



Nesneye Dayalı Programlama Ödevi

FATİH HAS

Soru 1.5) Sınıf ve nesne kavramlarını açıklayınız. Sınıf ve nesnenin kullanıldığı örnek yapınız.

```
JavaApplication10.java x
Source History
1  /*Soru 1.5
2  Sınıf ve nesnenin özellikleri örnek içinde açıklama
3  satırlarında açıklanmıştır.*/
4
5  package javaapplication10;
6  import java.util.Scanner;
7  class Daire
8  {
9      //Sınıf soyut bir veri tipidir.
10     //İçinde somut olan nesnelere taşır.
11     public int r;
12     public double alan()
13     {
14         return 3 * r * r;
15     }
16     public int cevre()
17     {
18         return 2 * 3 * r;
19     }
20 }
21 public class JavaApplication10 {
22     public static void main(String[] args) {
23         Scanner gir=new Scanner(System.in);
24         int r=gir.nextInt();
25         Daire x=new Daire();
26         /*Tanımlanan daire adlı sınıftan x
27         isimli nesne oluşturulmuş.
28         Nesnelere soyut sınıflardaki somut ifadelerdir.*/
29         x.r=r;
30         System.out.println(x.alan());
31         System.out.println(x.cevre());
32     }
33 }
34
```

Soru 1.12) Kullanıcı tarafından girilen iki sayıyla dört işlem yapan ve konsola yazan uygulama geliştiriniz.

```
JavaApplication9.java x
Source History
1  /*Soru 1.12
2  Kullanıcı tarafından girilen iki sayının
3  toplama, çıkarma, carpma ve bolme işlemlerini yapan uygulama...
4  */
5  package javaapplication9;
6  import java.util.Scanner;
7  public class JavaApplication9 {
8      public static void main(String[] args) {
9          Scanner a=new Scanner(System.in);
10         Scanner b=new Scanner(System.in);
11         int sayil,sayi2;
12         System.out.println("İki tam sayı girin.");
13         sayil=a.nextInt();
14         sayi2=b.nextInt();
15         int top_cozum=sayil+sayi2;
16         int cik_cozum=sayil-sayi2;
17         int carp_cozum=sayil*sayi2;
18         int bol_cozum=sayil/sayi2;
19         System.out.println("Sayıların toplami: "+top_cozum);
20         System.out.println("Sayıların farki: "+cik_cozum);
21         System.out.println("Sayıların carpimi: "+carp_cozum);
22         System.out.println("Sayıların bolumu: "+bol_cozum);
23     }
24 }
25 }
26
```

Soru 2.1) 1'den 100'e kadar olan sayıları ekrana yazan bir uygulama geliştiriniz. Aynı uygulamayı 100'den 1'e yazacak şekilde uyarlayınız.

```
JavaApplication11.java x
Source History
1  /*Soru 2.1
2  !'den 100'e kadar sayilari yazan uygulama
3  */
4  package javaapplication11;
5
6  public class JavaApplication11 {
7
8      public static void main(String[] args) {
9          for(int i=1;i<=100;i++){
10             System.out.println("Döngü: "+i);
11         }
12     }
13 }
14
15 }
16
```

```
JavaApplication12.java x
Source History
1  /*Soru 2.1
2  1'den 100'e sayilari yazan uygulamanin 100'den
3  1'e uyarlanmis hali.
4  */
5  package javaapplication12;
6
7  public class JavaApplication12 {
8
9      public static void main(String[] args) {
10         for(int i=100;i>=1;i--){
11             System.out.println("Döngü: "+i);
12         }
13     }
14 }
15
16 }
17
```

Soru 2.7) Üçlü if-else operatörünün kullanımı ;

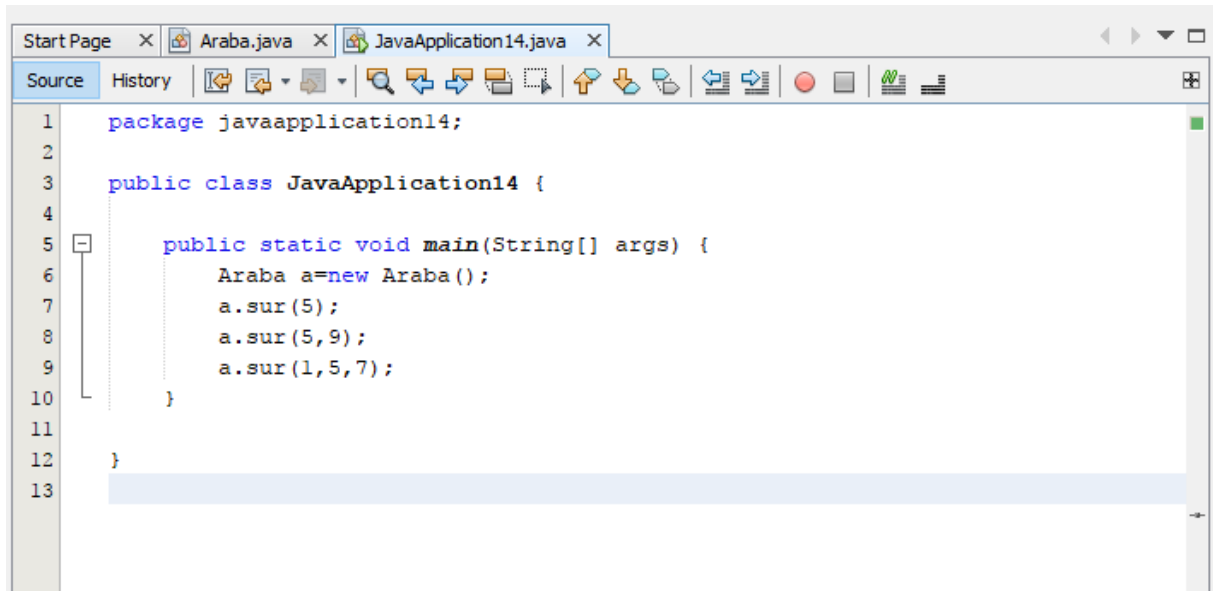
```
JavaApplication13.java x
Source History
1  /*Soru 2.7
2  Üçlü if-else kullanışı
3  */
4  package javaapplication13;
5  import java.util.Scanner;
6  public class JavaApplication13 {
7
8      public static void main(String[] args) {
9          Scanner i=new Scanner(System.in);
10
11          System.out.println("0'dan 3'e kadar bir adet Sayı giriniz..");
12          int x=i.nextInt();
13          if(x==0){
14              System.out.println("Sayı: "+x);
15          }
16          else if(x==1){
17              System.out.println("Sayı: "+x);
18          }
19          else if(x==2){
20              System.out.println("Sayı: "+x);
21          }
22          else{
23              System.out.println("Sayınız 3'e esittir..");
24          }
25
26      }
27
28  }
29
```

Soru 3.5) Araba adında sınıf oluşturup içinde 3 tane adaş metot yazınız.

Araba sınıfı;

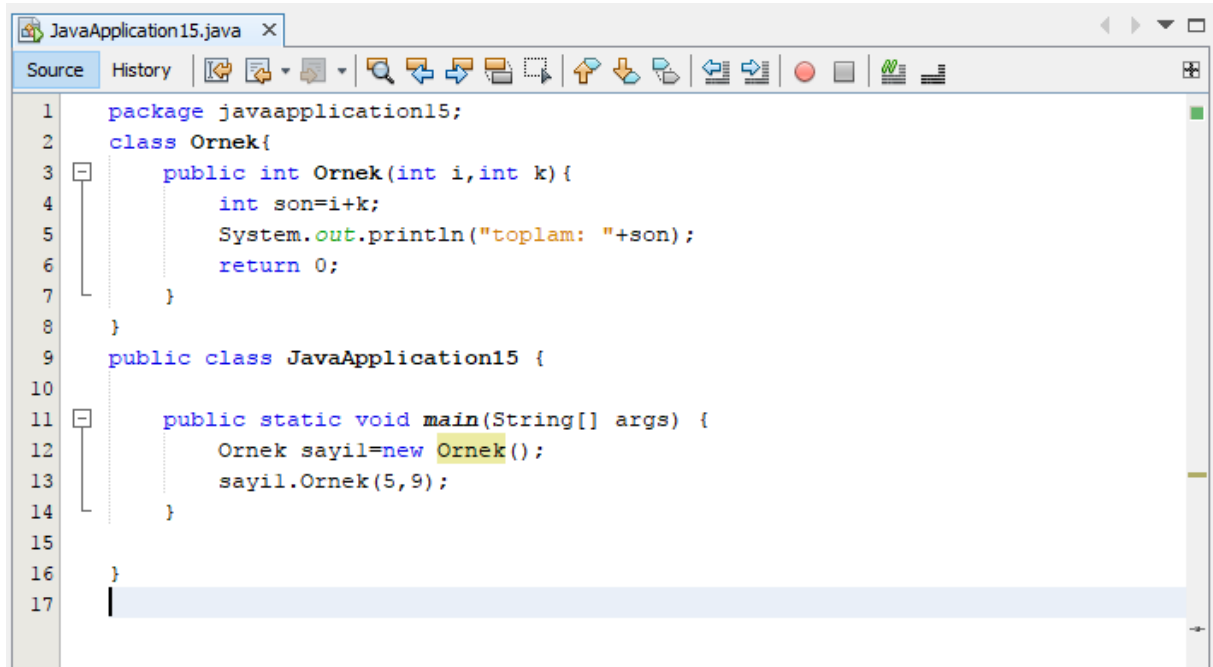
```
Start Page x Araba.java x JavaApplication14.java x
Source History
1
2  package javaapplication14;
3
4  public class Araba {
5      public void sur(int sayil){
6          System.out.println(sayil+=1);
7      }
8      public void sur(int sayil ,int sayi2){
9          System.out.println(sayil+sayi2);
10     }
11     public void sur(int sayil,int sayi2,int sayi3){
12         System.out.println(sayil+sayi2+sayi3);
13     }
14 }
15
```

Main yordamı;



```
1 package javaapplication14;
2
3 public class JavaApplication14 {
4
5     public static void main(String[] args) {
6         Araba a=new Araba ();
7         a.sur(5);
8         a.sur(5,9);
9         a.sur(1,5,7);
10    }
11
12 }
13
```

Soru 3.1) Varsayılan yapılandırıcısı olan bir sınıf yazınız. Bu sınıfı daha sonradan bir uygulama içinde new ile oluşturunuz.



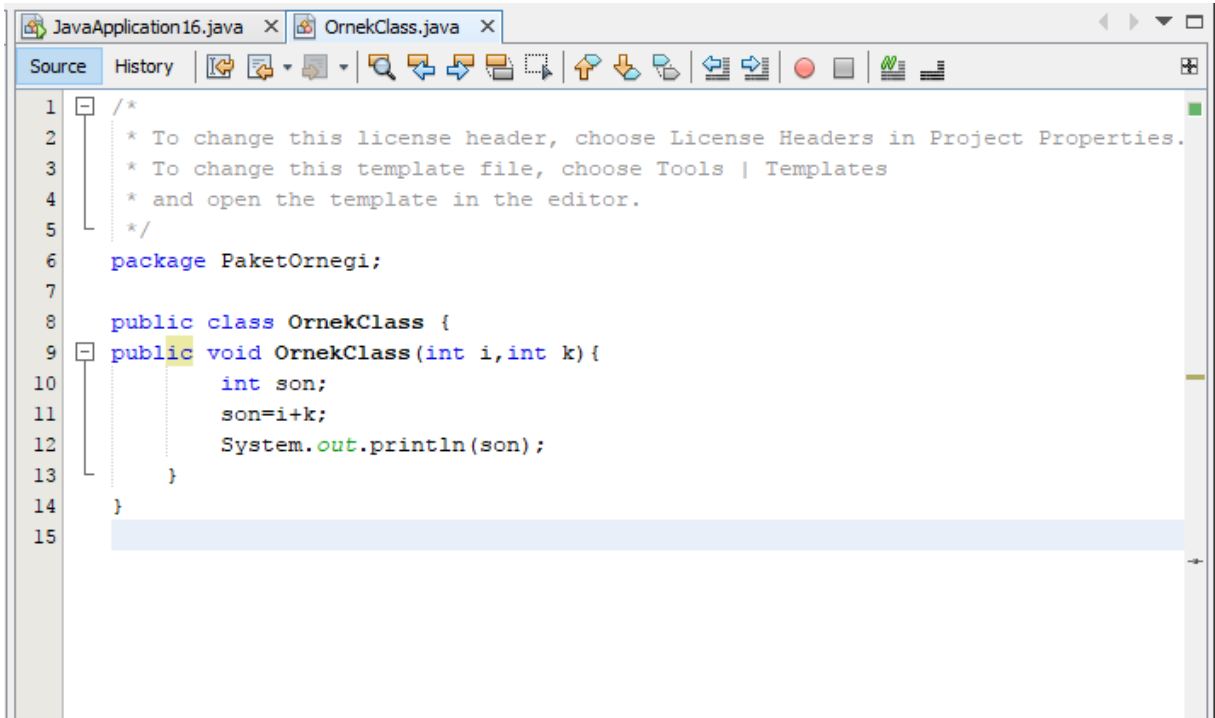
```
1 package javaapplication15;
2 class Ornek{
3     public int Ornek(int i,int k){
4         int son=i+k;
5         System.out.println("toplam: "+son);
6         return 0;
7     }
8 }
9 public class JavaApplication15 {
10
11     public static void main(String[] args) {
12         Ornek sayil=new Ornek();
13         sayil.Ornek(5,9);
14     }
15
16 }
17
```

Soru 4.9) Java programlama dilinde paket ne anlama gelmektedir. Örnek vererek açıklayınız.

Java'da paket kavramı ;

Aynı amaç üzerinde çalışan kod bloklarını bir arada tutmak için kullanılır.

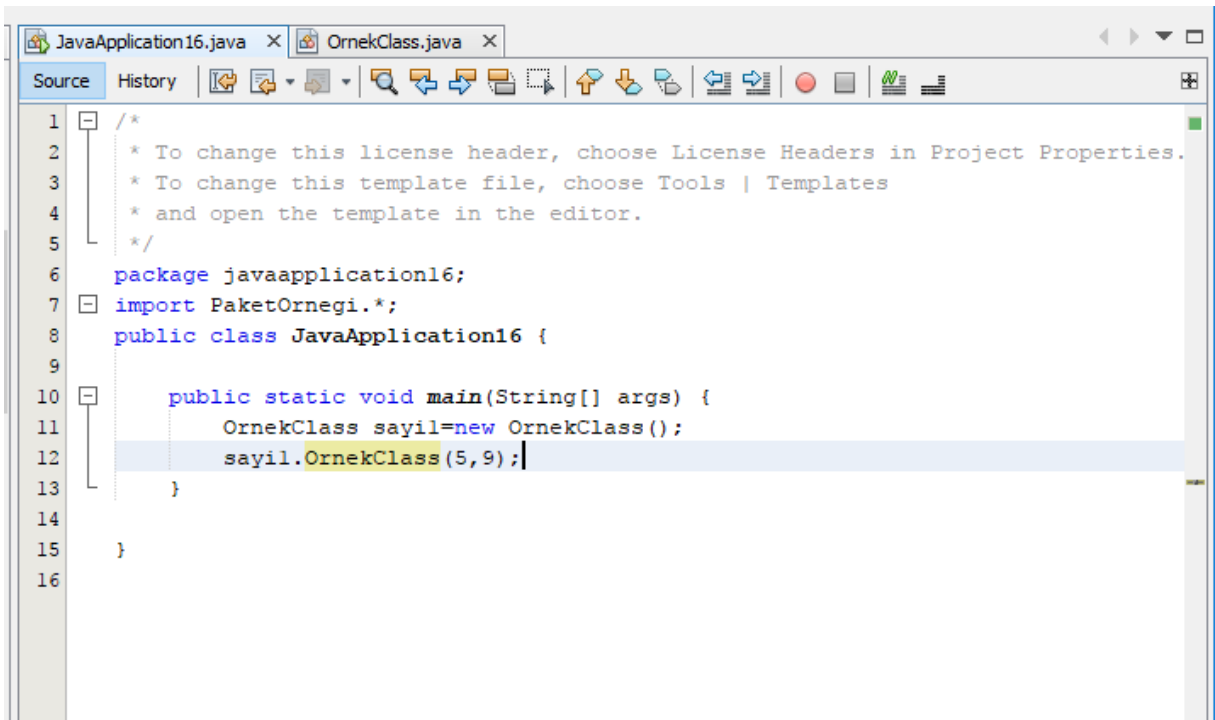
javaApplication16 projesinin altında oluşturulan paket ve pakete ait sınıf;



```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package PaketOrnegi;
7
8  public class OrnekClass {
9  public void OrnekClass(int i,int k) {
10     int son;
11     son=i+k;
12     System.out.println(son);
13 }
14 }
15
```

Main yordamı;

import anahtar sözcüğü ile tanımlanmış paket;



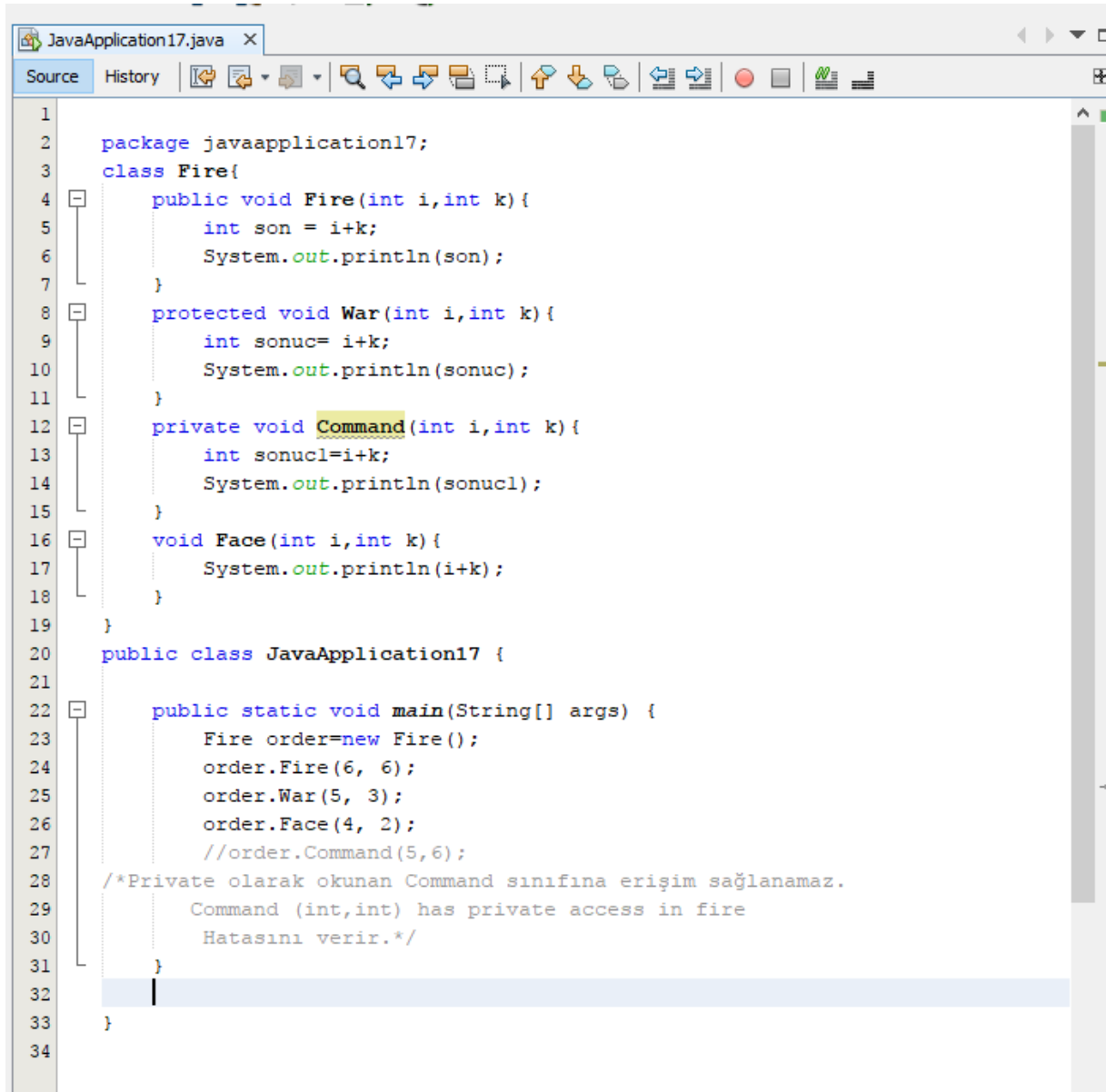
```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package javaapplication16;
7  import PaketOrnegi.*;
8  public class JavaApplication16 {
9
10     public static void main(String[] args) {
11         OrnekClass sayil=new OrnekClass();
12         sayil.OrnekClass(5,9);
13     }
14
15 }
16
```

Soru 4.6)

Public, protected, friendly, private erişim belirleyicilerine sahip global alanlar ve yordamlar olan bir sınıf yazınız.

	Aynı Paket	Ayrı Paket	Ayrı paket Türemiş
Public	Erişimli	Erişimli	Erişimli
Protected	Erişimli	Erişemez	Erişimli
Friendly	Erişimli	Erişemez	Erişemez
Private	Erişemez	Erişemez	Erişemez

Private ile aynı sınıftan dahi erişim sağlayamayız.



```
1
2 package javaapplication17;
3 class Fire{
4     public void Fire(int i,int k){
5         int son = i+k;
6         System.out.println(son);
7     }
8     protected void War(int i,int k){
9         int sonuc= i+k;
10        System.out.println(sonuc);
11    }
12    private void Command(int i,int k){
13        int sonucl=i+k;
14        System.out.println(sonucl);
15    }
16    void Face(int i,int k){
17        System.out.println(i+k);
18    }
19 }
20 public class JavaApplication17 {
21
22     public static void main(String[] args) {
23         Fire order=new Fire();
24         order.Fire(6, 6);
25         order.War(5, 3);
26         order.Face(4, 2);
27         //order.Command(5,6);
28         /*Private olarak okunan Command sınıfına erişim sağlanamaz.
29         Command (int,int) has private access in fire
30         Hatasını verir.*/
31     }
32 }
33
34
```

Soru 5.1) T sınıfı ve bu sınıftan türetilmiş Z sınıfı oluşturunuz. T sınıfının içerisinde 3 adet nesne yordamı (statik olmayan yordam) oluşturunuz ve bu yordamların türetilmiş Z sınıfında kullanılabilirliğini kanıtlayınız.

T sınıfında kullandığımız yordamları Z sınıfında kullandığımızda Override işlemi yapar.

Overloading işlemi ise Z'nin T sınıfında türememiş olması gerekmektedir.

```
JavaApplication18.java x
Source History
1 package javaapplication18;
2 class T{
3     public void ilkislem(){
4         System.out.println("Hello World!");
5     }
6     public void ikinciislem(){
7         System.out.println("Whats app?");
8     }
9     public void ucuncuislem(){
10        System.out.println("How are you?");
11    }
12 }
13 class Z extends T{
14     @Override
15     public void ilkislem(){
16         System.out.println("Hello World");
17     }
18     @Override
19     public void ikinciislem(){
20         System.out.println("Whats app?");
21     }
22     @Override
23     public void ucuncuislem(){
24         System.out.println("How are you?");
25     }
26 }
27 public class JavaApplication18 {
28
29     public static void main(String[] args) {
30         Z ar=new Z ();
31         ar.ilkislem();
32         ar.ikinciislem();
33         ar.ucuncuislem();
34     }
35
36 }
```

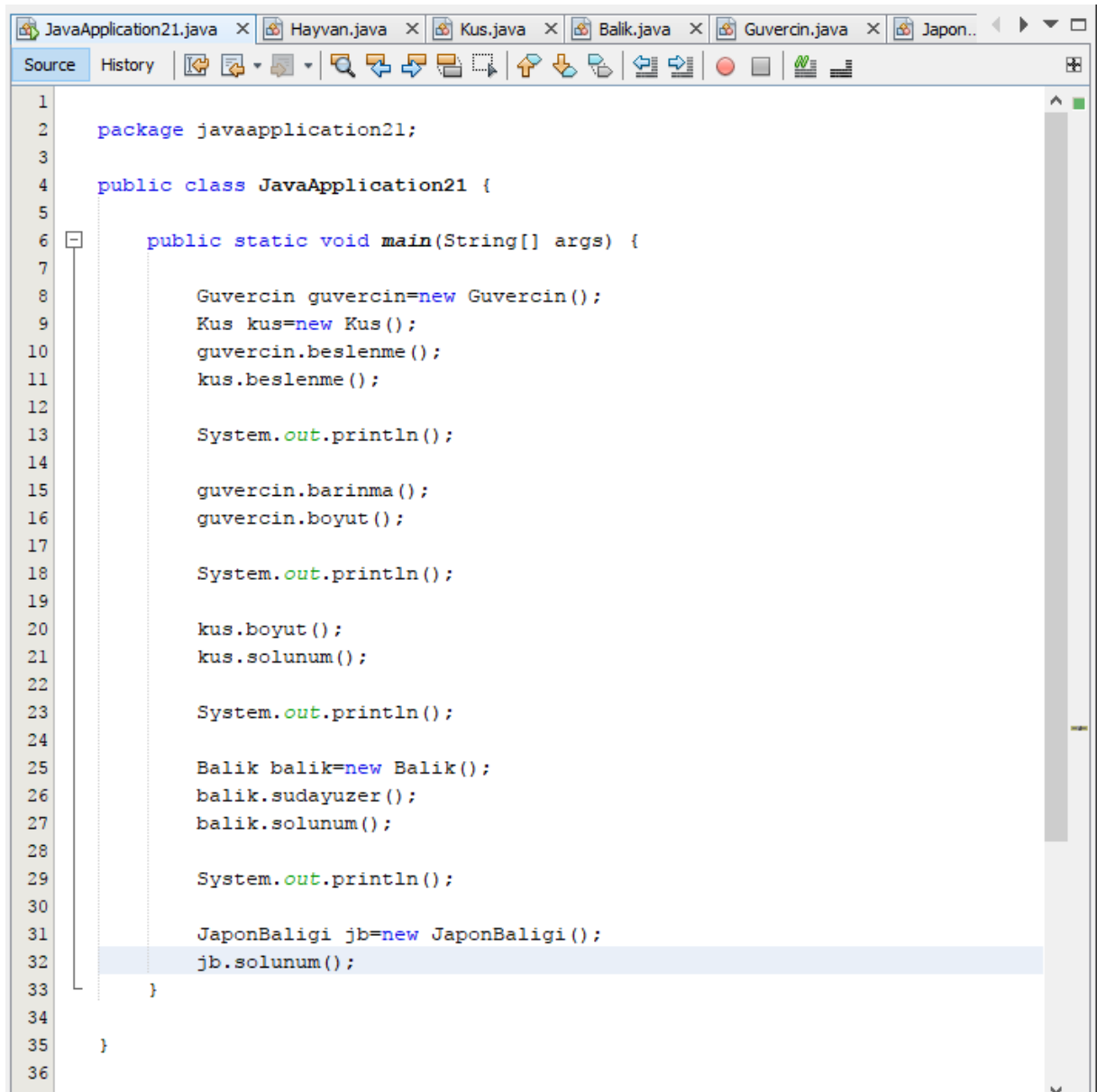

Soru 5.10) Kalıtım ve kompozisyon arasındaki en temel fark nedir?

Kompozisyon bir sınıfı başka sınıfın içinde değişken olarak kullanmaktır.

Eğer sınıfın başına final yazarsak o sınıftan yeni bir sınıf türetemeyiz. Başkası bizim sınıfımızı kalıtım yoluyla kullanmaması için final kullanabiliriz. Ama kompozisyon ile bizim sınıfımızı başkası kullanabilir.

Kalıtım örneği;

Main yordamı :



```
1 package javaapplication21;
2
3
4 public class JavaApplication21 {
5
6     public static void main(String[] args) {
7
8         Guvercin guvercin=new Guvercin();
9         Kus kus=new Kus();
10        guvercin.beslenme();
11        kus.beslenme();
12
13        System.out.println();
14
15        guvercin.barinma();
16        guvercin.boyut();
17
18        System.out.println();
19
20        kus.boyut();
21        kus.solunum();
22
23        System.out.println();
24
25        Balik balik=new Balik();
26        balik.sudayuzer();
27        balik.solunum();
28
29        System.out.println();
30
31        JaponBaligi jb=new JaponBaligi();
32        jb.solunum();
33    }
34
35 }
36
```

Hayvan sınıfı:

```
JavaApplication21.java x Hayvan.java x Kus.java x Balik.java x Guvercin.java x Japon..
Source History
1
2 package javaapplication21;
3
4 public class Hayvan {
5     public void beslenme() {
6         System.out.println("Hayvanlar beslenir.");
7     }
8
9     public void barinma() {
10        System.out.println("Hayvanlar barınır.");
11    }
12
13    public void diski() {
14        System.out.println("Hayvanlar diski bırakır.");
15    }
16
17    public void boyut() {
18        System.out.println("Hayvanların boyutları vardır.");
19    }
20
21    public void solunum() {
22        System.out.println("Hayvanlar solunum yapar.");
23    }
24
25 }
```

Kus sınıfı:

```
JavaApplication21.java x Hayvan.java x Kus.java x Balik.java x Guvercin.java x JaponBaligi.java...
Source History
1 /**
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package javaapplication21;
7
8 /**
9  *
10 * @author f
11 */
12 public class Kus extends Hayvan{
13     public void ucma() {
14         System.out.println("Kuslar ucar.");
15     }
16
17     @Override
18     public void barinma() {
19         System.out.println("Kuslar samanlardan yaptıkları yuvalarda yasarlar.");
20     }
21
22 }
23
```

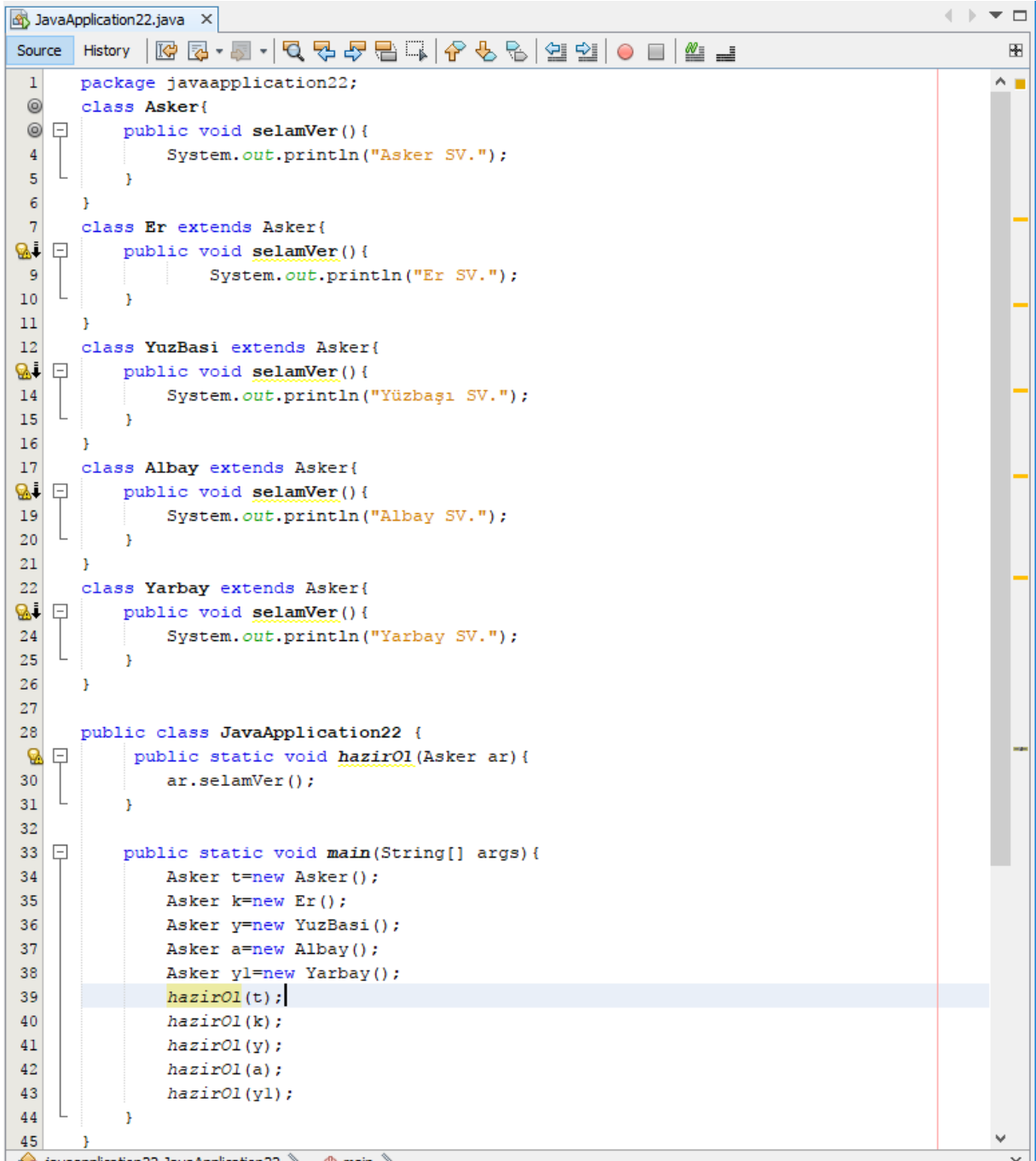
Güvercin sınıfı :

```
JavaApplication21.java x Hayvan.java x Kus.java x Balik.java x Guvercin.java x JaponBaligi.java...
Source History
1 package javaapplication21;
2
3 public class Guvercin extends Kus {
4     @Override
5     public void beslenme() {
6         System.out.println("Guvercin bugday yer.");
7     }
8
9     @Override
10    public void boyut() {
11        System.out.println("Guvercinlerin boyutları 20-30 arasındır.");
12    }
13 }
14 }
15
```

Japon Balığı sınıfı:

```
...ava Hayvan.java x Kus.java x Balik.java x Guvercin.java x JaponBaligi.java x
Source History
1
2 package javaapplication21;
3
4 public class JaponBaligi extends Balik {
5     @Override
6     public void solunum() {
7         super.solunum();
8     }
9 }
10
```

Soru 6.1) Polimorfizm sayesinde genel bir terimden alt terimleri üretebiliyoruz ve genel terimi tip değişkeni gibi kullanabiliyoruz.



```
1 package javaapplication22;
2 class Asker{
3     public void selamVer(){
4         System.out.println("Asker SV.");
5     }
6 }
7 class Er extends Asker{
8     public void selamVer(){
9         System.out.println("Er SV.");
10    }
11 }
12 class YuzBasi extends Asker{
13     public void selamVer(){
14         System.out.println("Yüzbaşı SV.");
15    }
16 }
17 class Albay extends Asker{
18     public void selamVer(){
19         System.out.println("Albay SV.");
20    }
21 }
22 class Yarbay extends Asker{
23     public void selamVer(){
24         System.out.println("Yarbay SV.");
25    }
26 }
27
28 public class JavaApplication22 {
29     public static void hazirOl(Asker ar) {
30         ar.selamVer();
31     }
32
33     public static void main(String[] args) {
34         Asker t=new Asker();
35         Asker k=new Er();
36         Asker y=new YuzBasi();
37         Asker a=new Albay();
38         Asker yl=new Yarbay();
39         hazirOl(t);
40         hazirOl(k);
41         hazirOl(y);
42         hazirOl(a);
43         hazirOl(yl);
44     }
45 }
```

Soru 6.10) Abstract sınıflar, yani soyut sınıflardır. Soyut sınıfların ortak özelliklerini kullanabilmekteyiz. Soyut sınıflar kendisinden türeyen sınıflardır. Soyut sınıflardan nesne oluşturulmaz. Bunun yerine extends edilerek yeni sınıflara o özelliği kullanarak soyut metotlar üretilir.

Bazı durumlarda, yapılacak işlere uyan somut bir üst sınıf ve ona ait somut metotlar tanımlamak mümkün olmayabilir. Böyle durumlarda, soyut bir üst-sınıf ve ona ait soyut metotlar tanımlamak sorunu kolayca çözebilir.

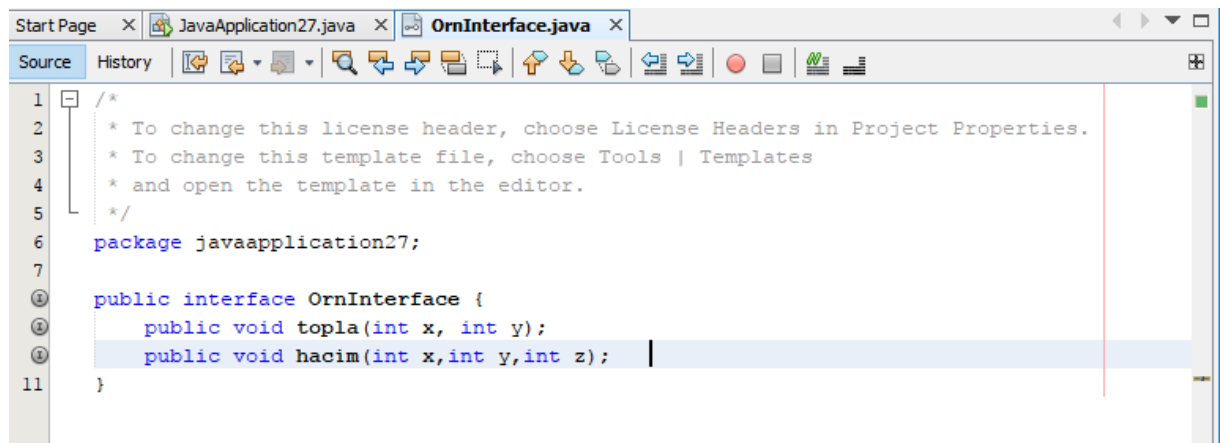
Soyut metot (Abstract class) adı ve parametreleri olduğu halde gövdesi olmayan bir metottur. Dolayısıyla belirli bir iş yapmaz. O, alt-sınıflarda örtülür (overriding).

Soru 7.3) Arayüzlerin, otomatik olarak public erişim belirleyicisine sahip olduklarını ispatlayan uygulama yazınız.

Java' da arayüz soyut sınıfların yerine kullanılır. Ama soyut sınıftan farklı ve daha kullanışlıdır. Arayüz kullanarak, bir sınıfın neler yapacağını belirlerken, onları nasıl yapacağını gizleyebiliriz. Arayüzün yapısı sınıfın yapısına benzese de aralarında önemli farklar vardır.

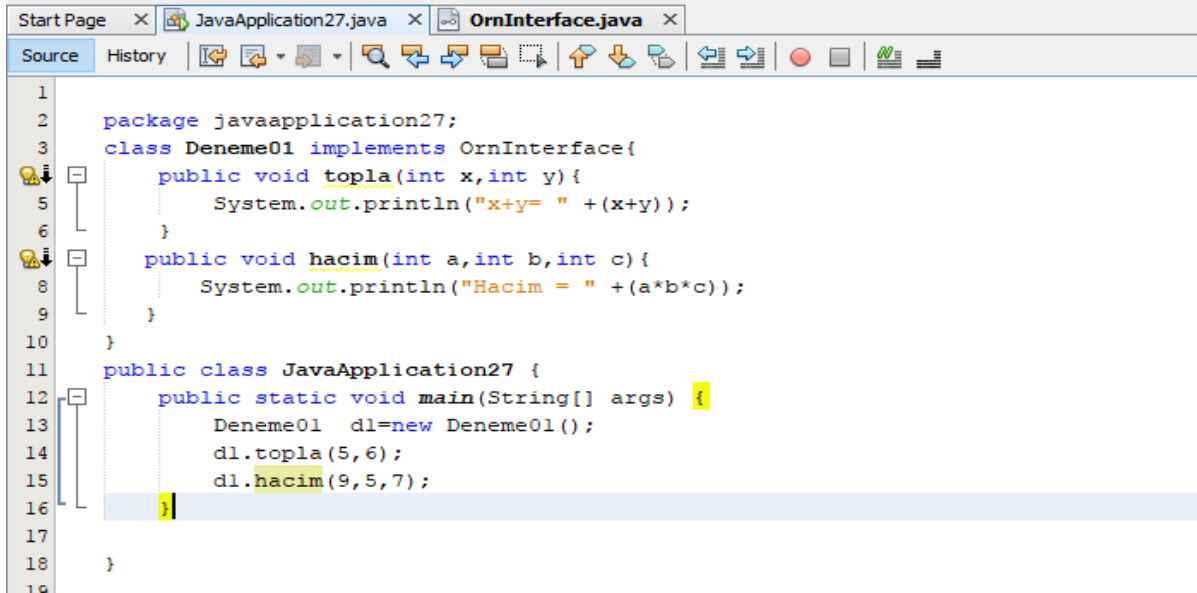
- Arayüz, interface anahtar sözcüğü ile tanımlanır. Arayüz abstract metotlar içerir.
- Arayüz, anlık(instance) değişkenler içeremez. Ancak belirtkeleri konmamış olsalar bile, arayüz içindeki değişkenler final ve static ve olur. Bu demektir ki arayüzde tanımlanan değişkenler, onu çağıran sınıflar tarafından değiştirilemez.
- Arayüz, yalnızca public ve ön tanımlı (default) erişim belirtkisi alabilir başka erişim belirtkisi alamaz.
- Public damgalı arayüz public damgalı class gibidir. Her kod ona erişebilir.
- Erişim damgasız arayüz, erişim damgasız class gibidir. Bu durumda, arayüze, ait olduğu paket içindeki bütün kodlar ona erişebilir. Paket dışındaki kodlar erişemez.
- Arayüzler, public erişim belirtkisi ile nitelenmişse, içindeki bütün metotlar ve değişkenler otomatik olarak public olur.
- Bir sınıf birden çok arayüze çağrılabilir.
- Aynı arayüzü birden çok sınıf çağırabilir.
- Sınıftaki metotlar tam olarak tanımlıdır, ama arayüzde metotların gövdesi yoktur. Onlar abstract metotlardır. Metodun tipi, adı, parametreleri vardır, ama gövde tanımı yoktur; yani yaptığı iş belirli değildir. Metotların gövdesi, o arayüze çağırılan sınıf içinde yapılır. Böylece bir metot, farklı sınıflarda farklı tanımlanabilir. Bu özellik, Java' da polimorfizmi olanaklı kılan önemli bir niteliktir.

Arayüz örneği ;



```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package javaapplication27;
7
8  public interface OrnInterface {
9      public void topla(int x, int y);
10     public void hacim(int x,int y,int z);
11 }
```

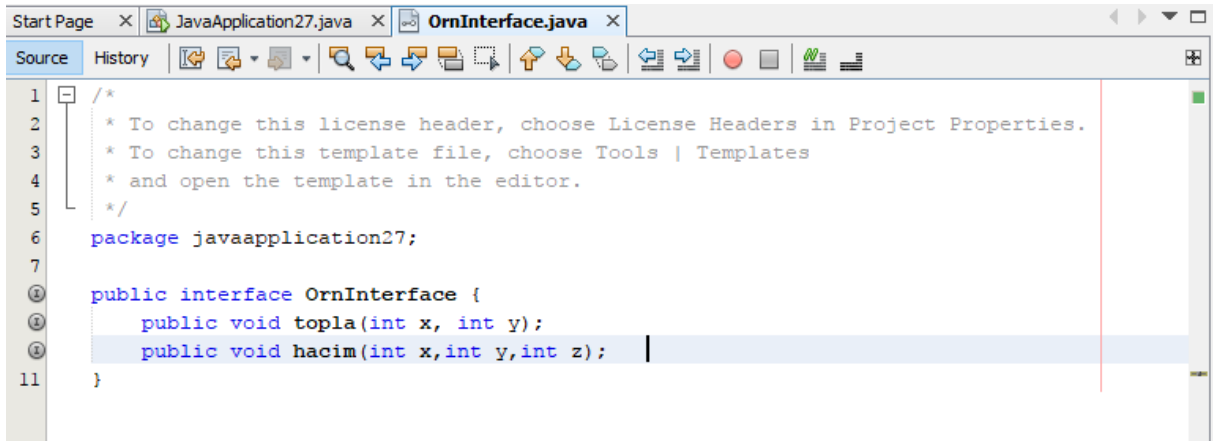
Main yordamında kullanımı ;



```
1
2 package javaapplication27;
3 class Deneme01 implements OrnInterface{
4     public void toplama(int x,int y){
5         System.out.println("x+y= " +(x+y));
6     }
7     public void hacim(int a,int b,int c){
8         System.out.println("Hacim = " +(a*b*c));
9     }
10 }
11 public class JavaApplication27 {
12     public static void main(String[] args) {
13         Deneme01 d1=new Deneme01();
14         d1.toplama(5,6);
15         d1.hacim(9,5,7);
16     }
17 }
18
19
```

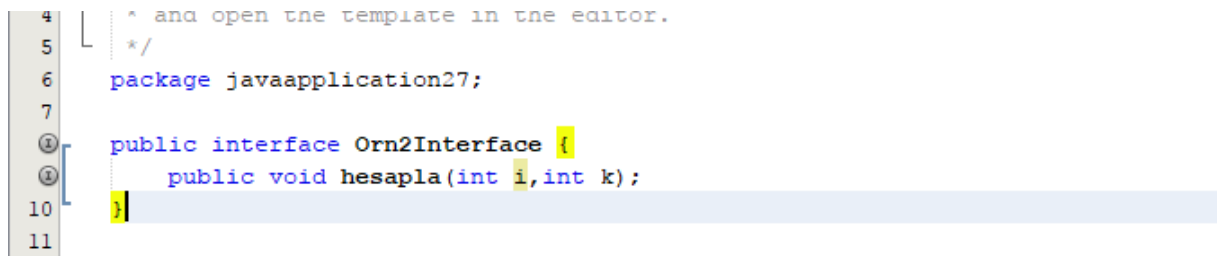
Soru 7.1) Bir sınıfın birden çok arayüze erişebildiğini gösteren bir uygulama yazınız?

İlk Arayüz;



```
1 /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package javaapplication27;
7
8 public interface OrnInterface {
9     public void toplama(int x, int y);
10    public void hacim(int x,int y,int z);
11 }
```

İkinci Arayüz;



```
4 /* and open the template in the editor.
5  */
6 package javaapplication27;
7
8 public interface Orn2Interface {
9     public void hesapla(int i,int k);
10 }
11
```

Main yordamı ;

```
Start Page x JavaApplication27.java x OrnInterface.java x Orn2Interface.java x
Source History | [Icons]
1
2 package javaapplication27;
3 class Deneme01 implements OrnInterface,Orn2Interface{
4     public void toplama(int x,int y){
5         System.out.println("x+y= " +(x+y));
6     }
7     public void hacim(int a,int b,int c){
8         System.out.println("Hacim = " +(a*b*c));
9     }
10    public void hesapla(int k,int i){
11        System.out.println("Toplam fiyat= " +(k*i));
12    }
13
14    public class JavaApplication27 {
15        public static void main(String[] args) {
16            Deneme01 dl=new Deneme01();
17            dl.toplama(5,6);
18            dl.hacim(9,5,7);
19            dl.hesapla(5,20);
20        }
21    }
```

Soru 8.2) Java dilinde var olan RuntimeException istisna tipleri bir uygulama üzerinde açıklayınız.

```
package javaapplication28;
```

```
/**
 *
 * @author f
 */
class RuntimeException extends Exception {

    public RuntimeException(String demo) {
        System.out.println("Yakalandı.. " +demo );
    }

}
```

Main yordamı ;

```
Start Page x JavaApplication28.java x RunTimeException.java x
Source History
1 package javaapplication28;
2 import java.util.logging.Level;
3 import java.util.logging.Logger;
4 class OrnSinif{
5     static void progA(){
6         try{
7             System.out.println("İcerde progA");
8             throw new RuntimeException("demo");
9         }
10        catch (RuntimeException ex) {
11        }
12        finally{
13            System.out.println("ProgA bitiş.");
14        }
15    }
16    static void progB(){
17        try{
18            System.out.println("İcerde progB");
19            return;
20        }
21        finally{
22            System.out.println("ProgB bitiş.");
23        }
24    }
25    static void progC(){
26        try{
27            System.out.println("ProgC icerde");
28        }
29        finally{
30            System.out.println("ProgC bitis.");
31        }
32    }
33 }
34 class JavaApplication28 {
35     public static void main(String[] args) {
36         OrnSinif x=new OrnSinif();
37         try{
38             x.progA();
39         }
40         catch(Exception e){
41             System.out.println("Exception yakalandi. ");
42         }
43         x.progB();
44         x.progC();
45     }
46 }
47
```

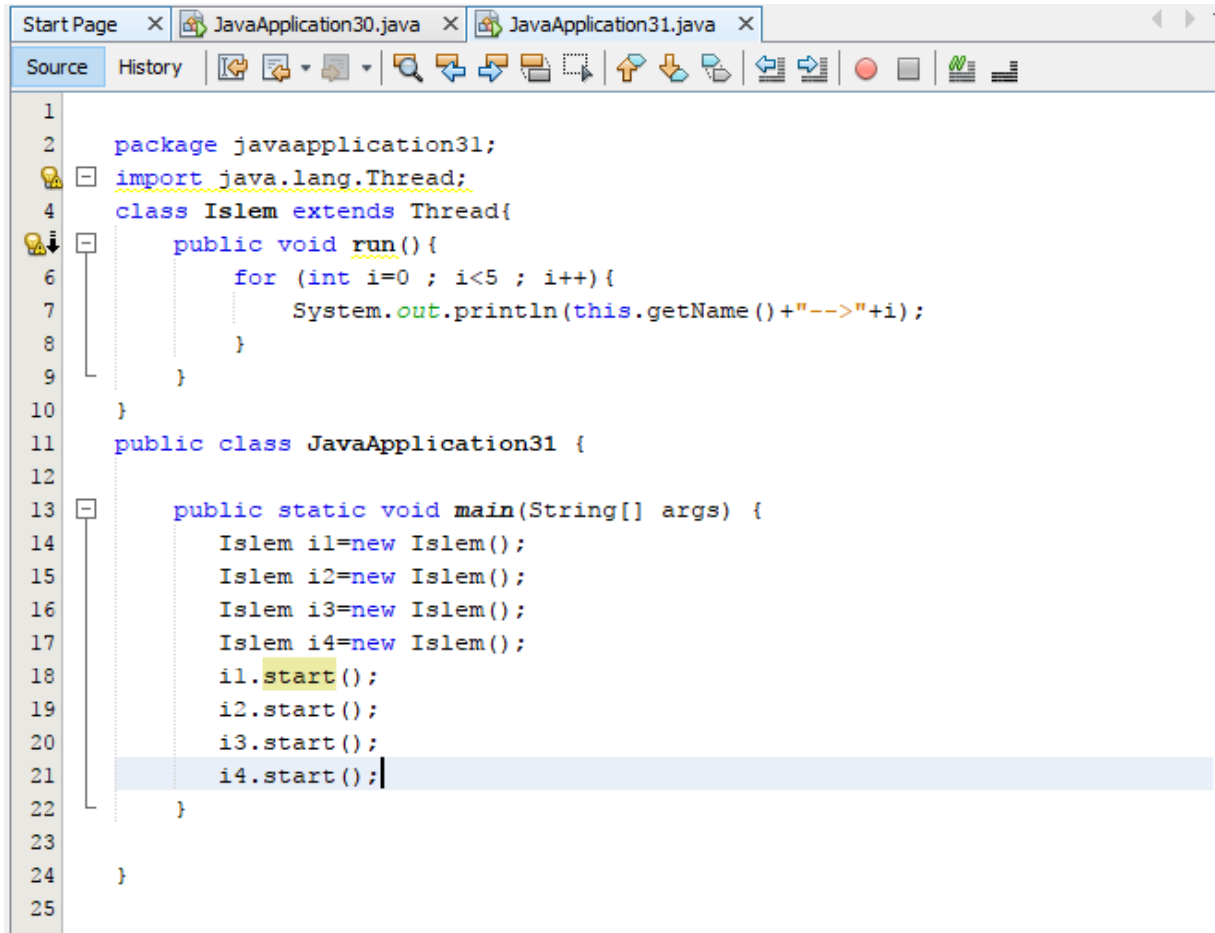

Soru 8.1) Java programlama dilinde dizilere erişimin her zaman güvenli olduğunu gösteren bir uygulama yazınız.

```
Start Page x JavaApplication30.java x
Source History
1  /*
2  * To change this license header, choose License Headers in Project Properties
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package javaapplication30;
7
8  /**
9   *
10  * @author f
11  */
12  public class JavaApplication30 {
13
14
15     public static void main(String[] args) {
16         double[] d = {2.1, 3.4, 4.6, 1.1, 0.11};
17         String[] s = {"defter", "kalem", "sarman", "tekir", "boncuk"};
18         for (int i = 0; i < d.length; i++) {
19             System.out.println("d[" + i + "] = " + d[i]);
20         }
21         System.out.println("-----");
22
23         for (int x = 0; x < s.length; x++) {
24             System.out.println("s[" + x + "] = " + s[x]);
25         }
26     }
27 }
28
29
```

Soru 11.1) CPU' nun etkin biçimde kullanılması için iş parçacıklarının yararları nelerdir ?

- Bağımsız işlemlerle karşılaştırıldığında iş parçacıkları daha hafiftir.
- İş parçacıkları aynı adres alanını paylaştıkları için veri ve kodları paylaşabilir.
- Context switching (içerik değiştirme) iş parçacıklarında işlemlere göre daha az pahalıdır.
- İş parçacıkları arası haberleşme işlemler arası haberleşmeye göre daha ucuzdur.
- İş parçacıkları farklı görevlerin aynı zaman aralığında gerçekleştirilmesine olanak sağlarlar.

Soru 11.2) Thread sınıfından türeyen bir sınıfın run() yordamıyla ekrana iş parçacığının ismini yazdıran bir yapı oluşturun. Başka bir sınıfın içerisinde, bu sınıfa ait 4 adet nesne oluşturup, bu nesnelerin start() yordamlarını çağıran uygulama yazınız.



```
1
2 package javaapplication31;
3 import java.lang.Thread;
4 class Islem extends Thread{
5     public void run(){
6         for (int i=0 ; i<5 ; i++){
7             System.out.println(this.getName()+"-->" +i);
8         }
9     }
10 }
11 public class JavaApplication31 {
12
13     public static void main(String[] args) {
14         Islem i1=new Islem();
15         Islem i2=new Islem();
16         Islem i3=new Islem();
17         Islem i4=new Islem();
18         i1.start();
19         i2.start();
20         i3.start();
21         i4.start();
22     }
23
24 }
25
```