SAKARYA
ÜNİVERSİTESİ

Nesneye Dayalı Programlama

Proje 1

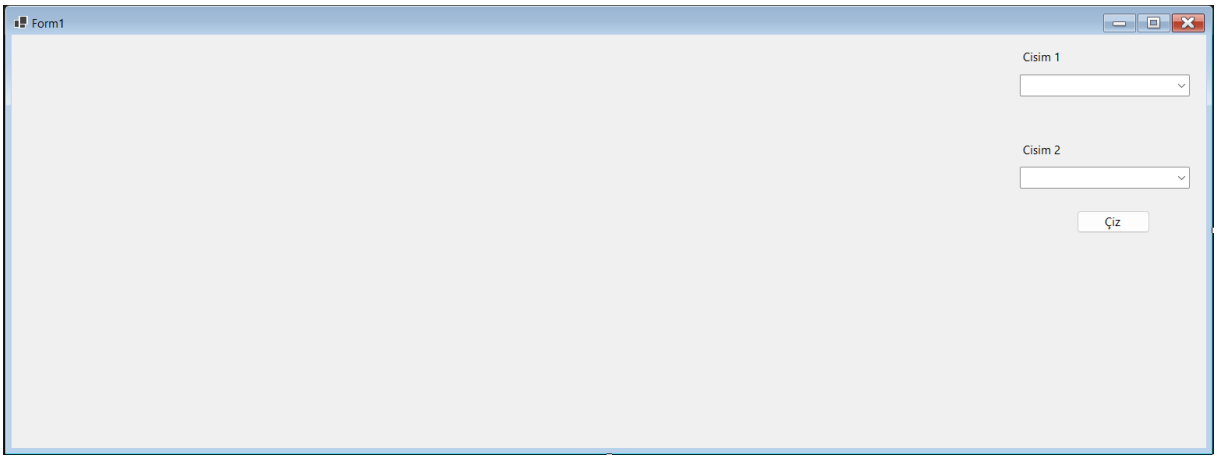
Ad Soyad: Dilara Çetin

Öğrenci NO: G221210039

Youtube sunuma buradan ulaşabilirsiniz: https://youtu.be/Bavbr4w8Udo

Form Uygulaması

İlk olarak bir form uygulaması açtım. Bu form uygulamasına iki adet combobox ve bir tane buton ekledim. Comboboxların isimlerini kullanımını kolaylaştırmak için cmbSekil1 ve cmbSekil2 yaptım. Bu comboboxlar sayesinde kullanıcıya şekil seçtireceğiz.



Daha sonra Cisimler adında bir class oluşturdum. Bu classta cisimlerin koordinatları ve çizdirme işlemleri yer alıcak. Cisimler classından kalıtım alarak da öbür cisimlerin classlarını oluşturdum.

```
56      public abstract class Cisimler
57      {
58          protected int x;
59          protected int y;
60
            7 başvuru
61          public Cisimler(int x, int y)
62          {
63              this.x = x;
64              this.y = y;
65          }
66
            9 başvuru
67          public abstract void Draw(Graphics g);
68
69      }
```

```
87      public class Cember : Cisimler
88      {
89          private int Yaricap;
90
            1 başvuru
91          public Cember(int x, int y, int Yaricap) : base(x, y)
92          {
93              this.Yaricap = Yaricap;
94          }
95
            3 başvuru
96          public override void Draw(Graphics g)
97          {
98              Pen pen = new Pen(Color.Black, 5);
99              g.DrawEllipse(pen, x - Yaricap, y - Yaricap, 2 * Yaricap, 2 * Yaricap);
00
01          }
02      }
```

Daha sonra bu Cisimler classından bir List oluşturdum. Bu classdan nesneler oluşturduğum CreateObjects adında bir metotta nesneleri List' e attım.

```
private List<Cisimler> objects;

1 başvuru
public Form1()
{
    InitializeComponent();
    objects = new List<Cisimler>();

}
```

```
private void CreateObjects()
{
    Cember cember = new Cember(190,176,129);
    Dikdortgen dikdortgen = new Dikdortgen(120, 140, 323, 122);
    Kure kure = new Kure(180, 150, 123);
    DikdortgenPrizma dprizma = new DikdortgenPrizma(199, 152, 185, 170,30);
    Yuzey yuzey = new Yuzey(15, 50, 1000, 1);
    Silindir silindir = new Silindir(145, 150, 145, 90, 30);
    Nokta nokta = new Nokta(34, 23);
    objects.Add(cember);
    objects.Add(dikdortgen);
    objects.Add(kure);
    objects.Add(dprizma);
    objects.Add(yuzey);
    objects.Add(silindir);
    objects.Add(nokta);

}
```

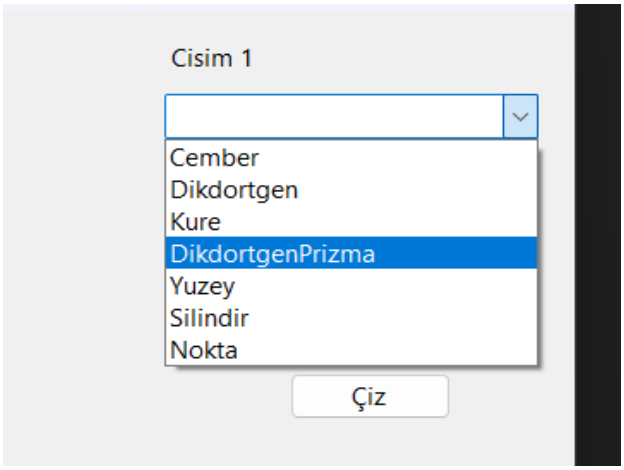
Daha sonra kullanıcının comboboxtan cisimleri seçebilmesi için List' i combobox a ekledim. Bunun için de AddObjectsToComboBox adında bir metot oluşturdum.

```
private void AddObjectsToComboBox()
{
    // Nesneleri ComboBox'a ekleme
    foreach (Cisimler obj in objects)
    {
        cmbSekil1.Items.Add(obj.GetType().Name);
    }
    foreach (Cisimler obj in objects)
    {
        cmbSekil2.Items.Add(obj.GetType().Name);
    }
}
```

Form 1 in Load eventini alıp (yani from1 ekrana yüklendiğinde) bu metotları çağırdım.

```
private void Form1_Load(object sender, EventArgs e)
{
    CreateObjects();
    AddObjectsToComboBox();


}
```

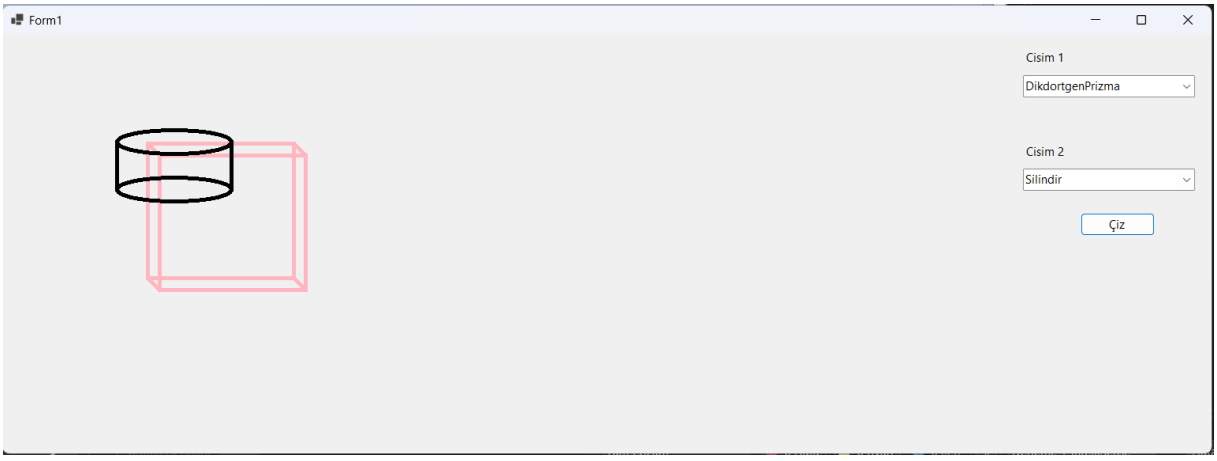
Bu sayede List' in içindeki nesneler comboboxların içine aktarılmış oldu ve kullanıcıya seçim sunuldu.



Daha sonra Çiz butonunun Click eventini çağırarak cisimleri ekrana yazdırma işlemini yaptım.

```
private void btnDraw_Click(object sender, EventArgs e)
{
    int selectedIndex = cmbSekil1.SelectedIndex;
    int selectedIndex2 = cmbSekil2.SelectedIndex;

    if (selectedIndex >= 0 && selectedIndex < objects.Count)
    {
        Graphics g = CreateGraphics();



        Cisimler selectedObject = objects[selectedIndex];
        selectedObject.Draw(g);


    }
    if (selectedIndex2 >= 0 && selectedIndex2 < objects.Count)
    {
        Graphics g = CreateGraphics();



        Cisimler selectedObject2 = objects[selectedIndex2];
        selectedObject2.Draw(g);


    }
}
```

Son olarak kullanıcı comboboxlardan cisimleri seçip çiz butonuna bastığında cisimlerin çarpışıp çarpışmadığını görmüş oldu ve form üzerinde bir çarpışma kontrolü gerçekleştirdik.



Konsol Uygulaması

İlk olarak nokta sınıfını oluşturarak cisimlerin koordinatlarını aldığım sınıfları oluşturdum.

2 boyutlu nokta için:

```
public class Nokta // 2d koordinat
    {
        int x;
        int y;
        public Nokta()
        {
            X = 0; Y = 0;
        }
        public Nokta(int x, int y)
        {
            X = x; Y = y;
        }
        public int X { get => x; set => x = value; }
        public int Y { get => y; set => y = value; }

    }
```

3 boyutlu nokta için:

```
public class Nokta3d : Nokta // 3d koordinat
    {
        int z;
        public Nokta3d() : base()
        {
            Z = 0;
        }
        public Nokta3d(int x, int y, int z) : base(x, y)
        {
            Z = z;
        }
        public int Z { get => z; set => z = value; }
    }
```

Çember için:

```
public class Cember
    {
        Nokta m; int r;
        public Cember()
        {
            M = new Nokta(); // Merkez noktası
            R = 0; // Yarıçapı

        }
        public Cember(Nokta p, int r)
        {
            M = p; R = r;
        }
        public int R { get => r; set => r = value; }
        internal Nokta M { get => m; set => m = value; }

    }
```

Daha sonra benzer şeyleri Dikdörtgen, Dikdörtgen Prizma, Silindir, Küre, Yüzey için de oluşturdum.

Yüzey için:

```
public class Yuzey { }
    /*
     * Y=0 olduğunda, x ve z ekseninin oluşturduğu taban alanı yüzey olarak
kabul ettim.
     * Yani x ve z'nin ne olduğu farketmeksizin şeklin herhangi bir kenarının
veya köşesinin Y koordinatı 0 olduğu sürece yüzeyle çarpışırlar.
     * Bundan kaynaklı Class' ın içini doldurmayıp formülde sadece cismin Y=0
doğrusuyla olan mesafesini ölçtüm.
     */
```

Yüzey için bir sınıf oluşturdum ancak içini doldurmadım çünkü yüzeyi Y=0 olacak şekilde X ve Z koordinatları olarak kabul ettim ve hesaplamalarda cismin sadece Y=0 doğrusuna olan uzaklığına baktım.

Ardından carpismalar adında bir sınıf oluşturarak cisimler arası çarpışmaları kontrol ettiğim metotları oluşturdum.

İki çember arası çarpışma:

```
public static bool CemberCarpisma(Cember c1, Cember c2)
        {
            // Eğer çemberlerin merkezleri arasındaki mesafe yarıçapları
toplamından küçük veya eşit ise çarpışma vardır.

            float d = (float)Math.Sqrt(Math.Pow((c1.M.X - c2.M.X), 2) +
Math.Pow((c1.M.Y - c2.M.Y), 2));

            if (c1.R + c2.R >= d)
                return true;
            else
                return false;
        }
```

İki silindir arası çarpışma:

```
public static bool SilindirCarpisma(Silindir s1, Silindir s2)
        {

             //Eğer silindirlerin merkezlerinin Y koordinatları arasındaki
mesafe yüksekliklerinin yarısından küçük veya eşitse
             // ve merkezlerinin X ve Z koordinatları üzerindeki mesafesi
yarıçapları toplamından küçük veya eşitse çarpışma vardır.

            float d = (float)Math.Sqrt(Math.Pow((s1.M.X - s2.M.X),2) +
Math.Pow((s1.M.Z - s1.M.Z), 2));

            if ((s1.R + s2.R) >= d && Math.Abs(s1.M.Y - s2.M.Y) <= ((s1.H +
s2.H) / 2))
                return true;
            else
                return false;
        }
```

Kalan çarpışmalar için metotları oluşturdum.

Ardından denetim sınıfını oluşturdum ve buradan menüyü çağırdım:

```
internal class denetim
    {
        public denetim() { }
```

```
        public static void Menu()
        {
            Console.WriteLine("---------- Çarpışma Denetimi (Collision
Detection)----------");
            Console.WriteLine();
            Console.WriteLine("1-  Nokta, dikdörtgen çarpışma denetimi.");
            Console.WriteLine("2-  Nokta, çember çarpışma denetimi.");
            Console.WriteLine("3-  Dikdörtgen, dikdörtgen çarpışma denetimi.");
            Console.WriteLine("4-  Dikdörtgen, çember çarpışma denetimi.");
            Console.WriteLine("5-  Çember, çember çarpışma denetimi.");
            Console.WriteLine("6-  Nokta, küre çarpışma dnetimi.");
            Console.WriteLine("7-  Nokta, dikdörtgen prizma çarpışma
denetimi.");
            Console.WriteLine("8-  Nokta, silindir çarpışma denetimi.");
            Console.WriteLine("9-  Silindir, silindir çarpışma denetimi.");
            Console.WriteLine("10- Küre, küre çarpışma denetimi.");
            Console.WriteLine("11- Küre, silindir çarpışma denetimi.");
            Console.WriteLine("12- Yüzey, küre çarpışma denetimi.");
            Console.WriteLine("13- Yüzey, dikdörtgen prizma çarpışma
denetimi.");
            Console.WriteLine("14- Yüzey, silindir çarpışma denetimi.");
            Console.WriteLine("15- Kare, dikdörtgen prizma çarpışma
denetimi.");
            Console.WriteLine("16- Dikdörtgen prizma, dikdörtgen prizma
çarpışma denetimi.");
            Console.WriteLine();
            Console.Write("Lütfen menüden bir seçim yapınız:");
        }
    }
```

Ardından main metodun altında menüyü çağırarak kullanıcıdan seçim yapmasını istedim. Kullanıcının girdiği seçime göre if-else yapısını kullanarak carpismalar sınıfının metotlarına örnek verdim ve çarpışmaların gerçekleşip gerçekleşmediğini kontrol ettim.

```
static void Main(string[] args)
        {
            baslangic:

            denetim.Menu();
            string secim = Console.ReadLine();

            Console.Clear();

            // carpismalar sınıfının fonksiyonlarına burada örnek verdim.

            if (secim == "1")
            {
                Console.WriteLine("Nokta, dikdörtgen çarpışma denetimine
hoşgeldiniz.");
                if(carpismalar.NoktaDortgenCarpisma(new Nokta(6,8),new
Dikdortgen(new Nokta(4,2),6,10)))
                    Console.WriteLine("Çarpışma var.");
                else
                    Console.WriteLine("Çarpışma yok.");
                Console.WriteLine();
                Console.WriteLine("Lütfen devam etmek için herhangi bir tuşa
basınız.");
                Console.ReadLine();
                Console.Clear();
            }
            if (secim == "2")
            {
                Console.WriteLine("Nokta, çember çarpışma denetimine
hoşgeldiniz.");
```

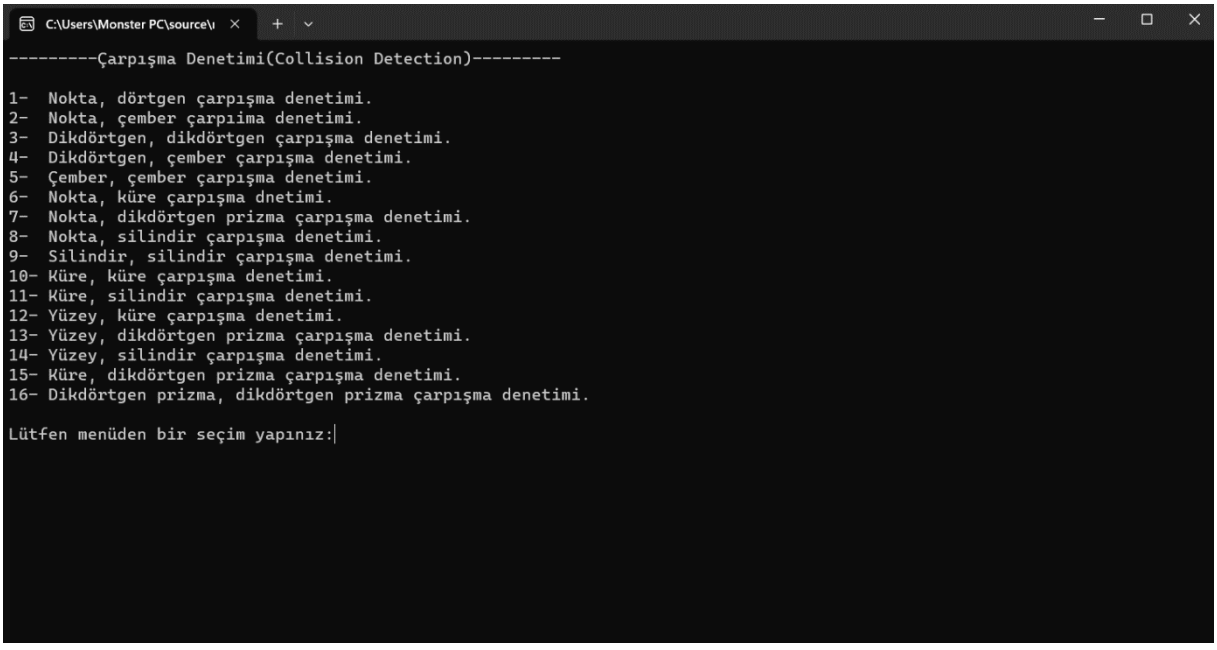
```
                if(carpismalar.NoktaCemberCarpisma(new Nokta(3,1),new
Cember(new Nokta(5,2),8)))
                    Console.WriteLine("Çarpışma var.");
                else
                    Console.WriteLine("Çarpışma yok.");
                Console.WriteLine();
            }
```

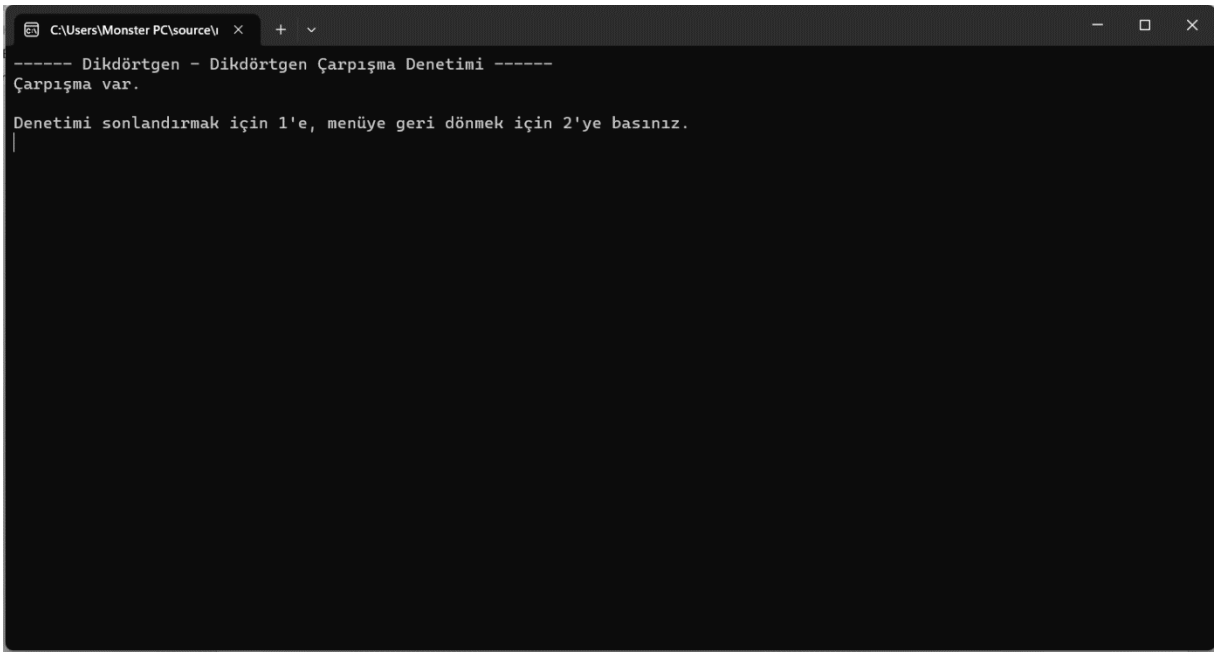
Daha sonra öbür koşullar için de örnekler oluşturdum ve döngüden çıkarttım:

```
Console.WriteLine("Lütfen devam etmek için herhangi bir tuşa basınız.");
                Console.ReadLine();
                Console.Clear();

            Console.WriteLine("Menüye geri dönmek için 1' e,");
            Console.WriteLine("Denetimi sonlandırmak için 2' ye basınız.");
            string secim2 = Console.ReadLine();
            Console.Clear();
            if (secim2 == "1")
                goto baslangic;


        }
    }
```

Çarpışmaları kontrol ettikten sonra baslangic etiketi yardımıyla kullanıcıya menüye geri dönme veya denetimi sonlandırma şansı tanıdım.

Bu şekilde konsol uygulaması üzerinden cisimlerin çarpışma kontrolü yapılmış oldu.




C:\Users\Monster PC\source\
------ Dikdörtgen - Dikdörtgen Çarpışma Denetimi ------
Çarpışma var.
Denetimi sonlandırmak için 1'e, menüye geri dönmek için 2'ye basınız.