



COMPUTER SCIENCE DEPARTMENT FACULTY OF  
ENGINEERING AND TECHNOLOGY  
**ADVANCED PROGRAMMING COMP231**

**Instructor :Murad Njoun**  
**Office : Masri322**

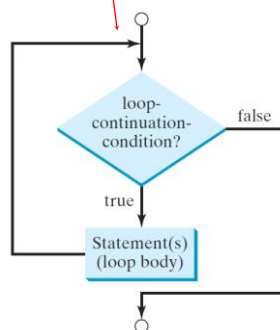


## Chapter 5 Loops

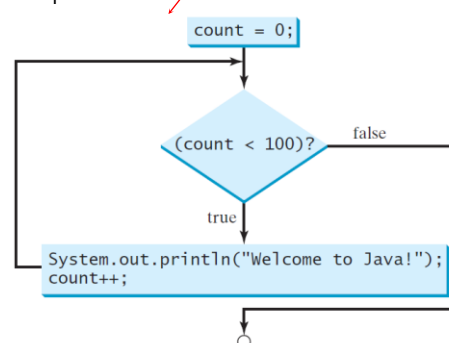
liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoun

### while Loop Flow Chart

```
while (loop-continuation-condition) {
    // loop-body;
    Statement(s);
}
```



```
int count = 0;
while (count < 100) {
    System.out.println("Welcome to Java!");
    count++;
}
```



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoun



## Trace while Loop

```
int count = 0;
```

Initialize count

```
while (count < 2) {
    System.out.println("Welcome to Java!");
    count++;
}
```

```
int count = 0;
```

(count < 2) is still true since count is 1

```
while (count < 2) {
    System.out.println("Welcome to Java!");
    count++;
}
```

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoun



## Trace while Loop, cont.

```
int count = 0;
```

Print Welcome to Java

```
while (count < 2) {
    System.out.println("Welcome to Java!");
    count++;
}
```

```
int count = 0;
```

Increase count by 1  
count is 2 now

```
while (count < 2) {
    System.out.println("Welcome to Java!");
    count++;
}
```

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoun



## Trace while Loop, cont.

```
int count = 0;
```

```
while (count < 2) {
```

```
    System.out.println("Welcome to Java!");
```

```
    count++;
```

```
}
```

```
int count = 0;
```

```
while (count < 2) {
```

```
    System.out.println("Welcome to Java!");
```

```
    count++;
```

```
}
```

(count < 2) is false since count is 2 now

The loop exits. Execute the next statement after the loop.



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Problem: Guessing Numbers

Write a program that randomly generates an integer between 0 and 100, inclusive. The program prompts the user to enter a number continuously until the number matches the randomly generated number. For each user input, the program tells the user whether the input is too low or too high, so the user can choose the next input intelligently. Here is a sample run:

GuessNumberOneTime	Run
GuessNumber	Run



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Problem: An Advanced Math Learning Tool

The Math subtraction learning tool program generates just one question for each run. You can use a loop to generate questions repeatedly. This example gives a program that generates five questions and reports the number of the correct answers after a student answers all five questions.

SubtractionQuizLoop

Run



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Ending a Loop with a Sentinel Value

Often the number of times a loop is executed is not predetermined. You may use an input value to signify the end of the loop. Such a value is known as a *sentinel value*.

Write a program that reads and calculates the sum of an unspecified number of integers. The input 0 signifies the end of the input.

SentinelValue

Run



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Caution

Don't use floating-point values for equality checking in a loop control. Since floating-point values are approximations for some values, using them could result in imprecise counter values and inaccurate results. Consider the following code for computing  $1 + 0.9 + 0.8 + \dots + 0.1$ :

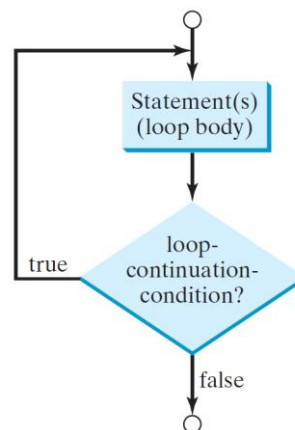
```
double item = 1; double sum = 0;
while (item != 0) { // No guarantee item will be 0
    sum += item;
    item -= 0.1;
}
System.out.println(sum);
```

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## do-while Loop

```
do {
    // Loop body;
    Statement(s);
} while (loop-continuation-condition).
```

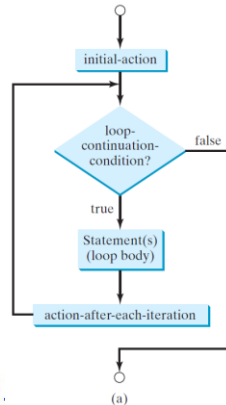


liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

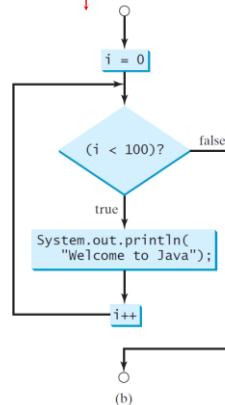


## for Loops

```
for (initial-action; loop-
continuation-condition; action-
after-each-iteration) {
// loop body;
Statement(s);
}
```



```
int i;
for (i = 0; i < 100; i++) {
System.out.println(
"Welcome to Java!");
}
```



liang introduction to

: Mr.Murad Njoum



## Trace for Loop

```
int i;
for (i = 0; i < 2; i++) {
System.out.println(
"Welcome to Java!");
}
```

Declare i

```
int i;
for (i = 0; i < 2; i++) {
System.out.println(
"Welcome to Java!");
}
```

Execute initializer  
i is now 0

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Trace for Loop, cont.

```
int i;
for (i = 0; i < 2; i++) {
    System.out.println("Welcome to Java!");
}
```

(i < 2) is true  
since i is 0

```
int i;
for (i = 0; i < 2; i++) {
    System.out.println("Welcome to Java!");
}
```

Print Welcome to Java



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Trace for Loop, cont.

```
int i;
for (i = 0; i < 2; i++) {
    System.out.println("Welcome to Java!");
}
```

Execute adjustment statement  
i now is 1

```
int i;
for (i = 0; i < 2; i++) {
    System.out.println("Welcome to Java!");
}
```

(i < 2) is still true  
since i is 1



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Trace for Loop, cont.

```
int i;
for (i = 0; i < 2; i++) {
  System.out.println("Welcome to Java!");
}
```

Print Welcome to Java

```
int i;
for (i = 0; i < 2; i++) {
  System.out.println("Welcome to Java!");
}
```

Execute adjustment statement  
i now is 2



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## Trace for Loop, cont.

```
int i;
for (i = 0; i < 2; i++) {
  System.out.println("Welcome to Java!");
}
```

(i < 2) is false  
since i is 2

```
int i;
for (i = 0; i < 2; i++) {
  System.out.println("Welcome to Java!");
}
```

Exit the loop. Execute the next  
statement after the loop



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Note

The initial-action in a for loop can be a list of zero or more comma-separated expressions. The action-after-each-iteration in a for loop can be a list of zero or more comma-separated statements. Therefore, the following two for loops are correct. They are rarely used in practice, however.

```
for (int i = 1; i < 100; System.out.println(i++));
```

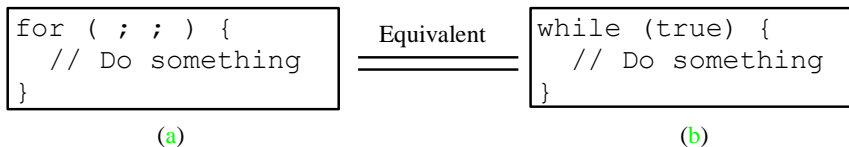
```
for (int i = 0, j = 0; (i + j < 10); i++, j++) {  
    // Do something  
}
```

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Note

If the loop-continuation-condition in a for loop is omitted, it is implicitly true. Thus the statement given below in (a), which is an infinite loop, is correct. Nevertheless, it is better to use the equivalent loop in (b) to avoid confusion:



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Caution


Adding a semicolon at the end of the for clause before the loop body is a common mistake, as shown below:

```

for (int i=0; i<10; i++);
{
    System.out.println("i is " + i);
}

```

Logic  
Error



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Caution, cont.


Similarly, the following loop is also wrong:

```

int i=0;
while (i < 10);
{
    System.out.println("i is " + i);
    i++;
}

```

Logic Error




In the case of the do loop, the following semicolon is needed to end the loop.

```

int i=0;
do {
    System.out.println("i is " + i);
    i++;
} while (i<10);

```

Correct



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Which Loop to Use?

The three forms of loop statements, while, do-while, and for, are expressively equivalent; that is, you can write a loop in any of these three forms. For example, a while loop in (a) in the following figure can always be converted into the following for loop in (b):

<pre>while (loop-continuation-condition) {   // Loop body }</pre>	Equivalent	<pre>for ( ; loop-continuation-condition; )   // Loop body</pre>
(a)		(b)

A for loop in (a) in the following figure can generally be converted into the following while loop in (b) except in certain special cases (see Review Question 3.19 for one of them):

<pre>for (initial-action;      loop-continuation-condition;      action-after-each-iteration) {   // Loop body; }</pre>	Equivalent	<pre>initial-action; while (loop-continuation-condition) {   // Loop body;   action-after-each-iteration; }</pre>
(a)		(b)

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Recommendations

Use the one that is most intuitive and comfortable for you. In general, a for loop may be used if the number of repetitions is known, as, for example, when you need to print a message 100 times. A while loop may be used if the number of repetitions is not known, as in the case of reading the numbers until the input is 0. A do-while loop can be used to replace a while loop if the loop body has to be executed before testing the continuation condition.

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Minimizing Numerical Errors

Numeric errors involving floating-point numbers are inevitable. This section discusses how to minimize such errors through an example.

Here is an example that sums a series that starts with 0.01 and ends with 1.0. The numbers in the series will increment by 0.01, as follows: 0.01 + 0.02 + 0.03 and so on.

TestSum

Run

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Problem:

### Finding the Greatest Common Divisor

Problem: Write a program that prompts the user to enter two positive integers and finds their greatest common divisor.

Solution: Suppose you enter two integers 4 and 2, their greatest common divisor is 2. Suppose you enter two integers 16 and 24, their greatest common divisor is 8. So, how do you find the greatest common divisor? Let the two input integers be  $n_1$  and  $n_2$ . You know number 1 is a common divisor, but it may not be the greatest common divisor. So you can check whether  $k$  (for  $k = 2, 3, 4$ , and so on) is a common divisor for  $n_1$  and  $n_2$ , until  $k$  is greater than  $n_1$  or  $n_2$ .

GreatestCommonDivisor

Run

liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum



## Case Study: *Converting Decimals to Hexadecimals*

Hexadecimals are often used in computer systems programming (see Appendix F for an introduction to number systems). How do you convert a decimal number to a hexadecimal number? To convert a decimal number  $d$  to a hexadecimal number is to find the hexadecimal digits  $h_n, h_{n-1}, h_{n-2}, \dots, h_2, h_1$ , and  $h_0$  such that

$$d = h_n \times 16^n + h_{n-1} \times 16^{n-1} + h_{n-2} \times 16^{n-2} + \dots + h_2 \times 16^2 + h_1 \times 16^1 + h_0 \times 16^0$$

These hexadecimal digits can be found by successively dividing  $d$  by 16 until the quotient is 0. The remainders are  $h_0, h_1, h_2, \dots, h_{n-2}, h_{n-1}$ , and  $h_n$ .

Dec2Hex

Run



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## break

```
public class TestBreak {
    public static void main(String[] args) {
        int sum = 0;
        int number = 0;

        while (number < 20) {
            number++;
            sum += number;
            if (sum >= 100)
                break;
        }
        System.out.println("The number is " + number);
        System.out.println("The sum is " + sum);
    }
}
```



liang introduction to java programming 11th edition ,2019 , Edit By : Mr.Murad Njoum

## continue

```
public class TestContinue {
    public static void main(String[] args) {
        int sum = 0;
        int number = 0;

        while (number < 20) {
            number++;
            if (number == 10 || number == 11)
                continue;
            sum += number;
        }

        System.out.println("The sum is " + sum);
    }
}
```

