```
public class Person{
	public int enerji;
	private static int bilgi;
	public Person(){
		enerji=100;
	}
	public Person (int e){
		enerji=e;
	}

	public void uyu(){
		enerji=enerji + 2;
	}
	public void kos(){
		enerji=enerji-3;
	}
	public void ogren(){
		bilgi=bilgi+50;
	}
	public int bilgi(){
		return bilgi;
	}
}
public class Main{
	public static void main(String[] args){
		Person p1 = new Person();
		Person p2 = new Person(100);
		for(int i=0;i<10;i++) p1.uyu();
		p1.kos();
		for(int k=0;k<5;k++) p2.uyu();
		p2.kos();
		System.out.println("p1 enerji="+p1.enerji);
		System.out.println("p2 enerji="+p2.enerji);
		p1.ogren();
		p2.ogren();
		System.out.println("p1 bilgi="+p1.bilgi());
		System.out.println("p2 bilgi="+p2.bilgi());
		Person p3 = new Person();
		Person p4 = new Person(400);
		Person p5=p3;
		p4.kos();
		p3=p4;
		System.out.println("p3 enerji=" + p3. enerji);
		System.out.println("p5 enerji =" + p5. enerji);
	}
}
```

(15 puan)

S1- a) p1 ve p2 nesnelerinin enerjileri olarak hangi değerler ekrana yazdırılır?(ilk 2 system.out.println çıktısı?)

p1 enerji = 117
p2 enerji=107

S1-b) p1 ve p2 bilgi değerleri ekran çıktısı?(3. ve 4. system.out.println çıktılar?)

// *bilgi* değişkeni static olarak tanımlandığından dolayı sınıf değişkeni haline gelmiştir. Bu şekilde her *Person* nesnesi için ayrı *bilgi* değişkeni olmayıp *Person* sınıfına ait tek bir *bilgi* değişkeni olması sağlanmış.(bu yüzden hep aynı değişken farklı nesneler (p1 ve p2) tarafından artırılıyor, önce 50 sonra 100.. Ekrana yazdırılınca son değer olan 100 yazdırılıyor)

p1 bilgi=100
p2 bilgi=100

S1-c) p3 ve p5 nesnelerinin enerji olarak ekran çıktısı?(son 2 system.out.println?)

p3 enerji=397
p5 enerji =100

S1-d) Bu kodda enerji değişkeni ve system.out.println komutları ile enerji seviyesi yazdırmada nesneye dayalı programlamada yazılım geliştirme prensiplerine aykırı durum nedir?

enerji değişkeninin public tanımlanması ve başka sınıflardan erişime açık olması

S-2) (15 puan)

```
Public interface Arayuz1 {
	public void is1();
	public void is2();
}
Public class sinifA implements Arayuz1 {
	Public void is1();
	Public void is2();
}
Public class sinifB extends sinifA {
	Public void is3();
}
Public class sinifC extends sinifA {
	Public void is4();
}
```

Hangileri doğru,derleme hatası,çalışma zamanı hatası?

a. **Arayuz1 nesne=new sinifA();**

   Doğru

b. **sinifA nesne02=new sinifC();**

   ((sinifB)nesne02).is3();

   Run-time Error (*sinifC sinifB*'ye dönüştürülemez. Aralarında hiçbir şekilde hiyeraşik kalıtımsal bir ilişki söz konusu değildir. Ancak böyle bir hatanın olduğu compile-time'da bilinememekte -derleyici *nesne02*'yi *sınıfA* zanneder ve *sinifA* örneği(instance) *sinifB*'ye cast ediliyor sanar. Runtime'da *nesne02*'nin referans ettiği örneğin aslında *sinifC* olduğunu anlar ve istisna(ClassCastException) mesajını yazdırır.)

c. **sinifB nesne03=new sinifA();**

   Compile-time Error (Tip uyuşmazlığı(Typemismatch). Her *sinifB* bir *sinifA*'dir ancak her *sinifA* bir *sinifB* değildir. Burada derleyici eşitliğin sağ tarafında *sinifB* veya altsınıflarından bir örnek(instance) oluşturulmasını beklemektedir.)

d. **sinifA nesne04=new sinifC();**

   nesne04.is4();

   Compile-time Error (Derleyici *sinifA* sınıfında *is4()* metodu arar ve bulamadığı için tanımlanmamış metot(undefined method) hatası verir)

e. **Arayuz1 nesne05=new sinifB();**

   nesne05.is1();

   Doğru

S-3) (10 puan)

```
public class yeniException{
        public yeniException(){}
        public static void main(String[args]){
                yeniException e=new yeni exception();

                // yorum
        }
        public int amethod(int p){

                try{

                        İnt sonuç=100/p;
                        System.out.println(“sonuc=”+sonuc);
                }

                catch (ArithmeticException ae){
                        system.out.println(“sifira bölme yapildi”);
                        return 0;

                }
                catch (RuntimeException re){

                        System.out.println(“runtime aykırı durumu”);
                }

                finaly{
                        System.out.println(“finaly blok içinde”);
                        return 1;
                }
        }
}
```

a-yorum satırının altına system.out.println (e.amethod()); yazdığımızda çıktı?

// Önce ArithmeticException yakalanır ve ilgili blok çalıştırılır. İstisna durumu olsa da olmasa da finally bloğu her zaman çalıştırılır. Dolaysıyla metodtan dönen değer 1(return 1; den dolayı) olacaktır.

```
sifira bölme yapildi
finaly blok içinde
1
```

b- yorum satırının altına system.out.println (e.amethod(10)); yazdığımızda çıktı?

```
sonuc = 10
finaly blok içinde
1
```

S4-) (25 puan)

```
public class classD{

        public static void D1() {}

        public static void D2() {}

        public static void D3() {}

        public static void D4() {}

}

public class classAA{

        public void methodA(){

        classD.D1();
        classD.D2();
        for (int i=0;i<15;i++)
                System.out.println("classAA method"+i);

        classD.D3();
        classD.D4();
        for (int=0;i<25;i++)
                system.out.println("j değeri:"+j);
}

public class classBB{
        classD.D1();
        classD.D2();
        for (int i=0;i<10;i++)
                System.out.println("Merhaba dünya");

        classD.D3();
        classD.D4();
        for (int i=0;i<5;i++)
        System.out.println("Merhaba java!")
}
```

Template method yapılandırarak (refactoring) yazınız.Bu şekilde yeniden yapılandırmanın getireceği avantajları yazınız.

NOT: Bu çözümde sadece template metod tasarım deseninin doğru uygulanmasını göz önünde bulundurdum. Sorudaki sınıf ve metotlar gerçek/kullanılır program olmaktan uzak birer taslak oldukları için OOP ilkelerini tam olarak uygulamaya özen göstermedim. Bu nedenle bu çözüm, esnek ve etkin nesneye yönelik programlamaya örnek değildir.

```
public abstract class ClassWithTemplateMethod
{
	public abstract void yazdir();

	public void method(){
		classD.D1();
		classD.D2();
		yazdir();
		classD.D3();
		classD.D4();
		yazdir();
	}
}


public class classAA extends ClassWithTemplateMethod
{
	private int sayi = 15;
	private String mesaj = "classAA method ";

	public void yazdir()
	{
		for(int i = 0; i < sayi; i++)
		{
			System.out.println(mesaj+i);
		}

		this.sayi = 25;
		this.mesaj = "j degeri: ";

	}
}

public class classBB extends ClassWithTemplateMethod{
	private int sayi = 10;
	private String mesaj = "Merhaba Dunya";

	public void yazdir()
	{
		for(int i = 0; i < sayi; i++)
		{
			System.out.println(mesaj);
		}

		this.sayi = 5;
		this.mesaj = "Merhaba Java!";
	}

}
```

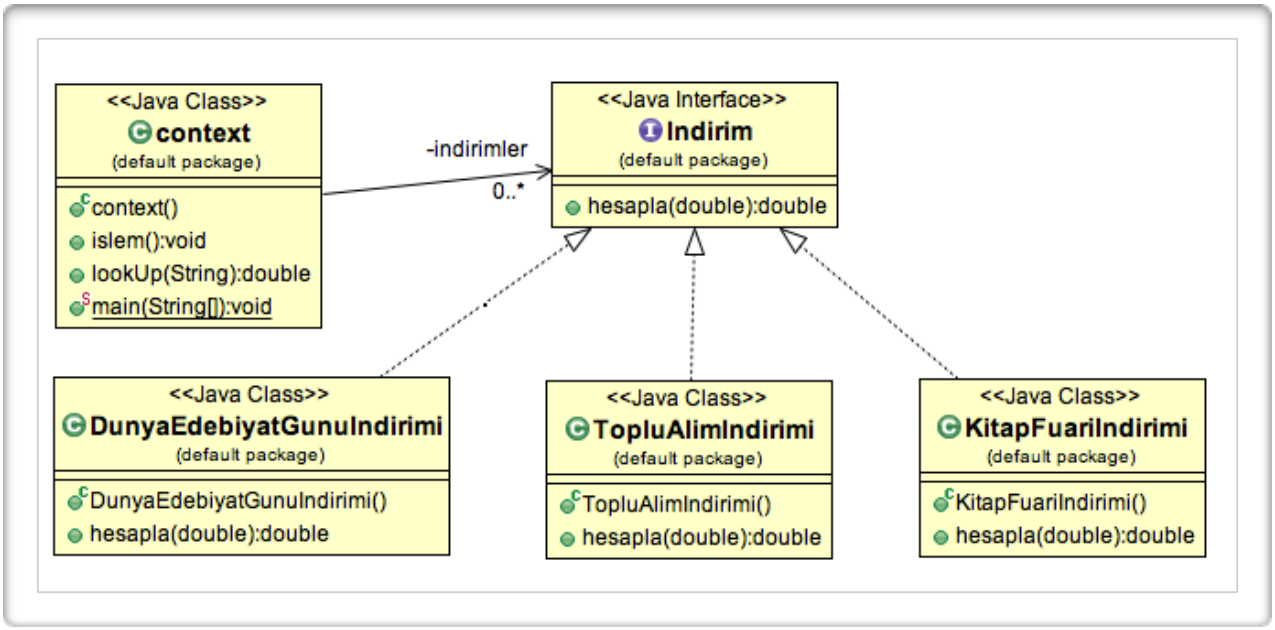
S5-) (35 puan)

Bir kitapçı müşterilerine çeşitli indirim stratejileri uygulamaktadır. Toplu alımlarda (10kitap üzeri) %20, Dünya Edebiyat Günü %15 ve Kitap Fuarı tarihlerinde %10 indirim vardır.

Diğer taraftan müşteriler ödedikleri son fiyat ilgili strateji hesaplama her yeni tutar üzerinden arka arkaya uygulanması ile oluşmaktadır. Diğer deyişle örnek olarak, yukarıda söz edilen 3 indirim türü de uygulanacak ise önce tutar %20 düşmekte, daha sonra yeni hesaplanan tutar üzerinden %15 daha düşmekte ve en son olarak da son bulunan tutar üzerinden %10 daha düşmektedir.

Kitapçı sistemini "strateji" tasarım desenini kullanarak tasarlayınız.

a-UML sini çiziniz.



b-public double hesapla (double tutar) imzalı metodu yazınız.

```
public interface Indirim
{
	public double hesapla(double tutar);
}

public class KitapFuariIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.9;
	}
}
```

```
public class TopluAlimIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.8;
	}
}

public class DunyaEdebiyatGunuIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.85;
	}
}
```

c-context tarafı için boşlukları doldurunuz.

Public class context{

#1 Aşağıya strateji nesneleri tutacak ArrayList tanımlayın.

```
	private ArrayList<Indirim> indirimler;
```

#2 ArrayList oluşturan parametresiz constructor yazınız.

```
		public context()
	{
		indirimler = new ArrayList<Indirim>();
	}
```

	public void işlem{

		//klavyeden veri alınmıyor programdan veriliyor

		String ISBN=”9999”;

		double fiyat=lookUp(ISBN);

		int miktar=20;

		double tutar=miktar*fiyat;

		String date=”23.01.2014” //edebiyat ve fuar günü//

#3 miktar ve günün tarihi kontrol edilerek ArrayListe eklenir.

```
		if(date == "23.01.2014")
		{
			// Dunya Edebiyat Gunu Indirimi uygulansin
			indirimler.add(new DunyaEdebiyatGunuIndirimi());
		}
```

```
		if(miktar > 10)
		{
			// Toplum Alim indirimi...
			indirimler.add(new TopluAlimIndirimi());
		}
```

#4 ilgili strateji nesneleri zincirleme uygulanarak son yeni tutar hesplanır.

```
		double yeniTutar = tutar;
		Iterator<Indirim> dolasici = indirimler.iterator();
		while(dolasici.hasNext())
		{
			Indirim siradakiIndirim= dolasici.next();
			yeniTutar = siradakiIndirim.hesapla(yeniTutar);
		}

		System.out.println("Ilk tutar: " + tutar);
		System.out.println("Yeni tutar: " + yeniTutar);

	}// islem metodu sonu

	public double lookUp(String ISBN)
	{
		return 100.0; //veri tabaninda kitap fiyati
	}

	public static void main(String[] args)
	{
		context ornek = new context();
		ornek.islem();
	}

}// context sinifi sonu
```

5.Soru - Çözümün tamamı:

context.java

```
import java.util.ArrayList;
import java.util.Iterator;

public class context
{
	private ArrayList<Indirim> indirimler;
	public context()
	{
		indirimler = new ArrayList<Indirim>();
	}

	public void islem()
	{
		//klavyeden veri alinmiyor programdan veriliyor
		String ISBN="9999";
		double fiyat=lookUp(ISBN);
		int miktar = 20;
		double tutar=miktar*fiyat;
		String date="23.01.2014"; //edebiyat ve fuar günü//

		if(date.equals(“23.01.2014"))
		{
			// Dunya Edebiyat Gunu Indirimi uygulansin
			indirimler.add(new DunyaEdebiyatGunuIndirimi());
		}
		if(miktar > 10)
		{
			// Toplum Alim indirimi...
			indirimler.add(new TopluAlimIndirimi());
		}

		double yeniTutar = tutar;
		Iterator<Indirim> dolasici = indirimler.iterator();
		while(dolasici.hasNext())
		{
			Indirim siradakiIndirim= dolasici.next();
			yeniTutar = siradakiIndirim.hesapla(yeniTutar);
		}

		System.out.println("Ilk tutar: " + tutar);
		System.out.println("Yeni tutar: " + yeniTutar);
	}

	public double lookUp(String ISBN)
	{
		return 100.0; //veri tabaninda kitap fiyati
```

```
	}

	public static void main(String[] args)
	{
		context ornek = new context();
		ornek.islem();
	}
}
```

Indirim.java

```
public interface Indirim
{
	public double hesapla(double tutar);
}
```

KitapFuariIndirimi.java

```
public class KitapFuariIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.9; // tutar-tutar*0.1
	}
}
```

TopluAlimIndirimi.java

```
public class TopluAlimIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.8;// tutar-tutar*0.2
	}
}
```

DunyaEdebiyatGunuIndirmi.java

```
public class DunyaEdebiyatGunuIndirimi implements Indirim
{
	public double hesapla(double tutar)
	{
		return tutar*0.85; // tutar-tutar*0.15
	}
}
```

Umarım faydalı olur :)

3 Şubat 2014 Ferhat Ezizi