

ALGORİTMA VE PROGRAMLAMA I

HAFTA 10 METOTLAR I

Ders İçeriği: Hafta 10

- Giriş
- Metot nedir?
- Metot Tanımlama
- Metot Çağrısı
- Örnekler

Giriş

- Her bilgisayar dili değişken ve metod-fonksiyon kavramlarına sahiptir.
- Değişkenler, bilgisayar programının işleyeceği ham maddedir.
- Bu ham maddelerin işlenip ürün haline getirilmesi için kullanılan araçlar ise metotlardır.
- Nesne yönelimli programlamada:
 - Her şey sınıflar (class) içinde tanımlanır.
 - Sınıfa ait değişkenler (özellik-property) o sınıfın özelliklerini ortaya koyar.
 - Sınıf içinde tanımlı metotlar (fonksiyon) da o sınıfın tavır, davranış ve eylemlerini ortaya koyar.

Metot Nedir?

- Belirli bir işlem gerçekleştirmek için tasarlanmış ve tekrar tekrar kullanılabilen kod parçalarıdır.
- Tasarım ve işlevlerine göre
 - Parametre alabilirler
 - Bir sonuç değeri döndürebilirler
- Java dilinde her ifade bir metot içerisinde icra edilir.
- Her program bir başlangıç noktasına ihtiyaç duyar ve Java programlarında başlangıç noktası Main() fonksiyonudur.

Metot Tanımlama

- Java dilinde metotların tanımlanma konumu sınıf (class) veya yapılardır (struct).
- Metotların bir ismi vardır.
- Metotların bir erişim seviyesi vardır.
- Metotları dönüş değeri olabilir, eğer bir dönüş değeri yoksa void anahtar kelimesi kullanılmalıdır.
- Metotlar parametre alabilirler, eğer birden fazla parametre varsa virgül (,) ile parametreler ayrılır.

Metot Tanımlama

```
erisim_seviyesi donus_turu metot_ismi (parametre1, parametre2)  
  
{  
  
    // işlemler  
  
    return dongus_degeri;  
  
}
```

Metot Tanımlama

➤ Main() Metodu

- Programın bütün işlevini yaptıran ana metottur.
- Programın kullanacağı bütün deyimleri tek başına içerebileceği gibi, modüler yapıda yazılan programlarda farklı sınıflarda tanımlanan metot ve değişkenleri çağırabilir.
- Her programda ana metot olma işlevini üstlenen ve sadece bir tane tanımlanan Main() metodu vardır.

Metot Tanımlama

➤ Main() Metodu

```
public static void main(String[] args) {
```

```
    //fonksiyon işlevleri
```

```
}
```

- Değer kümesi void yani bir değer almaz ve yalnızca başka bir metodu çağırarak onun işlevini yapmasını sağlar.
- Yani bu metot geriye herhangi bir değer döndürmez ve return ifadesi kullanılmaz.
- Main metodu tanımlanırken static anahtar kelimesinin kullanımı zorunludur.

Metot Tanımlama

➤ Static Anahtar Kelimesi

- Bir sınıfta tanımlı değişken veya metodu static nitelemenin nesne yönelimli programlamada özel bir işlevi vardır.
- Nesne yönelimli programlamada genel kural:
 - Bir sınıf içerisinde tanımlı değişken ve metotlar, ancak o sınıfa ait bir nesne yaratıldıktan sonra o nesne içinde işlevsellik kazanırlar.
- Bu kuralın istisnai hali:
 - Değişkenin veya metodun önüne static nitelemesi konularak yapılır.
- Static sözcüğü ile nitelenen değişken veya metot, ait olduğu bir nesne yaratılmadan da kullanılabilir.

Metot Tanımlama

➤ Metotların Dönüş Türü

- Eğer metot bir sonuç değeri döndürmüyorsa void anahtar kelimesi kullanılır.
- Fonksiyon adının soluna yazılan veri tipi bilgisi, fonksiyonun değer bölgesidir.
- Fonksiyonun alacağı değer return anahtar sözcüğü ile belirtilir.

```
public static int karesiniAl(int sayi) {  
    int karesi=sayi*sayi;  
    return karesi;  
}
```

Metot Tanımlama

➤ Parametreler ve Yerel Değişkenler

- Fonksiyon başlığında fonksiyon adının sağına yazılan parantez içi değişkenlere fonksiyon parametresi denir.
- Fonksiyonun { } gövdesinde tanımlanan değişkenlere de fonksiyonun yerel değişkeni adı verilir.

```
public static int toplam(int sayi1, int sayi2) {  
    int toplam=sayi1+sayi2;  
    return toplam;  
}
```

- Metot tanımlarındaki parametreler aksi belirtilmediği takdirde zorunludur.

Metot Çağrısı

- Metotları çağırırken aldıkları imzalarına uygun bir şekilde çağırılması gerekmektedir.
- Metotlara aldıkları parametrelerin veri türüne uygun parametreler gönderilmelidir.
- Parametreler değer veya referans olarak metotlara gönderilebilir.
- Değer türünde gönderilen parametrelerin bir kopyası oluşturulur.
- Referans türünde ise değişkenin referansı gönderilir.

Metot Çağrısı

```
public static void main(String[] args) {
```

```
    int sonuc = toplam(3, 6);
```

```
}
```

1 usage

```
public static int toplam(int sayi1, int sayi2) {
```

```
    int toplam = sayi1 + sayi2;
```

```
    return toplam;
```

```
}
```

Metot Çağrısı

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println(kareal(sayi: 5));  
        int x=6;  
        System.out.println(kareal(x));  
        System.out.println(kareal(x)*2);  
        System.out.println("x= "+x);  
    }  
    3 usages  
    public static int kareal(int sayi) {  
        return sayi*sayi;  
    }  
}
```

25
36
72
x= 6

Örnek 1

➤ Aşağıdaki kod bloğundaki hatayı bulalım.

```
public class Main {  
    1 usage      n değişkeninin main içinde kullanılabilmesi için static  
    int n;      tanımlı olması gerekir!  
    public static void main(String[] args) {  
        n=kareal(sayi: 9);  
    }  
    1 usage  
    public static int kareal(int sayi) {  
        int y;  
        y=sayi*sayi;  
        return y;  
    }  
}
```

Örnek 2

- Dört işlem yapan basit bir hesap makinesi tasarlayalım. Dört işlemin her biri için ayrı metot tanımlayalım.

Örnek 2

- Dört işlem yapan basit bir hesap makinesi tasarlayalım. Dört işlemin her biri için ayrı metot tanımlayalım.

```
public static float toplama(float sayi1, float sayi2){  
    return sayi1+sayi2;  
}
```

1 usage

```
public static float cikarma(float sayi1, float sayi2){  
    return sayi1-sayi2;  
}
```

1 usage

```
public static float carpma(float sayi1, float sayi2){  
    return sayi1*sayi2;  
}
```

1 usage

```
public static float bolme(float sayi1, float sayi2){  
    return sayi1/sayi2;  
}
```

Örnek 2

- Dört işlem yapan basit bir hesap makinesi tasarlayalım. Dört işlemin her biri için ayrı metot tanımlayalım.

```
public static void main(String[] args) {  
    float sayi1=49, sayi2=7;  
    System.out.println(toplama(sayi1, sayi2));  
    System.out.println(cikarma(sayi1, sayi2));  
    System.out.println(carpma(sayi1, sayi2));  
    System.out.println(bolme(sayi1, sayi2));  
}
```

Örnek 3

- Klavyeden girilen yaş bilgisi 18'den büyük ise girişi onaylayan değilse onaylamayan bir fonksiyon yazalım.

Örnek 3

- Klavyeden girilen yaş bilgisi 18'den büyük ise girişi onaylayan değilse onaylamayan bir fonksiyon yazalım.
- Önce yaş bilgisini parametre olarak alan metodu oluşturalım.

```
public static void YasKontrol(int yas){  
    if (yas>=18)  
        System.out.println("Giriş onaylandı.");  
    else  
        System.out.println("Giriş Onaylanmadı!!!");  
}
```

Örnek 3

- Klavyeden girilen yaş bilgisi 18'den büyük ise girişi onaylayan değilse onaylamayan bir fonksiyon yazalım.
- Main metodunda kullanıcıdan bilgi alma işlevini yerine getirelim.

```
public static void main(String[] args) {  
    Scanner klavye=new Scanner(System.in);  
    System.out.println("Yaşınızı giriniz:");  
    int yas= klavye.nextInt();  
    YasKontrol(yas);  
}
```

Örnek 4

- 5 farklı öğrenci için klavyeden vize ve final notlarını alarak dönem sonu notunu hesaplayıp harf notlarını ekrana yazdıran metodu oluşturalım.

Örnek 4

- Öncelikle öğrenciden alınan vize ve finali harf notuna dönüştüren metodu yazalım.

```
public static String NotHesaplama(int vize, int fiinal){  
    double ortalama=vize*0.4 + fiinal*0.6;  
    String harfNotu;  
    if (ortalama >= 90)  
        harfNotu="AA";  
    else if (ortalama >= 80)  
        harfNotu="BB";  
    else if (ortalama >= 70)  
        harfNotu="CC";  
    else if (ortalama >= 50)  
        harfNotu="DD";  
    else  
        harfNotu="FF";  
    return harfNotu;  
}
```

Örnek 4

- Main metodunda 5 öğrenci için klavyeden bilgileri alıp hesaplanan harf notunu ekrana yazdıralım.

```
public static void main(String[] args) {  
    Scanner klavye=new Scanner(System.in);  
    int vize, fiinal;  
    for (int i=1;i<=5;i++){  
        System.out.println("Vize Notunu Giriniz:");  
        vize= klavye.nextInt();  
        System.out.println("Final Notunu Giriniz:");  
        fiinal= klavye.nextInt();  
        System.out.println(i+". Öğrencinin Harf Notu= "+NotHesaplama(vize, fiinal));  
    }  
}
```

Örnek 5

- Klavyeden girilen bir sayının asal olup olmadığını metot ile kontrol ederek sonuç döndüren programı yazalım.

Örnek 5

- Klavyeden girilen bir sayının asal olup olmadığını metot ile kontrol ederek sonuç döndüren programı yazalım.
- Öncelikle sayının asal olmasını kontrol eden ve boolean (true/false) değer döndüren bir metot oluşturalım.

```
public static boolean AsalMi(int sayi){  
    boolean sonuc=true;  
    for(int i=2;i<sayi;i++){  
        if(sayi%i==0)  
            sonuc=false;  
    }  
    return sonuc;  
}
```

Örnek 5

- Klavyeden girilen bir sayının asal olup olmadığını metot ile kontrol ederek sonuç döndüren programı yazalım.
- Metodun dönüş değerine göre asal olma durumunu ekrana main metot içerisinde yazdıralım.

```
public static void main(String[] args) {  
    Scanner klavye = new Scanner(System.in);  
    System.out.println("Bir Sayı Giriniz:");  
    int sayi= klavye.nextInt();  
    if (AsalMi(sayi))  
        System.out.println("Sayı Asaldır.");  
    else  
        System.out.println("Sayı Asal Değildir.");  
}
```