

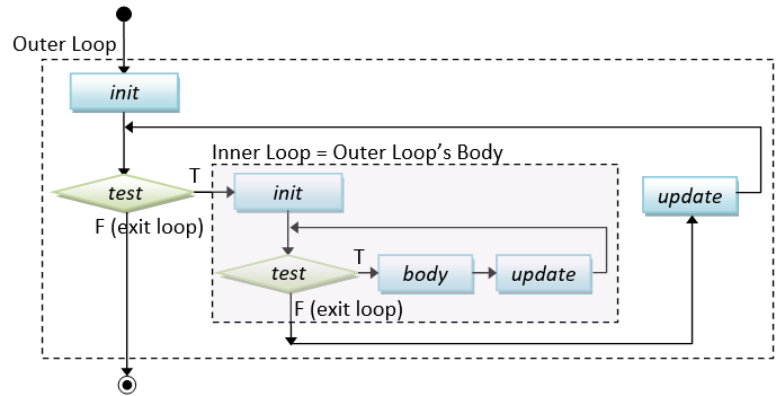
İleri Programlama

Döngü Yapıları ve Input-Output

Hüseyin Ahmetođlu

İç içe döngüler

```
for (...; ...; ...) { // outer loop
  // Before running the inner loop
  .....
  for (...; ...; ...) { // inner loop
    .....
  }
  // After running the inner loop
  .....
}
```



Örnek

Enter the size: 5

```
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

```
for (int row = 1; row <= size; row++) { // outer loop for rows
    ..... // before each row
    for (int col = 1; col <= size; col++) { // inner loop for columns
        if (.....) {
            System.out.print(.....); // without newline
        } else {
            System.out.print(.....);
        }
    }
    ..... // after each row
    System.out.println(); // Print a newline after all the columns
}
```

```
1 import java.util.Scanner;
2 /**
3  * Prompt user for the size; and print Square pattern
4  */
5 public class PrintSquarePattern {
6     public static void main (String[] args) {
7         // Declare variables
8         int size; // size of the pattern to be input
9         Scanner in = new Scanner(System.in);
10
11        // Prompt user for the size and read input as "int"
12        System.out.print("Enter the size: ");
13        size = in.nextInt();
14
15        // Use nested-loop to print a 2D pattern
16        // Outer loop to print ALL the rows
17        for (int row = 1; row <= size; row++) {
18            // Inner loop to print ALL the columns of EACH row
19            for (int col = 1; col <= size; col++) {
20                System.out.print("* ");
21            }
22            // Print a newline after all the columns
23            System.out.println();
24        }
25        in.close();
26    }
27 }
```

Enter the size: 10

*	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

```
import java.util.Scanner;
/**
 * Prompt user for the size and print the multiplication table.
 */
public class PrintTimeTable {
    public static void main(String[] args) {
        // Declare variables
        int size; // size of table to be input
        Scanner in = new Scanner(System.in);

        // Prompt for size and read input as "int"
        System.out.print("Enter the size: ");
        size = in.nextInt();

        // Print header row
        System.out.print(" * |");
        for (int col = 1; col <= size; ++col) {
            System.out.printf("%4d", col);
        }
        System.out.println(); // End row with newline
        // Print separator row
        System.out.print("----");
        for (int col = 1; col <= size; ++col) {
            System.out.printf("%4s", "----");
        }
        System.out.println(); // End row with newline

        // Print body using nested-loops
        for (int row = 1; row <= size; ++row) { // outer loop
            System.out.printf("%2d |", row); // print row header first
            for (int col = 1; col <= size; ++col) { // inner loop
                System.out.printf("%4d", row*col);
            }
            System.out.println(); // print newline after all columns
        }
    }
}
```

break ve continue | Döngü Akışını Kesmek

- ▶ Break deyimi mevcut (en içteki) döngüden çıkar.
- ▶ Continue deyimi geçerli yinelemeyi iptal eder ve geçerli (en içteki) döngünün bir sonraki yinelemesine devam eder.
- ▶ **break ve continue**, okunması ve takip edilmesi zor olduğu için zayıf yapılardır. Kullanımında dikkat edilmelidir.

Sonsuz Döngüler

```
for (;;) {  
    ..... // Need break inside the loop body  
}  
  
while (true) {  
    ..... // Need break inside the loop body  
}  
  
do {  
    ..... // Need break inside the loop body  
} while (true);
```

Örnek: break

- Aşağıdaki program, 2 ile bir üst sınır arasındaki asal olmayan sayıları listeler.

```
/**
 * List all non-prime numbers between 2 and an upperbound
 */
public class NonPrimeList {
    public static void main(String[] args) {
        int upperbound = 100;
        for (int number = 2; number <= upperbound; ++number) {
            // Not a prime, if there is a factor between 2 and sqrt(number)
            int maxFactor = (int)Math.sqrt(number);
            for (int factor = 2; factor <= maxFactor; ++factor) {
                if (number % factor == 0) { // Factor?
                    System.out.println(number + " is NOT a prime");
                    break; // A factor found, no need to search for more factors
                }
            }
        }
    }
}
```

Örnek: break

- Aşağıdaki program, 2 ile bir üst sınır arasındaki asal sayıları listeler.

```
/**
 * List all prime numbers between 2 and an upperbound
 */
public class PrimeListWithBreak {
    public static void main(String[] args) {
        int upperbound = 100;
        for (int number = 2; number <= upperbound; ++number) {
            // Not a prime, if there is a factor between 2 and sqrt(number)
            int maxFactor = (int)Math.sqrt(number);
            boolean isPrime = true; // boolean flag to indicate whether number is a prime
            for (int factor = 2; factor <= maxFactor; ++factor) {
                if (number % factor == 0) { // Factor?
                    isPrime = false; // number is not a prime
                    break; // A factor found, no need to search for more factors
                }
            }
            if (isPrime) System.out.println(number + " is a prime");
        }
    }
}
```


Örnek: break

- Aşağıdaki program, 2 ile bir üst sınır arasındaki asal sayıları listeler.

```
/**
 * List all prime numbers between 2 and an upperbound
 */
public class PrimeList {
    public static void main(String[] args) {
        int upperbound = 100;
        for (int number = 2; number <= upperbound; ++number) {
            // Not prime, if there is a factor between 2 and sqrt of number
            int maxFactor = (int)Math.sqrt(number);
            boolean isPrime = true;
            int factor = 2;
            while (isPrime && factor <= maxFactor) {
                if (number % factor == 0) { // Factor of number?
                    isPrime = false;
                }
                ++factor;
            }
            if (isPrime) System.out.println(number + " is a prime");
        }
    }
}
```

Örnek: continue

```
// Sum 1 to upperbound, exclude 11, 22, 33,...
int upperbound = 100;
int sum = 0;
for (int number = 1; number <= upperbound; ++number) {
    if (number % 11 == 0) continue; // Skip the rest of the loop body, continue to the next iteration
    sum += number;
}

// It is better to re-write the loop as:
for (int number = 1; number <= upperbound; ++number) {
    if (number % 11 != 0) sum += number;
}
```

Printf ()

- ▶ System.out.print () ve println (), bir int değişken yazdırmak için boşlukların sayısını ve bir double değişken için ondalık basamakların sayısını denetlemek gibi çıktı biçimlendirmelerini desteklemez.

Java SE 5, biçimlendirilmiş çıktı için printf () adında yeni bir yöntem sunmuştur (C Dilinin printf () 'inden sonra modellenmiştir). printf () şu biçimi alır:

```
printf(formattingString, arg1, arg2, arg3, ... );
```

Printf ()

- ▶ Biçimlendirme dizesi hem normal metinleri hem de Biçim Belirteçleri(*conversionCode*) olarak adlandırılanları içerir. Normal metinler (beyaz boşluklar dahil) olduğu gibi yazdırılacaktır.
- ▶ "%[*flags*][*width*]*conversionCode* " biçimindeki biçim belirteçleri, genellikle bire bir ve sıralı bir şekilde *formattingString* izleyen bağımsız değişkenlerle değiştirilir.
- ▶ Bir biçim belirticisi bir "%" ile başlar ve *ConversionCode* ile biter, ör. Tamsayı için %d, kayan noktalı sayı için %f, karakter için %c ve Dize için %s.
- ▶ Alan genişliğini belirtmek için arasına isteğe bağlı bir genişlik eklenebilir. Benzer şekilde, hizalama, doldurma ve diğerlerini kontrol etmek için isteğe bağlı bir bayraklar kullanılabilir.

Printf ()

<pre>// Without specifying field-width System.out.printf("Hi, %s %d %f ,@xyz%n", "Hello", 123, 45.6);</pre>	Hi, Hello 123 45.600000 ,@xyz
<pre>// Specifying the field-width and decimal places for double System.out.printf("Hi, %6s %6d %6.2f ,@xyz%n", "Hello", 123, 45.6);</pre>	Hi, Hello 123 45.60 ,@xyz
<pre>// Various way to format integers: // flag '-' for left-align, '0' for padding with 0 System.out.printf("Hi, %d %5d %5d %05d ,@xyz%n", 111, 222, 333, 444);</pre>	Hi, 111 222 333 00444 ,@xyz
<pre>// Various way to format floating-point numbers: // flag '-' for left-align System.out.printf("Hi, %f %7.2f %2f %-7.2f ,@xyz%n", 11.1, 22.2, 33.3, 44.4);</pre>	Hi, 11.100000 22.20 33.30 44.40 ,@xyz
<pre>// To print a '%', use %% (as % has special meaning) System.out.printf("The rate is: %.2f%%\n", 1.2);</pre>	The rate is: 1.20%.

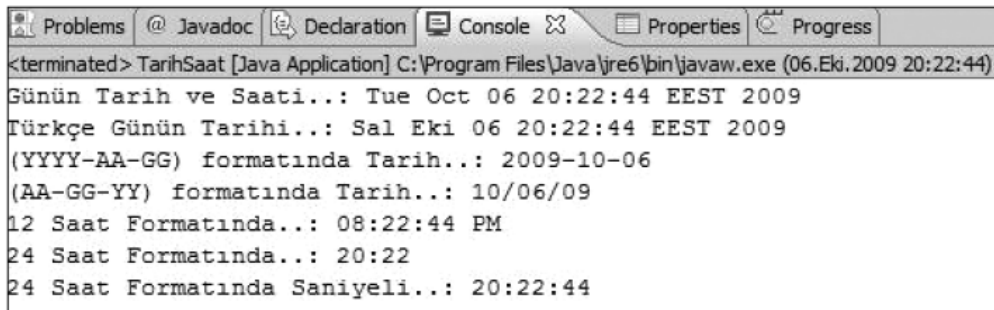
Printf ()

Basılacak Karakter veya İşlevi	Çıkış Karakteri	Örnek Kod	Ekran Çıktısı
Ondalıklı (Decimal) tamsayı	%d	System.out.printf("%d\n", +26); System.out.printf("%d\n", -26);	26 -26
Sekizli (Octal) tamsayı	%o	System.out.printf("%o", +26);	32
Onaltılı (Hexadecimal) tamsayı	%x veya %X	System.out.printf("%x\n", 26); System.out.printf("%X", 26);	1a 1A
Tek Karakter	%c	System.out.printf("%c", 'A');	A
String ifade	%s veya %S	System.out.printf("%s\n", "Ali"); System.out.printf("%S\n", "Veli"); System.out.printf("%3.2s\n", "bade"); (3 karakter boşluk bıraktıktan sonra 2 karakter gösterir)	Ali VELİ ØØba
Gerçel (float) sayı, Standart gösterim	%f	System.out.printf("%f", +26.0);	26,000000
Gerçel(float)sayı, Bilimsel gösterim	%e veya %E	System.out.printf("%e", +26.0);	2.600000e+01

Date() Sınıfı

```
import java.util.Date;
class TarihSaat{
    public static void main(String[] args) {
        Date bugun = new Date();
        System.out.printf("Günün Tarih ve Saati..: %s\n", bugun);
        System.out.printf("Türkçe Günü Tarihi..: %tc\n", bugun);
        System.out.printf("(YYYY-AA-GG) formatında Tarih..: %tF\n", bugun);
        System.out.printf("(AA-GG-YY) formatında Tarih..: %tD\n", bugun);
        System.out.printf("12 Saat Formatında..: %tr\n", bugun);
        System.out.printf("24 Saat Formatında..: %tR\n", bugun);
        System.out.printf("24 Saat Formatında Saniyeli..: %tT\n", bugun);
    }
}
```

Programın ekran çıktısı aşağıdaki gibi olur.



```
<terminated> TarihSaat [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (06.Eki.2009 20:22:44)
Günün Tarih ve Saati..: Tue Oct 06 20:22:44 EEST 2009
Türkçe Günü Tarihi..: Sal Eki 06 20:22:44 EEST 2009
(YYYY-AA-GG) formatında Tarih..: 2009-10-06
(AA-GG-YY) formatında Tarih..: 10/06/09
12 Saat Formatında..: 08:22:44 PM
24 Saat Formatında..: 20:22
24 Saat Formatında Saniyeli..: 20:22:44
```

Calendar() Sınıfı

Date() sınıfı gibi Calendar() sınıfının metotlarını kullanarak değişik formatlarda (YEAR, MONTH, DATE) tarih bilgisini elde edebiliriz.

Kullanım şekli:

```
Calendar bugün = Calendar.getInstance();
System.out.println("Yıl : " + bugün.get(Calendar.YEAR));
```

Örnek 3.4: Gün, Ay, Yıl ve Saat değerlerini ayrı ayrı olarak Calendar() sınıfını kullanarak veren programı aşağıdaki gibi yazabiliriz.

```
import java.util.Calendar;
public class Tarih2{
    public static void main(String[] args) {

        Calendar bugün = Calendar.getInstance();
        System.out.println("Yıl : " + bugün.get(Calendar.YEAR));
        // Ay değeri 0 ile 11 arasında üretildiği için 1 eklendi
        System.out.println("Ay : " + (bugün.get(Calendar.MONTH) + 1));
        System.out.println("Gün : " + bugün.get(Calendar.DATE));
        System.out.print("Saati : " + bugün.get(Calendar.HOUR));
        System.out.print(":" + bugün.get(Calendar.MINUTE));
    }
}
```



```
Problems @ Ja
<terminated> Tarih2
Yıl : 2009
Ay : 10
Gün : 6
Saati : 9:24
```

Programın herhangi bir andaki ekran çıktısı yandaki gibidir.

Scanner

- ▶ Java, diğer tüm diller gibi, üç standart giriş / çıkış akışını destekler: `System.in` (standart giriş), `System.out` (standart çıkış) ve `System.err` (standart hata).
- ▶ `System.in` varsayılan olarak klavyeden değer okur; `System.out` ve `System.err` varsayılan olarak konsola görüntü verir. Diğer cihazlara değer yönlendirebilirler, örneğin, `System.err`'ı bu hata mesajlarını kaydetmek için bir disk dosyasına değer yazabilir.

Scanner

- ▶ JDK 5 ile birlikte biçimlendirilmiş girdiyi basitleştirmek için `java.util` paketinde `Scanner` adlı yeni bir sınıf (ve daha önce açıklanan biçimlendirilmiş çıktı için yeni bir yöntem `printf ()`) tanıtıldı.
- ▶ `System.in`'deki (klavye) girdiyi taramak için bir `Scanner` oluşturabilir ve sonraki `int`, `double` ve `String` belirtecini (boşluk beyaz boşlukla sınırlandırılmış) ayrıştırmak için `nextInt ()`, `nextDouble ()`, `next ()` gibi yöntemleri kullanabilirsiniz.

Scanner

İşlevi	Yöntem Adı
Klavyeden girilen tamsayı (integer) türündeki sayıyı okur.	<code>nextInt()</code>
Klavyeden girilen tamsayı (byte) türündeki sayıyı okur.	<code>nextByte()</code>
Klavyeden girilen boolean türündeki önermeyi okur.	<code>nextBoolean()</code>
Klavyeden girilen ifadeyi kesirli (gerçek) sayı (float) türüne çevirir.	<code>nextFloat()</code>
Klavyeden girilen ifadeyi uzun kesirli (gerçek) sayı (double) türüne çevirir.	<code>nextDouble()</code>
Klavyeden girilen ifadenin ilk kelimesini okur. (Boşluk (space) karakterine kadar olan kısmı tarar.)	<code>next()</code>
Eğer veri kaynağında veri varsa true yoksa false değerini döndürür. Boolean türdedir. Klavyeden girilen veriler üzerinde işlem yapmak için kullanılır.	<code>hasNext()</code>
Klavyeden girilen tüm satırı okur. String türünde bir satır okur.	<code>nextLine()</code>
Tek bir karakteri okur. (Girilen ifadeyi karakter karakter okur.)	<code>findInLine(".").charAt(0)</code>

Scanner

```
import java.util.Scanner; // Needed to use the Scanner
/**
 * Test input scanner
 */
public class ScannerTest {
    public static void main(String[] args) {
        // Declare variables
        int num1;
        double num2;
        String str;
        // Construct a Scanner named "in" for scanning System.in (keyboard)
        Scanner in = new Scanner(System.in);

        // Read inputs from keyboard
        System.out.print("Enter an integer: "); // Show prompting message
        num1 = in.nextInt(); // Use nextInt() to read an int
        System.out.print("Enter a floating point: "); // Show prompting message
        num2 = in.nextDouble(); // Use nextDouble() to read a double
        System.out.print("Enter a string: "); // Show prompting message
        str = in.next(); // Use next() to read a String token, up to white space

        // Formatted output via printf()
        System.out.printf("%s, Sum of %d & %.2f is %.2f\n", str, num1, num2, num1+num2);

        // close the input
        in.close();
    }
}
```

Scanner

- ▶ Ayrıca, beyaz boşluklar dahil, ancak biten yeni satırı hariç tutarak tüm satırı okumak için `nextLine ()` yöntemini de kullanabilirsiniz.

```
/**
 * Test Scanner's nextLine()
 */
import java.util.Scanner; // Needed to use the Scanner
public class ScannerNextLineTest {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a string (with space): ");
        // Use nextLine() to read entire line including white spaces,
        // but excluding the terminating newline.
        String str = in.nextLine();
        System.out.printf("%s\n", str);
        in.close();
    }
}
```

Örnek

```
Enter first integer: 8
Enter second integer: 9
The sum is: 17
```

```
import java.util.Scanner; // For keyboard input
/**
 * 1. Prompt user for 2 integers
 * 2. Read inputs as "int"
 * 3. Compute their sum in "int"
 * 4. Print the result
 */
public class Add2Integer { // Save as "Add2Integer.java"
    public static void main (String[] args) {
        // Declare variables
        int number1, number2, sum;
        Scanner in = new Scanner(System.in); // Scan the keyboard for input

        // Put up prompting messages and read inputs as "int"
        System.out.print("Enter first integer: "); // No newline for prompting message
        number1 = in.nextInt(); // Read next input as "int"
        System.out.print("Enter second integer: ");
        number2 = in.nextInt();

        // Compute sum
        sum = number1 + number2;

        // Display result
        System.out.println("The sum is: " + sum); // Print with newline

        // Close Scanner
        in.close();
    }
}
```

Örnek

```
import java.util.Scanner; // For keyboard input
/**
 * 1. Prompt user for the taxable income in integer.
 * 2. Read input as "int".
 * 3. Compute the tax payable using nested-if in "double".
 * 4. Print the values rounded to 2 decimal places.
 */
public class IncomeTaxCalculator {
    public static void main(String[] args) {
        // Declare constants first (variables may use these constants)
        final double TAX_RATE_ABOVE_20K = 0.1;
        final double TAX_RATE_ABOVE_40K = 0.2;
        final double TAX_RATE_ABOVE_60K = 0.3;

        // Declare variables
        int taxableIncome;
        double taxPayable;
        Scanner in = new Scanner(System.in);

        // Prompt and read inputs as "int"
        System.out.print("Enter the taxable income: $");
        taxableIncome = in.nextInt();
```

```
// Compute tax payable in "double" using a nested-if to handle 4 cases
if (taxableIncome <= 20000) { // [0, 20000]
    taxPayable = 0;
} else if (taxableIncome <= 40000) { // [20001, 40000]
    taxPayable = (taxableIncome - 20000) * TAX_RATE_ABOVE_20K;
} else if (taxableIncome <= 60000) { // [40001, 60000]
    taxPayable = 20000 * TAX_RATE_ABOVE_20K
        + (taxableIncome - 40000) * TAX_RATE_ABOVE_40K;
} else { // >=60001
    taxPayable = 20000 * TAX_RATE_ABOVE_20K
        + 20000 * TAX_RATE_ABOVE_40K
        + (taxableIncome - 60000) * TAX_RATE_ABOVE_60K;
}

// Print result rounded to 2 decimal places
System.out.printf("The income tax payable is: $%.2f%n", taxPayable);
in.close(); // Close Scanner
}
```

```
Enter the taxable income: $41234
The income tax payable is: $2246.80
```

```
Enter the taxable income: $67891
The income tax payable is: $8367.30
```

```
Enter the taxable income: $85432
The income tax payable is: $13629.60
```

```
Enter the taxable income: $12345
The income tax payable is: $0.00
```

Örnek: Vergi Hesabı

- ▶ Önceki örneğe dayanarak, kullanıcı -1 girene kadar hesaplamaları tekrarlayacak olan `IncomeTaxCalculatorSentinel` adlı bir program yazın.
- ▶ -1, sentinel değer olarak bilinir. (Programlamada, aynı zamanda bir bayrak değeri, açma değeri, sahte değer, sinyal değeri veya yapay veri olarak da adlandırılan bir sentinel değer, varlığını bir sonlandırma koşulu olarak kullanan özel bir değerdir.)

```
Enter the taxable income: $41000
The income tax payable is: $2200.00
Enter the taxable income: $62000
The income tax payable is: $6600.00
Enter the taxable income: $73123
The income tax payable is: $9936.90
Enter the taxable income: $84328
The income tax payable is: $13298.40
Enter the taxable income: $-1
bye!
```

```
// Get first input to "seed" the while loop
input = .....;
while (input != SENTINEL) {
    // Process input
    .....
    .....
    // Get next input and repeat the loop
    input = .....;    // Need to repeat these statements
}
.....
```



```

import java.util.Scanner; // For keyboard input
/**
 * 1. Prompt user for the taxable income in integer.
 * 2. Read input as "int".
 * 3. Compute the tax payable using nested-if in "double".
 * 4. Print the values rounded to 2 decimal places.
 * 5. Repeat until user enter -1.
 */
public class IncomeTaxCalculatorSentinel {
    public static void main(String[] args) {
        // Declare constants first (variables may use these constants)
        final double TAX_RATE_ABOVE_20K = 0.1;
        final double TAX_RATE_ABOVE_40K = 0.2;
        final double TAX_RATE_ABOVE_60K = 0.3;
        final int SENTINEL = -1; // Terminating value for input

        // Declare variables
        int taxableIncome;
        double taxPayable;
        Scanner in = new Scanner(System.in);

        // Read the first input to "seed" the while loop
        System.out.print("Enter the taxable income: $");
        taxableIncome = in.nextInt();

```

```

        while (taxableIncome != SENTINEL) {
            // Compute tax payable in "double" using a nested-if to handle 4 cases
            if (taxableIncome > 60000) {
                taxPayable = 20000 * TAX_RATE_ABOVE_20K
                    + 20000 * TAX_RATE_ABOVE_40K
                    + (taxableIncome - 60000) * TAX_RATE_ABOVE_60K;
            } else if (taxableIncome > 40000) {
                taxPayable = 20000 * TAX_RATE_ABOVE_20K
                    + (taxableIncome - 40000) * TAX_RATE_ABOVE_40K;
            } else if (taxableIncome > 20000) {
                taxPayable = (taxableIncome - 20000) * TAX_RATE_ABOVE_20K;
            } else {
                taxPayable = 0;
            }

            // Print result rounded to 2 decimal places
            System.out.printf("The income tax payable is: $%.2f\n", taxPayable);

            // Read the next input
            System.out.print("Enter the taxable income: $");
            taxableIncome = in.nextInt();
            // Repeat the loop body, only if the input is not the SENTINEL value.
            // Take note that you need to repeat these two statements inside/outside the loop!
        }
        System.out.println("bye!");
        in.close(); // Close Scanner
    }
}

```

Örnek: Sayı Tahmini

```
boolean done = false;

while (!done) {
    if (.....) {
        done = true; // exit the loop upon the next iteration
        .....
    }
    ..... // done remains false. repeat loop
}
```

```
import java.util.Scanner;
/**
 * Guess a secret number between 0 and 99.
 */
public class NumberGuess {
    public static void main(String[] args) {
        // Define variables
        int secretNumber; // Secret number to be guessed
        int numberIn; // The guessed number entered
        int trialNumber = 0; // Number of trials so far
        boolean done = false; // boolean flag for loop control
        Scanner in = new Scanner(System.in);

        // Set up the secret number: Math.random() generates a double in [0.0, 1.0)
        secretNumber = (int)(Math.random()*100);

        // Use a while-loop to repeatedly guess the number until it is correct
        while (!done) {
            ++trialNumber;
            System.out.print("Enter your guess (between 0 and 99): ");
            numberIn = in.nextInt();
            if (numberIn == secretNumber) {
                System.out.println("Congratulation");
                done = true;
            } else if (numberIn < secretNumber) {
                System.out.println("Try higher");
            } else {
                System.out.println("Try lower");
            }
        }
        System.out.println("You got in " + trialNumber + " trials");
        in.close();
    }
}
```

Metin Dosyalarını Scanner ile Okumak

- ▶ System.in (klavye) taraması dışında, Tarayıcınızı bir disk dosyası veya ağ soketi gibi herhangi bir giriş kaynağını taramak için bağlayabilir ve aynı yöntem setini nextInt (), nextDouble (), next (), nextLine () sonraki int, double, String ve satırı ayrıştırmak için kullanabilir.

```
Scanner in = new Scanner(new File("in.txt")); // Construct a Scanner to scan a text file
// Use the same set of methods to read from the file
int anInt = in.nextInt();           // next String
double aDouble = in.nextDouble(); // next double
String str = in.next();             // next int
String line = in.nextLine();       // entire line
```

Metin Dosyalarını Scanner ile Okumak

- Bir dosyayı `new File(filename)` yoluyla açmak için, `FileNotFoundException` kullanmanız gerekir, yani açmaya çalıştığınız dosya bulunamıyor ise hata dönecektir. Aksi takdirde, programınızı derleyemezsiniz. Bu istisnai halletmenin iki yolu vardır: *throws* veya *try-catch*.

```
/**
 * Input from File.
 * Technique 1: Declare "throws FileNotFoundException" in the enclosing main() method
 */
import java.util.Scanner;           // Needed for using Scanner
import java.io.File;               // Needed for file operation
import java.io.FileNotFoundException; // Needed for file operation
public class TextFileScannerWithThrows {
    public static void main(String[] args)
        throws FileNotFoundException { // Declare "throws" here

        int num1;
        double num2;
        String name;
        Scanner in = new Scanner(new File("in.txt")); // Scan input from text file
        num1 = in.nextInt(); // Read int
        num2 = in.nextDouble(); // Read double
        name = in.next(); // Read String
        System.out.printf("Hi %s, the sum of %d and %.2f is %.2f\n", name, num1, num2, num1+num2);
        in.close();
    }
}
```

Metin Dosyalarını Scanner ile Okumak

```
/**
 * Input from File.
 * Technique 2: Use try-catch to handle exception
 */
import java.util.Scanner;           // Needed for using Scanner
import java.io.File;               // Needed for file operation
import java.io.FileNotFoundException; // Needed for file operation
public class TextFileScannerWithCatch {
    public static void main(String[] args) {
        int num1;
        double num2;
        String name;
        try {                        // try these statements
            Scanner in = new Scanner(new File("in.txt"));
            num1 = in.nextInt();      // Read int
            num2 = in.nextDouble();   // Read double
            name = in.next();         // Read String
            System.out.printf("Hi %s, the sum of %d and %.2f is %.2f\n", name, num1, num2, num1+num2);
            in.close();
        } catch (FileNotFoundException ex) { // catch and handle the exception here
            ex.printStackTrace();         // print the stack trace
        }
    }
}
```

Metin Dosyaların Formatted ile Yazmak

```
/**
 * Output to File.
 * Technique 1: Declare "throws FileNotFoundException" in the enclosing main() method
 */
import java.io.File;
import java.util.Formatter;           // <== note
import java.io.FileNotFoundException; // <== note
public class TextFileFormatterWithThrows {
    public static void main(String[] args)
        throws FileNotFoundException { // <== note
        // Construct a Formatter to write formatted output to a text file
        Formatter out = new Formatter(new File("out.txt"));
        // Write to file with format() method (similar to printf())
        int num1 = 1234;
        double num2 = 55.66;
        String name = "Paul";
        out.format("Hi %s,%n", name);
        out.format("The sum of %d and %.2f is %.2f%n", num1, num2, num1 + num2);
        out.close(); // Close the file
        System.out.println("Done"); // Print to console
    }
}
```

Metin Dosyaların Formatted ile Yazmak

```
/**
 * Output to File.
 * Technique 2: Use try-catch to handle exception
 */
import java.io.File;
import java.util.Formatter;           // <== note
import java.io.FileNotFoundException; // <== note

public class TextFileFormatterWithCatch {
    public static void main(String[] args) {
        try { // try the following statements
            // Construct a Formatter to write formatted output to a text file
            Formatter out = new Formatter(new File("out.txt"));
            // Write to file with format() method (similar to printf())
            int num1 = 1234;
            double num2 = 55.66;
            String name = "Pauline";
            out.format("Hi %s,%n", name);
            out.format("The sum of %d and %.2f is %.2f%n", num1, num2, num1 + num2);
            out.close(); // Close the file
            System.out.println("Done"); // Print to console
        } catch (FileNotFoundException ex) { // catch the exception here
            ex.printStackTrace(); // Print the stack trace
        }
    }
}
```

Dialog Box

- ▶ JOptionPane sınıfını kullanarak grafik iletişim kutusu aracılığıyla kullanıcılardan girdi alabilirsiniz.

```
1  /**
2   * Input via a Dialog box
3   */
4  import javax.swing.JOptionPane;    // Needed to use JOptionPane
5  public class JOptionPaneTest {
6      public static void main(String[] args) {
7          String radiusStr;
8          double radius, area;
9          // Read input String from dialog box
10         radiusStr = JOptionPane.showInputDialog("Enter the radius of the circle");
11         radius = Double.parseDouble(radiusStr);    // Convert String to double
12         area = radius*radius*Math.PI;
13         System.out.println("The area is " + area);
14     }
15 }
```


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