Introduction to OOP with Java

Instructor: AbuKhleif, Mohammad Noor
Sep 2017
www.abukhleif.com

Lecture 04:

Control Flow Statements:
Loops

Instructor: AbuKhleif, Mohammad Noor
Sep 2017
www.abukhleif.com
Instructor

• AbuKhleif, ‘Mohammad Noor’
• Computer Engineer (JU 2012-2017)
• Software Automation Engineer @ Atypon – John Wiley and Sons Company - Jordan Branch

• Reach me at:
  • www.abukhleif.com
  • moh.noor94@gmail.com
  • facebook.com/moh.noor94
  • twitter.com/moh_noor94

Course

• Java SE Basics
• Object Oriented Programming
• Course Page:
• Or, go to: www.abukhleif.com → Courses → Java 101 Course – Sep 2017
• Course Facebook Group:
  www.facebook.com/groups/AKF2017Java
Let’s Start!

Motivations

Suppose that you need to print a string (e.g., "Welcome to Java!") a hundred times. It would be tedious to have to write the following statement a hundred times:

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

So, how do you solve this problem?
Loops

• A loop can be used to tell a program to execute statements repeatedly.
• Three types of loop statements:
  • While loops.
  • Do-while loops.
  • For loops.

Types of Loops
Types of Loops

• Two types of loops:
  • Counter-controlled loops:
    • A control variable is used to count the number of iterations.
  • Sentinel-controlled loops:
    • A special input value signifies the end of the iterations.

while Loops
**while Loops**

- A while loop executes statements repeatedly while the condition is **true**.
- The syntax for the while loop is:
  ```java
  while (loop-continuation-condition) {
    statement(s);
  }
  ```

**Example**

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

- This is the way to print "Welcome to Java" a hundred times.
Let’s Code

- Write a program that accepts integers by the user, stops when the user enters 0. Find the summation of all entered integers.

---

do-while Loops
**do-while** Loops

- Same as the while loop except that it executes the loop body first then checks the loop continuation condition.
- The syntax for the do-while loop:
  ```java
do {
  statement(s);
} while (loop-continuation-condition);
```

---

**Let’s Code**

- Rewrite the previous exercise using the do-while loop.

---

**Enough Talk Let’s Code**
for Loops

• A for loop has a concise syntax for writing loops.
• The syntax for the for loop is:
  
```java
for (initial-action; loop-continuation-condition; action-after-each-iteration) {
  statement(s);
}
```

![Diagram of for loop flow]
Example

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

• This is the second way to print "Welcome to Java" a hundred times.

Let’s Code

• Write a program that accepts two integers from the user, and find the summation for all the numbers between them (inclusive).
• The order of numbers entered by the user is not guaranteed.
Notes

• The initial-condition in a for loop can be a list of zero or more comma-separated variable declaration / assignment statements:
  
  ```java
  for (int i = 0, j = 0; (i + j < 10); i++, j++) {
    // Do something
  }
  ```

• The action-after-each-iteration in a for loop can be a list of zero or more comma-separated statements:
  
  ```java
  for (int i = 1; i < 100; System.out.println(i), i++);
  ```

Notes, cont.

• 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:

  ```java
  for ( ; ; ) {
    // Do something
  }
  ```

    Equivalent

  ```java
  while (true) {
    // Do something
  }
  ```

(a)                                (b)
Caution

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

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

Caution, cont.

• Similarly, the following loop is also (wrong?):

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

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

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

```
while (loop-continuation-condition) {
  // Loop body
}
```

```
for (; loop-continuation-condition; ) {
  // Loop body
}
```
Which Loop to Use? cont.

• 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:

![Diagram showing conversion of for loop to while loop]

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.
    • 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.
Nested Loops

- Nested loops consist of an outer loop and one or more inner loops.
- Each time, the outer loop is repeated, the inner loops are reentered.
Let’s Code

• Problem: Write a program that uses nested for loops to print a multiplication table.

break and continue
break and continue

• The **break** and **continue** keywords provide additional controls in a loop.
• The **break** keyword is used in a loop to immediately terminate the loop.
• The **continue** keyword is used in a loop to end the current iteration and program control goes to the end of the loop body.

```java
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);
    }
}
```
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);
    }
}
Task 01

(Count positive and negative numbers and compute the average of numbers)

• Write a program that reads an unspecified number of integers, determines how many positive and negative values have been read, and computes the total and average of the input values (not counting zeros). Your program ends with the input 0.
• Display the average as a floating-point number.
• Next slide shows sample runs.

Task 01 – Sample Runs

Enter an integer, the input ends if it is 0: 1 2 -1 3 0
The number of positives is 3
The number of negatives is 1
The total is 5.0
The average is 1.25

Enter an integer, the input ends if it is 0: 0
No numbers are entered except 0
Task 02

*(Find the two highest scores)*

- Write a program that prompts the user to enter the number of students and each student’s name and score, and finally displays the student with the highest score and the student with the second-highest score.

Task 03

*(Perfect number)*

- A positive integer is called a *perfect number* if it is equal to the sum of all of its positive divisors, excluding itself.
- For example, 6 is the first perfect number because \(6 = 3 + 2 + 1\). The next is \(28 = 14 + 7 + 4 + 2 + 1\).
- There are four perfect numbers less than 10,000. Write a program to find all these four numbers.
Task 04 (Optional)

*(Reverse a string)* *

• Write a program that prompts the user to enter a string and displays the string in reverse order.

• Hint: Search the web for the string `charAt` method.

Task 05 (Optional)

*(Display four patterns using loops)*

• Use nested loops that display the following patterns in four separate programs:

<table>
<thead>
<tr>
<th>Pattern A</th>
<th>Pattern B</th>
<th>Pattern C</th>
<th>Pattern D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
<td>1 1 1 1 1</td>
</tr>
</tbody>
</table>
Tasks Submission

- Submit a zipped file contains all the .java files, 1 file for each task.
- Name your zipped file as follow [Lect4_YourName.zip].
- Upload your zipped file to the Facebook group.
- Submission due: Tuesday, Sep 19 - 10:00 PM
- Late submission will not be reviewed by the instructor.
- Public solutions upload goal is to share knowledge, you can see other’s solutions, but, please, don’t cheat yourself!
- Don’t forget, all tasks should be well-documented, well-designed, and well-styled.

Test Yourself

- Answer all questions (exclude 5.5):
References:

- Liang, Introduction to Java Programming 10/e
- Eng. Asma Abdel Karim Computer Engineering Department, JU Slides.

Instructor: AbuKhleif, Mohammad Noor
Sep 2017

www.abukhleif.com