öldüyse nesne o oyuncunun tüm ArrayListlerinden "remove" methoduyla silinir.Ardından diğer nesnenin parametresiz durumGuncelle methodu çağırılır ve seviye puanı arttırılır.Eğer oyunculardan birinin nesnesinin seviye puani otuz ve üzeri olursa o nesne terfi

eder.Tüm ArrayListlerde “set” methoduyla yerine terfi ettiği nesne yazılır.(tas-agirtas , makas- ustamakas , kagit-ozelkagit) Eğer oyunculardan birinin nesneleri tükenirse diğer oyuncu kazanır.”sonuc=true” ataması yapılır ve hamle kadar dönen for döngüsünden “break” komutuyla çıkılır. Eğer oyunculardan birinin nesneleri tükenmeden hamle sayısı biterse “sonuc=false” olduğu için “if(!sonuc)” yapısının içinde oyunun hamle sayısının bittiği belirtilir.Her iki oyuncu için de ellerindeki nesne sayısı kadar dönen for döngüsü kurulur ve ”getNesneler.get(i).getDayaniklilik()” methoduyla her bir nesnenin dayanikliliklari toplanır.İki oyuncu için de “setSkor” methoduyla elde edilen değerler oyuncuların skorlarına atanır.Skorlar karşılaştırılır eşit ,büyük ve küçük olma durumlarına göre sonuç belli olur ve ekrana yazdırılır. Tüm bu adımlar “oyun.txt” dosyasında ve “JOptionPane.showMessageDialog “ methoduyla arayüzde hamle hamle belirtilir. Sıradaki hamleye ekranda çıkan penceredeki ok tuşla geçilir.

Bizden istenen oyun yapısını oluşturabilmek için abstract class ve class yapılarını oluşturmamız gerekiyordu.“oyuncu” abstract classını oluşturduk. Bu classın içinde hepsi private yapıda olan “String oyuncuAdi” , “int oyuncuID” , “double skor” , “ArrayList ¡nesne¿ nesneler “ ve “ArrayList ¡String¿ nesneAdlari “ olan değişkenleri atadık.Yapıcı methoda (String ad,int id,ArrayList ¡nesne¿ nesneler,ArrayList ¡String¿ nesneAdlari) değişkenlerini gönderdik ve methodta atamasını yaptık.Çağırdığımızda oyucunun elindeki nesneleri gösterebilmesi için “nesneYazdir” methodunu oluşturduk ve nesneAdlari ArrayListimizle nesneleri yazdırabildik.Ayrıca absterct yapıda “int nesneSec” ve “void SkorGoster” methodlarını oluşturduk.

Oyuncu sınıfından kalıtılan “kullanici” classını oluşturduk.Bu classın içinde kullanıcıya özgü olan “ArrayList ¡Integer¿ temp” değişkenini oluşturduk.Bu classın yapıcı methodu (String ad,int id,ArrayList ¡nesne¿ nesneler,ArrayList ¡String¿ nesneAdlari) değişkenlerini aldı ve “super” fonksiyonu sayesinde kalıtıldığı classın değişkenlerini atadı. Üst sınıftan ovverride ettiğimiz “SkorGöster” methodumuz ile oyuncunun skorunu “getSkor “ methoduyla yazdırdık. Üst sınıftan override ettiğimiz “nesneSec” methodunda yazdığımız algoritmada ilk beş hamlede oyuncu elindeki nesnelerden herhangi birisini ikinci kez seçemiyor ve kalan hamlelerde Elindeki nesnelerden istediğini seçebiliyor.Algoritmamız: temp ArrayListimiz boşsa yani ilk hamledeysek elimizdeki nesnelerin hepsi gösterilir , istenen nesne indisi yardımıyla seçilir , indis ArrayListimize eklenir ve oyun classımıza return yapılır. ArrayListimizin boyutu eğer bir ,iki , üç veya dört ise daha önce seçilmeyen nesneler ekranda gösterilir , daha önce seçilmiş nesnelerin tekrar seçilmeme kontrolü sağlanır , sonunda seçilen indis ArrayListimize eklenir ve oyun classımıza return yapılır.ArrayListimizin boyutu eğer beşse artık oyuncu elindeki nesnelerden istediğini seçebilir , tüm nesneler ekranda gösterilir , oyuncu indis seçimi yapar ve seçim oyun classına return edilir.

Oyuncu sınıfından kalıtılan “bilgisayar” classını oluşturduk.Bu classın içinde bilgisayara özgü olan “ArrayList ¡Integer¿ temp” değişkenini oluşturduk.Bu classın yapıcı methodu (String ad,int id,ArrayList ¡nesne¿ nesneler,ArrayList ¡String¿ nesneAdlari) değişkenlerini aldı ve “super” fonksiyonu sayesinde kalıtıldığı classın değişkenlerini atadı. Üst sınıftan ovverride ettiğimiz “SkorGöster” methodumuz ile oyuncunun skorunu “getSkor “ methoduyla yazdırdık. Üst sınıftan override ettiğimiz “nesneSec” methodunda yazdığımız algoritmada ilk beş hamlede oyuncu elindeki nesnelerden herhangi birisini ikinci kez rastgele seçemiyor.Kalan hamlelerde elindeki nesnelerden hepsini rastgele atayabiliyor.Algoritmamız:temp ArrayListimiz boşsa yani ilk hamledeysek random methodu sayesinde sıfırdan elimizdeki nesne sayısına kadar bir atama yapılır, ArrayListimize eklenir ve değer oyun classımıza return yapılır.ArrayListimizin boyutu eğer bir ,iki veya üç ise daha önce seçilmiş nesnelerin tekrar seçilmeme kontrolü sağlanır, ArrayListimize eklenir ve değer oyun classımıza return yapılır.Eğer ArrayListimizin boyutu dörtse seçilen değer önceden seçilmiş mi tek tek kontrol edilir , seçilmemişse ArrayListimize eklenir ve değer oyun classımıza return yapılır. ArrayListimizin boyutu eğer beş ise bilgisayar artık elindeki nesnelerin hepsini kullanabilir.Rastgele atanan değer oyun classımıza return yapılır.Tüm bu kontroller içinde ekstra olarak eğer kullanıcının elinde bir nesne kaldıysa random methodu kullanılmaz ve direkt oyun classımıza sıfırıncı indis return edilir.

Bizden istenen oyun yapısını oluşturabilmek için abstract class ve class yapılarını oluşturmamız gerekiyordu.“nesne” abstract classını oluşturduk. Bu classın içinde ikiside private olan “double dayaniklilik” değeri ve “int seviyepuani “ oluşturduk.Nesne classının parametresiz olan yapıcı methodunda “dayaniklilik” değerine yirmi ve “seviyepuani” değerine sıfırı atadık.Böylece parametresiz oluşturduğumuz her bir nesneye başlangıçta bu değerleri atamış olduk.Çağırdığımızda her bir nesnenin değerlerini gösterebilmek için “nesnePuaniGoster” public methodunu yazdık ve o nesnenin dayanıklılık ve seviye puanlarını yazdırabildik. Ayrıca abstract yapıda “durumGuncelle (double etki)” ,”etkihesapla(Arraylist ¡String¿ nesneler ,int s2)” ve “durumGuncelle()” methodlarını yazdık.

Nesne sınıfından kalıtılan “tas” classını oluşturduk.Bu classın içinde taşlara özgü olan “int katilik” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda katılık değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her taş nesnesinin katılık değeri iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden

çıkartılıp son dayanıklılık değeri hesaplanır.”setDayaniklilik” methoduyla atama yapılır. Override ettiğimiz “durumGuncelle()” methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz “durumGuncelle()” methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

Taş sınıfından kalıtılan “agirtas” classını oluşturduk.Bu classın içinde ağır taşlara özgü olan “int sicaklik” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda sıcaklık değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her ağır taş nesnesinin sıcaklık değerine iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden çıkartılıp son dayanıklılık değeri hesaplanır.”setDayaniklilik” methoduyla atama yapılır. Override ettiğimiz “durumGuncelle()” methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz “durumGuncelle()” methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

Nesne sınıfından kalıtılan “kagit” classını oluşturduk.Bu classın içinde kağıtlara özgü olan “int nufuz” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda nüfuz değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her kağıt nesnesinin nüfuz değerine iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden çıkartılıp son dayanıklılık değeri hesaplanır.”setDayaniklilik” methoduyla atama yapılır. Override ettiğimiz “durumGuncelle()” methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz “durumGuncelle()” methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

Kağıt sınıfından kalıtılan “ozelkagit” classını oluşturduk.Bu classın içinde özel kağıtlara özgü olan “int kalinlik” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda kalınlık değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her özel kağıt nesnesinin kalınlık değerine iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden çıkartılıp son dayanıklılık değeri hesaplanır.”setDayaniklilik” methoduyla atama yapılır. Override ettiğimiz “durumGuncelle()” methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz “durumGuncelle()” methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

Nesne sınıfından kalıtılan “makas” classını oluşturduk.Bu classın içinde makaslara özgü olan “int keskinlik ” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda keskinlik değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her makas nesnesinin keskinlik değerine iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden çıkartılıp son dayanıklılık değeri hesaplanır.”setDayaniklilik” methoduyla atama yapılır. Override ettiğimiz “durumGuncelle()” methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz “durumGuncelle()” methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

Makas sınıfından kalıtılan “ustamakas” classını oluşturduk.Bu classın içinde usta makaslara özgü olan “int direnc” değerini private olarak oluşturduk. Parametresiz yapıcı methodunda Direnç değerine ikiyi atadık.Böylece parametresiz oluşturduğumuz her usta makas nesnesinin direnç değerine iki atanmış oldu.Override ettiğimiz “etkihesapla(ArrayList ¡nesne¿ nesneler int i)” methodundaki nesneler ArrayListi rakip oyuncunun nesne adlarını ve i değeri ise rakip oyuncunun o hamlede savaşmak istediği nesnesinin indisini ifade eder.Proje Tanıtım Raporunda belirtilen formüllere göre etki değerleri hesapalanır ve oyun classına return yapılır.(makas nesnesi makas ve usta makasa ; kağıt nesnesi kağıt ve özel kağıda ;taş nesnesi taş ve ağır taşa ve bunların tam tersi durumlarında da birbirlerine etki etmemektedirler.) Override ettiğimiz “durumGuncelle(double etki)” methodu

etkihesapla Methodunda return edilen etki değerini alır ve nesnenin dayanıklılık değerinden çıkartılıp son dayanıklılık değeri hesaplanır."setDayaniklilik" methoduyla atama yapılır. Override ettiğimiz "durumGuncelle()" methoduyla nesnelerin seviye puanlarını güncelledik. Oyun classımızda bir nesne diğer bir nesneyi öldürdüğünde öldüren nesnenin parametresiz "durumGuncelle()" methodunu çağırarak nesnemizin seviye puanını yirmi arttırdık.

## IV. KATKILAR

Ekibimiz projenin her bir noktasında beraber hareket etmiş olup,algoritma oluşturma,arayüz tasarımı,metod kullanımları vb. gibi tüm noktalarda ekibin her üyesi sorunun çözümünde ve projenin tamamlanmasında eşit katkılar vermiştir. Proje baştan sona ekip üyelerinin yüz yüze iletişimde olduğu şekilde yazılmıştır.

## V. SONUÇ

Sonuç olarak bu proje sonunda,Java dilinin genel yapısını,kalıtım yapsını, nesne kavramını,override ve overloading işlemlerini, dosya operasyonlarını,arayüz tasarlamayı ve dinamik olarak ilerleyen bir proje yapmayı öğrenmeye çalıştık ve bunun yanı sıra eksiklerimizin farkına vardık.

## VI. KAYNAKÇA


https://www.udemy.com/course-dashboard-redirect/?course$_id = 1910934$

https://www.youtube.com/watch?v=k1$_o A8v6Rc0$

https://www.youtube.com/playlist?list=PL4yfBYtaNjbRRGLQnrTPU2eiAB5rgi9lx