# *Programming I (420-B10-HR)*

# *Lab 14 – The for Loop*

Date assigned: Tuesday, December 1, 2015

Date due: **Tuesday, December 1, 2015**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

1. use a **for** loop in a program to execute a loop a fixed number of times using numeric and character counters;
2. use a **for** loop in a program to create a table;
3. use a **for** loop in a program to accumulate a total;
4. create an overloaded method;
5. use a nested for loop.

**Commands Used:**

**for** *(initializer; condition; updater)*

*statement***;**

**To Be Handed In:**

1. The **Lab 14 Review Quiz** should be completed in Moodle.
2. The files ***username*\_B10\_L14\_for\_loop** folder should be zipped and uploaded to Moodle. Make sure that you have reformatted all your Java files to make them properly indented.

**To Start:**

1. Download and unzip the **B10\_L14\_for\_loop** folder to your **H:\420-B10\Labs** folder. Rename it to ***username*\_B10\_L14\_for\_loop**.
2. Start **Eclipse**. Use your **H:\420-B10\Labs** folder as the workspace.
3. Create a **New Java Project** called ***username*\_B10\_L14\_for\_loop**.

# Using the for Loop

***Purpose:*** Learn to create counter loops with the **for** statement.

***To Do:***

## Create the following class in the **B10\_L14\_for\_loops** package. (Replace *username* with your logon username.)

public class *username\_B10\_L14\_for\_loop*

public static void main(String args[])

{

int j;

System.out.println("Part A First Loop:");

j = 1;

while (j <= 3)

{

System.out.println("j=" + j );

++j;

} // while()

} // main()

} // *username\_B10\_L14\_for\_loop* class

## Run the program.

***Questions:***

Fill in the following:

First value of j: \_1\_\_\_\_\_\_\_\_\_

Second value of j: \_\_\_\_\_2\_\_\_\_\_

Third value of j : \_\_3\_\_\_\_\_\_\_\_

***To Do:***

## Change the **while** statement to:

for (j = 1; j <= 3; ++j)

*Notice that the* ***for*** *statement contains both the initializer (j = 1;) and the updater (++j;). This means that you don't have to remember to do them.*

## Delete the initializer and the updater. Run the program again. Has the output changed? \_\_\_\_\_\_\_\_\_\_\_\_\_\_

***To Do:***

## Change the **for** statement to:

for (j = 1; j <= 6; ++j)

Rerun the program.

***Questions:***

Fill in the following:

Number of values of j: \_\_\_6\_\_\_\_\_\_\_ First value of j: \_\_\_\_1\_\_\_\_\_\_

Second value of j: \_\_\_2\_\_\_\_\_\_\_ Last value of j : \_\_\_\_6\_\_\_\_\_\_

***To Do:***

## Change the **for** statement to:

for (j = 3; j <= 6; ++j)

Run the program again.

***Questions:***

Fill in the following:

Number of values of j: \_\_\_\_4\_\_\_\_\_\_ First value of j: \_\_\_3\_\_\_\_\_\_\_

Second value of j: \_\_\_4\_\_\_\_\_\_\_ Last value of j : \_6\_\_\_\_\_\_\_\_\_

To Do:

## Change the loop to print the numbers 7, 8, 9, 10. Test your revised program.

## Add the following line before the **} // main()** statement.

System.out.println("Final value of j is " + j);

***Questions:***

What is the final value of *j*? \_\_\_\_11\_\_\_\_\_\_\_\_\_\_\_

Why? \_\_\_\_because it’s read as the final part of the loop, but not read into the loop itself. \_\_

***To Do:***

## Create a second **for** loop identical to the first. Change the **for** statement in the second loop to:

for (j = 6; j <= 3; ++j)

Add the following statement before the loop you just added:

System.out.println("\nPart A Second Loop:");

## Run the program.

***Questions:***

How many values of j were printed by the loop you just added? \_\_\_None\_\_\_\_\_\_

Why? \_Because the initial value of the loop is greater than the value that it stops looping at \_\_\_

## Change the second **for** statement to:

for (j = 1; j <= 7; j += 2)

and rerun the program.

***Questions:***

Fill in the following:

Number of values of j: \_\_4\_\_\_\_\_\_\_\_ First value of j: \_\_\_1\_\_\_\_\_\_\_

Second value of j: \_\_\_\_3\_\_\_\_\_\_ Last value of j : \_\_\_7\_\_\_\_\_\_\_

Difference between each j value: \_2\_\_\_\_\_\_\_\_\_

To Do:

## Change the second **for** statement to:

for (j = 1; j <= 8; j += 2)

and rerun the program.

***Questions:***

Fill in the following:

Number of values of j: \_\_\_\_4\_\_\_\_\_\_ First value of j: \_\_1\_\_\_\_\_\_\_\_

Second value of j: \_\_\_\_\_3\_\_\_\_\_ Last value of j : \_7\_\_\_\_\_\_\_\_\_

Is the answer different than the previous one. Why or why not?

\_\_No, it’s the same as last time. The numbers increase by 2, but we only increased the limiting value by 1. \_\_

To Do:

## Change the second loop to print the numbers 0, 5, 10, 15, 20. Test your changes.

# Counting Down using the for Loop

***Purpose:*** Learn to create **for** loops that count down.

To Do:

## Add the following loop to your ***username\_B10\_L14\_for\_loop*** class and run it.

System.out.println("\nPart B Counting Down");

for (j = 5; j >= 1; --j)

System.out.println("j="+ j );

To Do:

## Modify the loop you just added to count down from 100 to 90. Test your changes.

# Creating a table using the for Loop

***Purpose:*** Learn to create a table using **for** loops and review adding components to a JFrame. Learn to right and left-align columns under headings in frame text areas.

To Do:

## Open **ArithmeticTablesFrame** in the **arithmetic** package. Run the program. At the moment it doesn't do anything except offer you the option of displaying a table of squares and cubes. We want to display a table of squares and cubes if the user selects the option.

## Add a new private method called **displaySquaresAndCubes()**. Use a **setText()** and **append()** with **String.format()** to display titles aligned according to the following grid :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  | 2 |  |  |  |  |  |  |  |  |  | 3 |  |  |  |  |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |  |  |  | **S** | **q** | **u** | **a** | **r** | **e** | **s** |  | **a** | **n** | **d** |  | **C** | **u** | **b** | **e** | **s** |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | **J** |  |  |  |  |  |  | **j** |  | **s** | **q** | **u** | **a** | **r** | **e** | **d** |  |  |  |  |  |  |  |  | **j** |  | **c** | **u** | **b** | **e** | **d** |
|  |  |  | **-** | **-** |  |  |  |  |  |  | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** |  |  |  |  |  |  |  |  | **-** | **-** | **-** | **-** | **-** | **-** | **-** |
|  |  | **9** | **9** | **9** |  |  |  |  |  |  |  |  |  |  |  | **9** | **9** | **9** | **9** |  |  |  |  |  |  |  |  |  |  |  | **9** | **9** | **9** | **9** |

## Add a statement in the **actionPerformed()** method to call **displaySquaresAndCubes()** if the **btnSquaresAndCubes** button was clicked.

## Run the program and click the Squares and Cubes button to test your changes. It should display the titles.

## In the **displaySquaresAndCubes()** method, add a **for** loop to loop for **j** from 1 to 15 times and display j, j2 and j3. Use the **String.format()** method to display the output aligned under the headings. Your output should look something like:

# Using a Character Variable as the Counter in a for Loop

***Purpose:*** Learn to use a character variable as the counter in a **for** loop.

To Do:

## Add the following loop to the end of your ***username*\_B10\_L14\_for\_loop** class and run the program. (Notice that the loop control variable, **ch**, is declared as well as initialized in the loop heading. When a variable is only used as a loop control variable, this gives an efficient and clear specification.)

System.out.println("Part D Character Counters");

for (char ch = 'A'; ch <= 'C'; ++ch)

System.out.println("ch=" + ch);

***Questions:***

Fill in the following:

Number of values of ch: \_\_3\_\_\_\_\_\_\_\_ First value of ch: \_\_A\_\_\_\_\_\_\_\_

Second value of ch: \_\_ B\_\_\_\_\_\_\_\_ Last value of ch: \_\_\_ C\_\_\_\_\_\_\_

To Do:

## Modify the program to print the alphabet in both upper and lower case letters. (Use 2 loops.) Print the upper case alphabet on one line and the lower case alphabet on a second line. Test the program. Your output should look like the following:

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

# Using a for Loop to accumulate a Total

***Purpose:*** Learn to use a **for** loop to accumulate a total.

To Do:

## The **Accumulator** class contains a method to calculate the sum of the numbers from **1** to **n**. Open **Accumulator** and look at the method to see how it works.

## Create a new class called **TestAccumulator** in the **b10\_l14\_for\_loops** package which contains the following **main()** method:

public static void main(String args[])

{

Scanner input = new Scanner(System.in);

Accumulator accumulator = new Accumulator();

int n;

System.out.print("Enter number of numbers: ");

n = input.nextInt();

System.out.println("The sum of 1 to " + n + " = "

+ accumulator.accumulateSum(n));

System.out.println("\*\*\* End of TestAccumulator \*\*\*");

} // main()

## Run the program to see what it does. Where does the output come from?

The output is just all numbers between 1 and 5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Modify the program to use a while loop and a sentinel of -1 to repeat the calculation of the sum as many times as the user wants. (For example, if you found the sum of the numbers 1 to 5 and then the sum of the numbers 1 to 6, your output should look like:

Enter number of numbers (-1 to end): 5

Sum of 1 to 5 = 15

Enter number of numbers (-1 to end): 6

Sum of 1 to 6 = 21

Enter number of numbers (-1 to end): -1

\*\*\* End of TestAccumulator \*\*\*

## What happens? \_The second sum is added to the first one\_\_\_\_\_\_\_

## Use the debugger to find the problem in the **Accumulator** class to see if you can find the problem. Correct the problem and run the program again. Fill in the following table. (Use your head or a calculator to calculate the answer in the last column.)

|  |  |  |
| --- | --- | --- |
| **N** | **sum** | **n\*(n+1)/2** |
| 10 | 55 | 55 |
| 50 | 1275 | 1275 |
| 100 | 5050 | 5050 |

#### What does the formula in the last column calculate? \_The sum of all integers between two numbers\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

To Do:

## Add an overloaded **accumulateSum()** method with two integer parameters to the **Accumulator** class. It should add all the numbers between the first and second parameter values. Add the code necessary to test your new method to the **main** method. Input the starting and ending values from the user. A sample run might look like:

Enter the first number to sum: 5

Enter the last number to sum: 10

The sum of 5 to 10 is 45

## Add a method to the **Accumulator** class called **calculateAverage()** to calculate the double precision average of the numbers from *m* to *n*, where *m* and *n* are the method parameters. Use the **accumulateSum** method you created in the previous step in calculating the average. Add the necessary code to the **main()** method to print the average. A sample run of the program might look like:

Enter number of numbers (-1 to end): 15

Sum of 1 to 15 = 120

Enter number of numbers (-1 to end): -1

Enter the first number to sum: 5

Enter the last number to sum: 10

The sum of 5 to 10 is 45

The average of 5 to 10 is 7.5

\*\*\* End of TestAccumulator \*\*\*

# Review Quiz

## Complete the **Moodle** **Lab 14 - Review Quiz**.