Lab 05

Repetition Control Structures-II

Objective:

This lab will give you practical implementation of different types of Repetition Control Structures.

Activity Outcomes:

On completion of this lab student will be able

- Write a for loop
- Nested Loops
- Use break and continue statements in loops

Instructor Note:

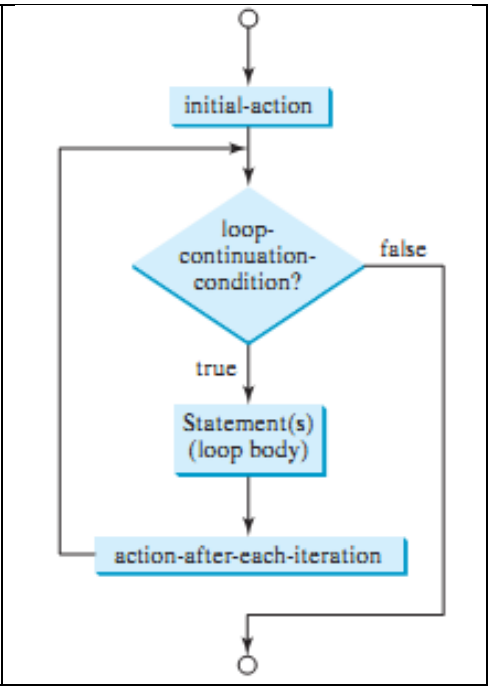
As a pre-lab activity, read Chapter 05 from the text book "Java How to Program, Deitel, P. & Deitel, H., Prentice Hall, 2019".

1) Useful Concepts The `for` Loop

A **`for`** loop has a concise syntax for writing loops.

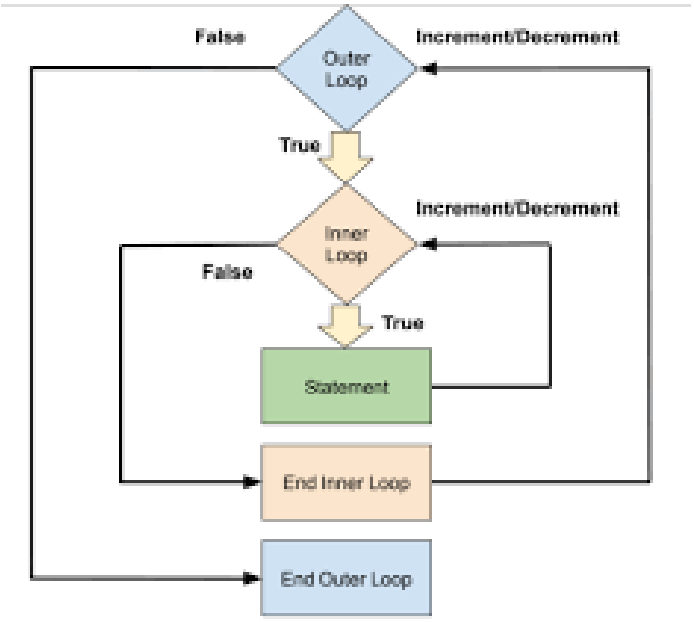
Syntax

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



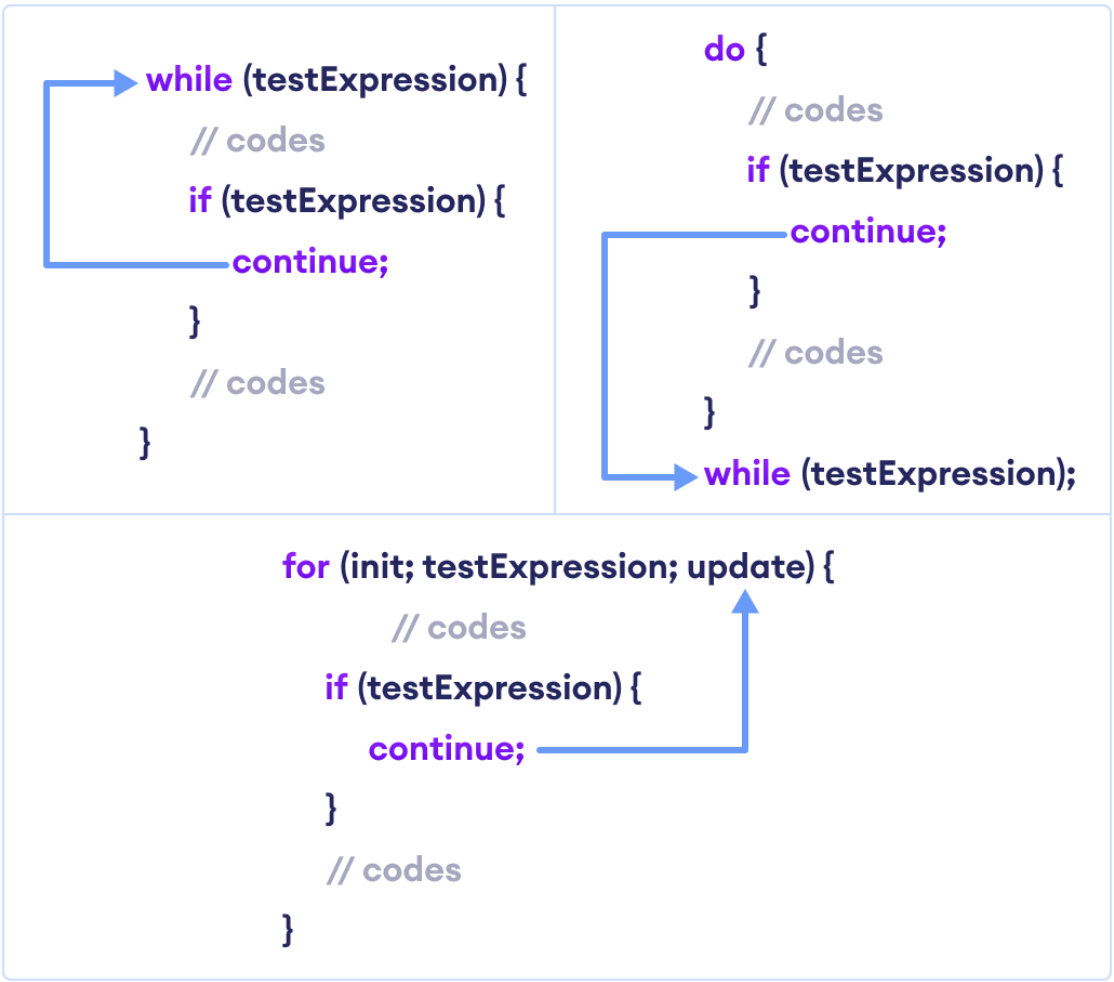
Nested Loops

A loop can be nested inside another loop. 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 and started a new.



The `break` and `continue` Statements

The **`break`** statement causes a loop to terminate early. The **`continue`** statement causes a loop to stop its current iteration and begin the next one.



2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
Activity 1	*10 mins*	*Midum*	*CLO-5*
Activity 2	*10 mins*	*Midum*	*CLO-5*
Activity 3	*10 mins*	*Midum*	*CLO-5*
Activity 4	*10 mins*	*Midum*	*CLO-5*
Activity 5	*10 mins*	*Medium*	*CLO-5*

Activity 1:

This activity demonstrate the working of `for` loop. The following program calculates sum of five numbers entered by the user. Suppose the input is 2 3 4 5 0.

Solution:

```
import java.util.Scanner;
public class Activity1 {
      public static void main(String[] args) {
            Scanner input = new Scanner(System.in);
            int number, sum = 0, count;
            for (count = 0; count < 5; count++) {
                  number = input.nextInt();
                  sum += number;
            }
      System.out.println("sum is " + sum);
      }
}
```

Output

```
Sum is 14
```

Activity-2:

This program uses nested for loops to display a multiplication table

Solution:

```
public class Activity2{
      public static void main(String[] args){
        System.out.println("           Multiplication Table");
        // Display the number title
        System.out.print("    ");
        for (int j = 1; j <= 9; j++)
           System.out.print("   " + j);
      System.out.println("\n———————————————————————————————————————
      ");
     // Display table body
      for (int i = 1; i <= 9; i++){
         System.out.print(i + " | ");
        for (int j = 1; j <= 9; j++) {
            // Display the product and align properly
            System.out.printf("%4d", i * j);
        }
        System.out.println();
      }
    }
```

```
}
```

Output

```
           Multiplication Table
       1   2   3   4   5   6   7   8   9
------------------------------------------
1 |    1   2   3   4   5   6   7   8   9
2 |    2   4   6   8  10  12  14  16  18
3 |    3   6   9  12  15  18  21  24  27
4 |    4   8  12  16  20  24  28  32  36
5 |    5  10  15  20  25  30  35  40  45
6 |    6  12  18  24  30  36  42  48  54
7 |    7  14  21  28  35  42  49  56  63
8 |    8  16  24  32  40  48  56  64  72
9 |    9  18  27  36  45  54  63  72  81
```

Activity-3:

This program demonstrate the effect of using `break` in a loop.

Solution:

```
public class Activity3{
  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);
  }
}
```

Output

```
The number is 14
The sum is 105
```

Activity 4:

This program demonstrate the effect of using `continue` statement in a loop.

Solution:

```
public class Activity4{
  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);
  }
}
```

Output

```
The sum is 189
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

The index of the maximum of a sequence: *A sequence consists of integer numbers and ends with the number 0. Determine the index of the largest element of the sequence. If the highest element is not unique, print the index of the first of them.*

```
Input:1 2 3 2 1 0                    Output:3
```

Lab Task 2

The number of even elements of the sequence: *Determine the number of even elements in the sequence ending with the number 0.*

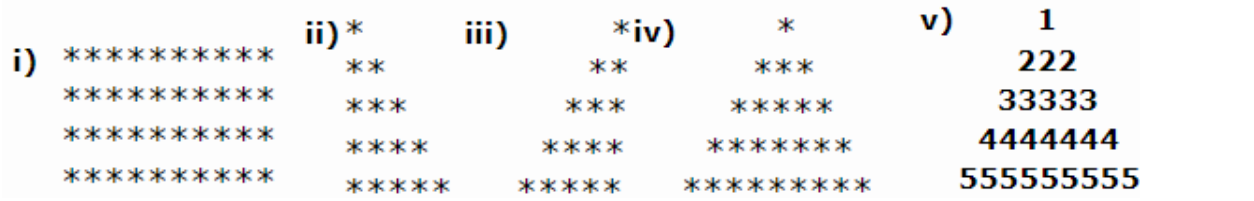
Lab Task 3

The number of elements that are greater than the previous one: *A sequence consists of integer numbers and ends with the number 0. Determine how many elements of this sequence are greater than their neighbours above.*

```
Input: 1 5 2 4 3 0                   Output: 2
```

Lab Task 4

Write a program to print following :



The program asks the user to enter the number which pattern he/she wants to print. The loop should ask the user whether he or she wishes to perform the operation again. If so, the loop should repeat; otherwise it should terminate.

Lab Task 5

Write a program that prompts the user to enter the year and first day of the year and displays the calendar table for the year on the console. For example, if the user entered the year 2013, and 2 for Tuesday, January 1, 2013, your program should display the calendar for each month in the year, as follows:

January 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

December 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				