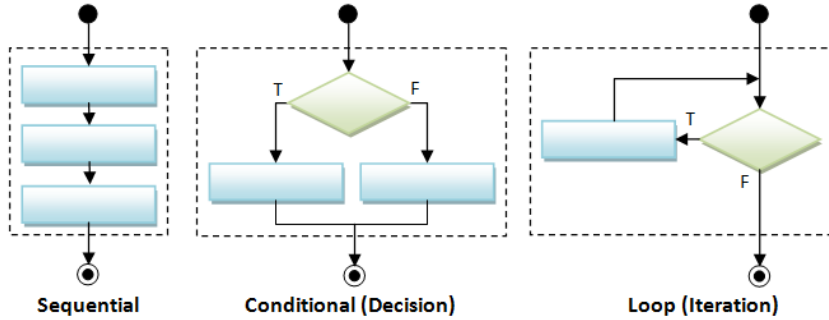


İleri Programlama

Denetim Yapıları ve Döngüler

Hüseyin Ahmetođlu

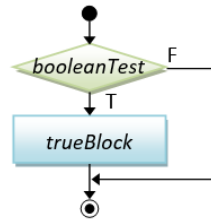
Denetim Yapıları



if-then

Syntax

```
// if-then
if (booleanTest) {
    trueBlock;
}
// next statement
```

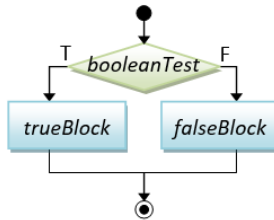


```
int mark = 80;
if (mark >= 80) {
    System.out.println("Well Done!");
    System.out.println("Keep it up!");
}
System.out.println("Life goes on!");

double temperature = 80.1;
if (temperature > 80) {
    System.out.println("Too Hot!");
}
System.out.println("yummy!");
```

if-then-else

```
// if-then-else
if (booleanTest) {
    trueBlock;
} else {
    falseBlock;
}
// next statement
```



```
int mark = 50; // Assume that mark is [0, 100]
if (mark >= 50) { // [50, 100]
    System.out.println("Congratulation!");
    System.out.println("Keep it up!");
} else { // [0, 49]
    System.out.println("Try Harder!");
}
System.out.println("Life goes on!");

double temperature = 80.1;
if (temperature > 80) {
    System.out.println("Too Hot!");
} else {
    System.out.println("Too Cold!");
}
System.out.println("yummy!");
```

Örnek

Örnek 4.1: Dışarıdan girilen sayı negatif ise girilen sayının karesini alan programı yazınız.

Çözüm:

Algoritması:

1. **Başla**
2. Gir bir sayı
3. Eğer sayı negatif ($\text{sayi} < 0$) ise sayının karesini al
4. Yaz kare'yi
5. **Dur**

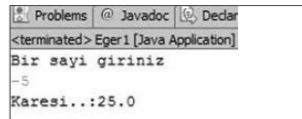
Java programlama dilinde kodlanması:

```
import java.util.Scanner;
class Eger1 {
public static void main(String[] args){
    Scanner tara =new Scanner( System.in );
    double sayi, kare=0;
    System.out.println ("Bir sayi giriniz");

    sayi=tara.nextInt();
    if (sayi<0) // negatifse
    {
        kare=sayi*sayi;
    }
    System.out.println("Karesi..."+kare);
}}

```

Programın ekran çıktısı:



```
Problems @ Javadoc [D] Declar
<terminated> Eger1 [Java Application]
Bir sayi giriniz
-5
Karesi...:25.0

```

Örnek

```
public class test1 {  
    public static void main(String[] args) {  
        int sonuc = 77;  
        if (sonuc >= 80) System.out.print("A");  
        if (sonuc >= 70) System.out.print("B");  
        if (sonuc >= 60) System.out.print("C");  
        else System.out.println("D");  
    }  
}
```

Programın ekran çıktısı: Programa dikkat edersek başlangıçta ‘sonuc’ isimli değişkene ‘77’ değeri aktarılmıştır. ‘77’ sayısı ‘70’ ve ‘60’ dan büyük olduğu için program ekrana “BC” değerini yazacaktır.

Örnek

```
import java.util.Scanner;
class NP {
public static void main(String[] args){
    Scanner tara =new Scanner( System.in );
    System.out.println ("Bir sayı giriniz");
    int sayi=tara.nextInt();
    if (sayi>=0)
        System.out.println("Sayı Pozitifdir");
    else
        System.out.println("Sayı Negatifdir");
}}
```

Programın ekran çıktısı: Sayı değeri olarak, -3 ve +3 girildiğinde programın ekran çıktısı yandaki gibi olur.

```
<terminated> NP [Java Application]
Bir sayı giriniz
-3
Sayı Negatifdir
```

```
<terminated> NP [Java Application]
Bir sayı giriniz
3
Sayı Pozitifdir
```

Örnek 4.6: Dışarıdan girilen sayının tek mi, çift mi olduğuna karar veren programı yazınız.

```
public class TekCift {
    public static void main(String[] args) {
        int sayi = 0;
        Scanner gir = new Scanner(System.in);
        System.out.println("Sayı gir");
        sayi = gir.nextInt();
        if (sayi % 2 == 0) { // Çift mi
            System.out.println("Çift sayı");
        } else {
            System.out.println("Tek sayı");
        }
    }
}
```

Programın ekran çıktısı:

```
<terminated> TekCift [Java Application]
Sayı gir
5
Çift sayı
```

Küme Ayraçları

- Küme ayraçları: Blok içinde yalnızca bir ifade varsa, küme ayraçlarını {} atlayabilirsiniz.

```
// if-then
int absValue = -5;
if (absValue < 0) absValue = -absValue; // Only one statement in the block, can omit { }

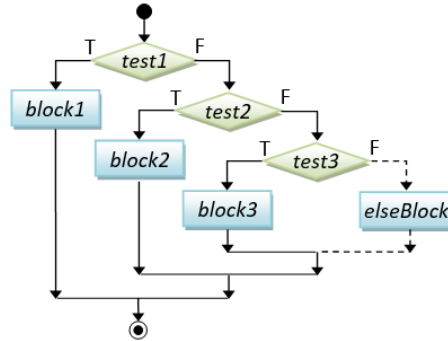
int min = 0, value = -5;
if (value < min) { // More than one statements in the block, need { }
    min = value;
    System.out.println("Found new min");
}

// if-then-else
int mark = 50;
if (mark >= 50)
    System.out.println("PASS"); // Only one statement in the block, can omit { }
else { // More than one statements in the block, need { }
    System.out.println("FAIL");
    System.out.println("Try Harder!");
}

// Harder to read without the braces
int number1 = 8, number2 = 9, absDiff;
if (number1 > number2) absDiff = number1 - number2;
else absDiff = number2 - number1;
```


İç içe - if

```
// nested-if
if (booleanTest1) {
    block1;
} else if (booleanTest2) {
    block2;
} else if (booleanTest3) {
    block3;
} else if (booleanTest4) {
    .....
} else {
    elseBlock;
}
// next statement
```



```
int mark = 62; // Assume that mark is [0, 100]
if (mark >= 80) { // [80, 100]
    System.out.println("A");
} else if (mark >= 65) { // [65, 79]
    System.out.println("B");
} else if (mark >= 50) { // [50, 64]
    System.out.println("C");
} else { // [0, 49]
    System.out.println("F");
}
System.out.println("Life goes on!");

double temperature = 61;
if (temperature > 80) { // > 80
    System.out.println("Too Hot!");
} else if (temperature > 75) { // (75, 80]
    System.out.println("Just right!");
} else { // <= 75
    System.out.println("Too Cold!");
}
System.out.println("yummy!");
```

İç içe if ve Sıralı if

```
// Assume mark is between 0 and 100
// This "sequential-if" works but NOT efficient!
// Try mark = 81, which will run thru ALL the if's.
int mark = 81;
if (mark > 80) { // [81, 100]
    grade = 'A';
}
if (mark > 65 && mark <= 80) { // [66, 80]
    grade = 'B';
}
if (mark >= 50 && mark <= 65) { // [50, 65]
    grade = 'C';
}
if (mark < 50) { // [0, 49]
    grade = 'F';
}
```

```
// This "nested-if" is BETTER
// Try mark = 81, which only run thru only the first if.
int mark = 81;
if (mark > 80) { // [81, 100]
    grade = 'A';
} else if (mark > 65 && mark <= 80) { // [66, 80]
    grade = 'B';
} else if (mark >= 50 && mark <= 65) { // [50, 65]
    grade = 'C';
} else {
    grade = 'F'; // [0, 49]
}
```

```
// This "nested-if" is the BEST with fewer tests
int mark = 81;
if (mark > 80) { // [81, 100]
    grade = 'A';
} else if (mark > 65) { // [66, 80]
    grade = 'B';
} else if (mark >= 50) { // [50, 65]
    grade = 'C';
} else { // [0, 49]
    grade = 'F';
}
```

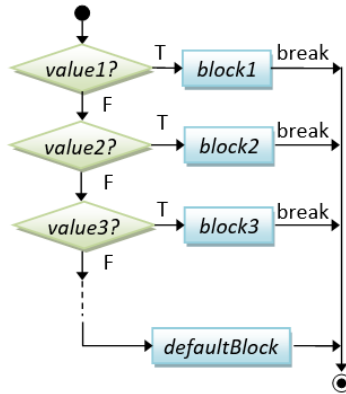
Örnek

```
import java.util.Scanner;
class Yas{
public static void main(String[] args){
Scanner tara =new Scanner( System.in );
int yas;
System.out.println ("Yaşınızı giriniz");
yas=tara.nextInt();
    if (yas >= 0 && yas <= 2)
        System.out.println("Bebek");
    else if (yas > 2 && yas <= 12)
        System.out.println("Çocuk");
    else if (yas > 12 && yas <= 24)
        System.out.println("Yetişkin");
    else if (yas > 24 && yas <= 39)
        System.out.println("Genç");
    else if (yas > 39 && yas <= 54)
        System.out.println("Orta Yaşlı");
    else
        System.out.println("İhtiyar");
}}
```

switch-case-default

```
// switch-case-default
switch (selector) {
  case value1:
    block1; break;
  case value2:
    block2; break;
  case value3:
    block3; break;
  .....
  case valueN:
    blockN; break;
  default: // not the above
    defaultBlock;
}
// next statement

// Selector Types:
// byte, short, int,
// char, String
```



```
// Print number in word
int number = 3;
switch (number) { // "int" selector
  case 1: // "int" value
    System.out.println("ONE"); break;
  case 2:
    System.out.println("TWO"); break;
  case 3:
    System.out.println("THREE"); break;
  default:
    System.err.println("OTHER");
}

// Select arithmetic operation
char operator = '*';
int num1 = 5, num2 = 8, result;
switch (operator) { // "char" selector
  case '+': // "char" value
    result = num1 + num2; break;
  case '-':
    result = num1 - num2; break;
  case '*':
    result = num1 * num2; break;
  case '/':
    result = num1 / num2; break;
  default:
    System.out.println("Unknown operator");
}
```

switch-case-default

- Bir switch-case deyiminde, her bir durum için bir break ifadesi gereklidir. break yoksa, yürütme genellikle bir hata olan bir aşağıdaki durumdan geçer. Ancak, bu özelliği çoklu değer seçiciyi işlemek için kullanabiliriz.

```
// Converting Phone keypad letter to digit
char inChar = 'x';
switch (inChar) {
    case 'a': case 'b': case 'c': // 'a' and 'b' (without break) fall thru 'c'
        System.out.print(2); break;
    case 'd': case 'e': case 'f':
        System.out.print(3); break;
    case 'g': case 'h': case 'i':
        System.out.print(4); break;
    case 'j': case 'k': case 'l':
        System.out.print(5); break;
    .....
    default:
        System.out.println("Invalid Input");
}
```

Koşullu ifade (...? ... : ...)

```
// Conditional Expression
booleanExpr ? trueExpr : falseExpr
// An expression that returns
// the value of trueExpr
// or falseExpr
```

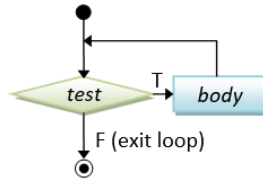
```
int num1 = 9, num2 = 8, max;
max = (num1 > num2) ? num1 : num2; // RHS returns num1 or num2
// same as
if (num1 > num2) {
    max = num1;
} else {
    max = num2;
}

int value = -9, absValue;
absValue = (value > 0) ? value : -value; // RHS returns value or -value
// same as
if (value > 0) absValue = value;
else absValue = -value;

int mark = 48;
System.out.println((mark >= 50) ? "PASS" : "FAIL"); // Return "PASS" or "FAIL"
// same as
if (mark >= 50) System.out.println("PASS");
else System.out.println("FAIL");
```

Döngüler - while

```
// while-do loop
while (booleanTest) {
    body;
}
// next statement
```

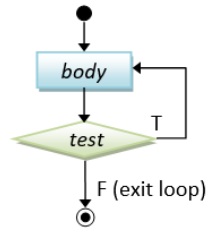


```
// Sum from 1 to upperbound
int sum = 0, upperbound = 100;
int number = 1; // init
while (number <= upperbound) {
    // number = 1, 2, 3, ..., upperbound for each iteration
    sum += number;
    ++number; // update
}
System.out.println("sum is: " + sum);

// Factorial of n (=1*2*3*...*n)
int n = 5;
int factorial = 1;
int number = 1; // init
while (number <= n) {
    // num = 1, 2, 3, ..., n for each iteration
    factorial *= number;
    ++num; // update
}
System.out.println("factorial is: " + factorial);
```

do-while

```
// do-while loop
do {
    body;
}
while (booleanTest);
// next statement
// Need a semi-colon to
// terminate do-while statement
```

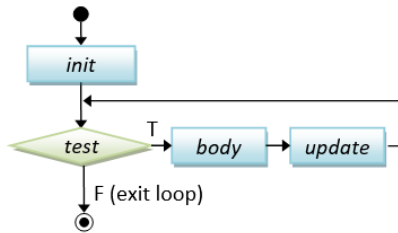


```
// Sum from 1 to upperbound
int sum = 0, upperbound = 100;
int number = 1; // init
do {
    // number = 1, 2, 3, ..., upperbound for each iteration
    sum += number;
    ++number; // update
} while (number <= upperbound);
System.out.println("sum is: " + sum);

// Factorial of n (=1*2*3*...*n)
int n = 5;
int factorial = 1;
int number = 1; // init
do {
    // num = 1, 2, 3, ..., n for each iteration
    factorial *= number;
    ++number; // update
} while (number <= n);
System.out.println("factorial is: " + factorial);
```


for

```
// for-loop
for (init; booleanTest; update) {
    body;
}
// next statement
```



```
// Sum from 1 to upperbound
int sum = 0, upperbound = 100;
for (int number = 1; number <= upperbound; ++number) {
    // num = 1, 2, 3, ..., upperbound for each iteration
    sum += number;
}
System.out.println("sum is: " + sum);

// Factorial of n (=1*2*3*...*n)
int n = 5;
int factorial = 1;
for (int number = 1; number <= n; ++number) {
    // number = 1, 2, 3, ..., n for each iteration
    factorial *= number;
}
System.out.println("factorial is: " + factorial);
```

Döngüler

- ▶ Sayac değişkeni döngünün her yinelenmesi için 1, 2, 3, ..., üst sınır değerlerini kontrol eden indis değişkeni olarak hizmet eder. İndeks değişkenini açıkça artırmanız / azaltmanız / değiştirmeniz gerekir(++/--). Aksi takdirde, test (sayı <= üst sınır) aynı sayı değeri için aynı sonucu döndüreceğinden döngü sonsuz bir döngü haline gelir.

Döngü İndeksi / Sayaç Değişkeni

```
// Sum from 1 to upperbound using for-loop
int sum = 0
int upperbound = 100;
for (int number = 1; number <= upperbound; ++number) { // number = 1, 2, 3, ..., upperbound for each iteration
    sum += number;
}

// Sum from 1 to upperbound using while-loop
int sum = 0
int upperbound = 100;
int number = 1;
while (number <= upperbound) { // number = 1, 2, 3, ..., upperbound for each iteration
    sum += number;
    ++number;
}
```

For döngüsü için, indis değişken numarası döngü içinde bildirilir ve bu nedenle yalnızca döngü içinde kullanılabilir. Döngüden sonra yok edildiği için değişkene döngüden sonra erişemezsiniz. Öte yandan, while döngüsü için, dizin değişken numarası döngünün içinde ve dışında mevcuttur.

```

/**
 * Sum the running integers from lowerbound to an upperbound.
 * Also compute the average.
 */
public class SumAverageRunningNumbers {
    public static void main(String[] args) {
        // Declare variables
        int sum = 0;        // store the accumulated sum
        int lowerbound = 1;
        int upperbound = 1000;
        double average;

        // Use a for-loop to accumulate the sum
        for (int number = lowerbound; number <= upperbound; ++number) {
            // number = lowerbound, lowerbound+1, lowerbound+2, ..., upperbound for each iteration
            sum += number;
        }
        average = (double)sum / (upperbound - lowerbound + 1); // need to cast int to double first
        // Print results
        System.out.println("The sum from " + lowerbound + " to " + upperbound + " is: " + sum);
        //The sum from 1 to 1000 is: 500500
        System.out.println("The average is: " + average);
        //The average is: 500.5

        // Sum only the ODD numbers
        int count = 0;        // counts of odd numbers
        sum = 0;              // reset sum for accumulation again
        // Adjust the lowerbound to the next odd number if it is a even number
        if (lowerbound % 2 == 0) {
            lowerbound++;
        }
        // Use a for-loop to accumulate the sum with step size of 2
        for (int number = lowerbound; number <= upperbound; number += 2) {
            // number = lowerbound, lowerbound+2, lowerbound+4, ..., (<=)upperbound for each iteration
            ++count;
            sum += number;
        }
        average = (double)sum / count;
        System.out.println("The sum of odd numbers is: " + sum);
        //The sum of odd numbers is: 250000
        System.out.println("The average of odd numbers is: " + average);
        //The average of odd numbers is: 500.0
    }
}

```

Döngü Kontrolü için Boole Bayrağı Kullanma

```
// Game loop
boolean gameOver = false;
while (!gameOver) {
    // play the game
    .....
    // Update the game state
    // Set gameOver to true if appropriate to exit the game loop
    if ( ..... ) {
        gameOver = true;    // exit the loop upon the next iteration test
    }
}
```

```
// for (init; test; update) { ..... }
for (int row = 0, col = 0; row < SIZE; ++row, ++col) {
    // Process diagonal elements (0,0), (1,1), (2,2),...
    .....
}
```

```
// Input with validity check
boolean isValid = false;
int number;
do {
    // prompt user to enter an int between 1 and 10
    .....
    // if the number entered is valid, set done to exit the loop
    if (number >= 1 && number <= 10) {
        isValid = true;    // exit the loop upon the next iteration test
        // Do the operations
        .....
    } else {
        // Print error message and repeat (isValid remains false)
        .....
    }
} while (!isValid);    // Repeat for invalid input
```

Programı Sonlandırma

- ▶ **System.exit (int exitCode):** Programı sonlandırmak ve denetimi Java Runtime'a döndürmek için System.exit (int exitCode) yöntemini çağırabilirsiniz. Geleneksel olarak, sıfır dönüş kodu normal sonlandırmayı gösterir; sıfır olmayan bir exitCode ise anormal sonlandırmayı gösterir.
- ▶ **Return ifadesi:** main () yöntemini sonlandırmak ve denetimi Java Runtime'a geri döndürmek için main () yönteminde bir "return" ifadesi de kullanabilirsiniz.

```
if (errorCount > 10) {  
    System.out.println("too many errors");  
    System.exit(1); // Terminate the program with abnormal exit code of 1  
}
```

```
public static void main(String[] args) {  
    ...  
    if (errorCount > 10) {  
        System.out.println("too many errors");  
        return; // Terminate and return control to Java Runtime from main()  
    }  
    ...  
}
```

KAYNAKLAR

- ▶ Programming Notes. (2021, March 11). Retrieved from <https://www3.ntu.edu.sg/home/ehchua/programming/index.html>
- ▶ Çobanoğlu B. (2020) Java ile Programlama ve Veri Yapıları. İstanbul Pusula Yayıncılık. 978-605-2359-84-6