

En Basit Java Programı

HelloWorld.java isimli dosyada

```

public class HelloWorld
{
    public static void main(String[] args)
    {
        // Konsola "Hello, World" yazdırır
        System.out.print("Hello, World");
    }
}

```

class ismi
main() metodu
ifadeler
metod gövdesi

Değişken Tanımlama ve Atama

değişken deklarasyonu

```

int a, b;
a = 1234;
b = 99;
int c = a + b;

```

değişken adı
değer
değer atama ifadesi
aynı satırda ilklendirme (deklarasyon ve atama bir arada)

byte → short → int → long
float → double

Metod Tanımı

metod imzası
static: nesneye değil sınıfa aitlik
dönüş tipi
metod ismi
parametre tipi
parametre değişkeni

```

public static double harmonic (int n)
{
    double sum = 0.0;
    for (int i = 1; i <= n; i++);
        sum += 1.0/i;
    return sum;
}

```

yerel değişken
metod gövdesi
metoddan değer dönme ifadesi

Örnek çağırım: `double deger = harmonic(5);`

void	denotes a method that returns no value	public void print() { ... }
------	--	-----------------------------

Veri Tipleri

veri tipi	alabileceği değerler	ortak operatörler	örnek değerler
int	Tam sayılar	+ - * / %	99 12 2147483647
double	Ondalık sayılar	+ - * /	3.14 2.5 6.022e23
boolean	Mantıksal ifadeler	&& !	true false
char	Tek bir karakter		'A' '1' '%' '\n'
String	Sıralı karakterler	+	"AB" "Hello" "2.5"
long	Büyük tam sayılar	+ - * / %	2147483647147L

Tip Dönüşümleri

ifade (expression)	ifade tipi	ifade değeri
(1 + 2 + 3 + 4) / 4.0	double	2.5
Math.sqrt(4)	double	2.0
"1234" + 99	String	"123499"
11 * 0.25	double	2.75
(int) 11 * 0.25	double	2.75
11 * (int) 0.25	int	0
(int) (11 * 0.25)	int	2
(int) 2.71828	int	2
Math.round(2.71828)	long	3
(int) Math.round(2.71828)	int	3
Integer.parseInt("1234")	int	1234

Örnek: `String mesaj = "1" + 7; // "17"`

Konsola Yazdırma

`System.out.print(s)` konsola s değişkeninin değerini bas
`System.out.println(s)` konsola s değişkeninin değerini bas ve alt satıra geç
`System.out.println()` konsola boş bir satır bas, alt satıra geç

Switch Kontrol İfadesi

```

switch (day) {
    case 0: System.out.println("Sun"); break;
    case 1: System.out.println("Mon"); break;
    case 2: System.out.println("Tue"); break;
    case 3: System.out.println("Wed"); break;
    case 4: System.out.println("Thu"); break;
    case 5: System.out.println("Fri"); break;
    case 6: System.out.println("Sat"); break;
}

```

If Kontrol İfadesi

mutlak değer hesabı	if (x < 0) x = -x;
küçük değer x'e büyük değer y'ye koyulur	<pre> if (x > y) { int t = x; x = y; y = t; } </pre>
x ve y'nin büyük olanı max'a atanır	<pre> if (x > y) max = x; else max = y; </pre>
0'a bölme hatası kontrolü	<pre> if (den == 0) System.out.println("Division by zero"); else System.out.println("Quotient = " + num/den); </pre>

List

<pre> List<String> strs = new ArrayList<>(); strs.add("Hello"); strs.add("World!"); for (String str : strs) System.out.println(str); </pre>	<pre> strs.addAll(strColl); strs.addAll(Arrays.asList(args)); strs.removeAll(coll); strs.insert(i, "Hello"); str = strs.remove(i); </pre>
---	---

İç içe geçmiş if-else ifadeleri

```

if (income < 0) rate = 0.00;
else if (income < 8925) rate = 0.10;
else if (income < 36250) rate = 0.15;
else if (income < 87850) rate = 0.23;
else if (income < 183250) rate = 0.28;
else if (income < 398350) rate = 0.33;
else if (income < 400000) rate = 0.35;
else rate = 0.396;

```

Mantıksal Kontroller

op	true	false
==	2 == 2	2 == 3
!=	3 != 2	2 != 2
<	2 < 13	2 < 2
<=	2 <= 2	3 <= 2
>	13 > 2	2 > 13
>=	3 >= 2	2 >= 3

Try-catch

```

try {
    try {
        fred.print(out);
    } catch (PrinterException ex) {
        ex.printStackTrace();
    }
} finally {
    out.close();
}

```

Döngüler

ilk deger ayrı bir ifade olarak ayrı değişkende

```
int power = 1;
while (power <= n/2)
{
    power = 2*power;
}
```

Tek ifade varsa blok opsiyonel, aksi halde zorunlu

döngü devam koşulu

döngü gövdesi (bir dizi ifade)

yardımcı değişken döngü dışında tanımlanır

```
int power = 1;
for (int i = 0; i <= n; i++)
{
    System.out.println(i + " " + power);
    power = 2*power;
}
```

döngü kontrol değişkeni tanımlanır ve ilk deger verilir

döngü devam koşulu

her döngü sonunda çalışan ifade (genelde döngü kontrol değişkeni değeri değiştirilir. ";" ile ayrı birden fazla ifade yazmak mümkün)

döngü gövdesi (bir dizi ifade)

Diziler

```
String[] a = { "Cl", "Di", "Hea", "S" };
```

a
a[0]
a[1]
a[2]
a[3]

Döngü Örnekleri

2 sayısının n'e eşit veya küçük olan en büyük değerini bulma

```
int power = 1;
while (power <= n/2)
    power = 2*power;
System.out.println(power);
```

1'den n'e kadar olan sayıları toplama
(1 + 2 + ... + n)

```
int sum = 0;
for (int i = 1; i <= n; i++)
    sum += i;
System.out.println(sum);
```

1'den n'e kadar olan sayıların çarpımı (n faktoriyel)
(n! = 1 × 2 × ... × n)

```
int product = 1;
for (int i = 1; i <= n; i++)
    product *= i;
System.out.println(product);
```

Tablo halinde bir fonksiyonun değerlerini basma

```
for (int i = 0; i <= n; i++)
    System.out.println(i + " " + 2*i);
```

Sınıf Tanımlama

```
public class Charge
{
    private final double rx, ry;
    private final double q;

    public Charge(double x0, double y0, double q0)
    { rx = x0; ry = y0; q = q0; }

    public double potentialAt(double x, double y)
    {
        double k = 8.99e09;
        double dx = x - rx;
        double dy = y - ry;
        return k * q / Math.sqrt(dx*dx + dy*dy);
    }

    public String toString()
    { return q + " at " + "(" + rx + ", " + ry + ")"; }

    public static void main(String[] args)
    {
        double x = Double.parseDouble(args[0]);
        double y = Double.parseDouble(args[1]);
        Charge c1 = new Charge(0.51, 0.63, 21.3);
        Charge c2 = new Charge(0.13, 0.94, 81.9);
        double v1 = c1.potentialAt(x, y);
        double v2 = c2.potentialAt(x, y);
        StdOut.printf("%.2e\n", (v1 + v2));
    }
}
```

nesne değişkenleri

yapılandırıcı constructor

nesne metodları

programın ilk çalışma noktası (main)

Nesne tanımla ve oluştur

nesne değişkenine erişim

Nesne metodunu tetikleme

nesne değişkenlerine erişim

sınıf ismi

Nesne Kullanımı

Bir nesne değişkeni tanımla (nesne adı)

```
String s;
```

Nesne oluşturmak için yapılandırıcıyı (constructor) tetikle

```
s = new String("Hello, World");
```

```
char c = s.charAt(4);
```

nesneye erişme ve sınıfta tanımlı elemanlara "." ile erişme

nesnenin değerleri üzerinde çalışan bir nesne metodunu tetikleme

Nesne Değişkenleri

```
public class Charge
{
    private final double rx, ry;
    private final double q;
    ...
}
```

nesne değişkenleri tanımla

erişim belirteçleri

- public: dışarıdan erişim
- private: sadece sınıftan erişim
- final: sadece bir kez deger atanabilir

String Tipi

```
String a = new String("now is");
String b = new String("the time");
String c = new String(" the");
```

nesne metodunu çağırma	dönüş tipi	dönüş değeri
a.length()	int	6
a.charAt(4)	char	'i'
a.substring(2, 5)	String	"w i"
b.startsWith("the")	boolean	true
a.indexOf("is")	int	4
a.concat(c)	String	"now is the"
b.replace("t", "T")	String	"The Tim"
a.split(" ")	String[]	{ "now", "is" }
b.equals(c)	boolean	false

Miras

```
class <classname> extends <superclass name>
implements <interface name>
{
    class body;
}
```

// Miras alan sınıf interface'deki ve abstract sınıftaki
// abstract metodları tanımlamak zorundadır

Abstract Class & Method

```
abstract class Writable {
    public abstract void
    write(Writer out);
    public void save(String
    filename) { ... }
}
```

// Hem normal hem de abstract metod
// içerebilir, direkt nesne oluşturulamaz

Interface

```
interface <interface name>
{
    void abc();
    void xyz();
}
```

// Tüm metodlar abstract ve public'dir.
// Direkt nesne oluşturulamaz.