Final Term Exam Summer 2020

Object Oriented Programming

Submitted By: Shahid Ali
ID# 13942
7th Semester
BS(SE)

Instructor: M.Ayub Khan
(Lecturer)

Date: September 30, 2020

Question No: 01

How many variables are being supported by java justify your answer with the help java coded example for each variable?

Answer:

Variable is name of reserved area allocated in memory. In other words, it is a name of memory location. It is a combination of "vary + able" that means its value can be changed.

There are three types of variables in Java:

- local variable
- instance variable
- static variable n be changed.

1. Local Variable:

A variable declared inside the body of the method is called local variable. You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.

A local variable cannot be defined with "static" keyword.

Example to understand the Local variables in java:

```
class Question1 //main class
{
        void method()
        {
                int n=90;//local variable
        }
}//end of main class
```

2. Instance Variable:

A variable declared inside the class but outside the body of the method, is called instance variable. It is not declared as static.

It is called instance variable because its value is instance specific and is not shared among instances.

Example to understand the Instance variables in java:

```
class Question1 //main class
{
        int data=50;//instance variable
        void method()
        {
                int n=90;//local variable
        }
}//end of main class
```

3. Static variable:

A variable which is declared as static is called static variable. It cannot be local. You can create a single copy of static variable and share among all the instances of the class. Memory allocation for static variable happens only once when the class is loaded in the memory.

Example to understand the Static variables in java:

```
class Question1  //main class
{
        int data=50;//instance variable
        static int m=100;//static variable
        void method()
        {
        int n=90;//local variable
        }
}//end of main class
```

Question No: 02

Why "If" is used in java justify your answer with the help java coded example and explain in detail?

Answer:

The if keyword is used to check if a condition is true or not. If it is true, then the specified code inside the curly braces are executed.

In Java, the syntax of the "if" statement is:

```
if (expression)
{
   // statements
}
```

Here expression is a boolean expression. A boolean expression returns either true or false.

- if the expression is evaluated to true, statement(s) inside the body of if (statements inside parenthesis) are executed
- if the expression is evaluated to false, statement(s) inside the body of if are skipped from execution

Example: Java if Statement

```
class IfStatement
{
   public static void main(String[] args)
   {
      int number = 10;
      // checks if number is greater than 0
      if (number > 0)
      {
         System.out.println("The number is positive.");
      }
      System.out.println("This statement is always executed.");
   }
}
```

Output:

The number is positive.
This statement is always executed.

- In the above example, we have a variable named number. Here, the test expression checks if the number is greater than 0 (number > 0).
- Since the number is greater than 0. So the test expression evaluates to true. Hence code inside the body of if is executed.
- Now, change the value of the number to a negative integer. Let's say -5.

  int number = -5;
- If we run the above program with the new value of the number, the output will be:

  This statement is always executed.
- Here, the value of number is less than 0. So, the test expression number > 0 evaluates to false. Hence, the body of if is executed.

.

Question No: 03

Why "if else if" is used in java justify your answer with the help java coded example and explain in detail?

Answer:

In Java, we have an if...else...if ladder, that can be used to execute one block of code among multiple other blocks.

```
if (expression1)
{
  // codes
}
else if(expression2)
{
  // codes
}
else if (expression3)
{
  // codes
}
else {
  // codes
}
```

Here, if statements are executed from the top towards the bottom. As soon as the test expression is true, codes inside the body of that the if statement is executed. Then, the control of the program jumps outside the if-else-if ladder.

If all test expressions are false, codes inside the body of else is executed.

Example: Java if..else..if Statement

```
class Ladder
{
        public static void main(String[] args)
        {
                int number = 0;
                if (number > 0)          // checks if number is greater than 0
                {
                        System.out.println("The number is positive.");
                }
                else if (number < 0)    // checks if number is less than 0
                {
                        System.out.println("The number is negative.");
                }
                else
                {
                        System.out.println("The number is 0.");
                 }
        }
}
```

Output:
The number is 0.

In the above example, we are checking whether the number is positive, negative or zero. Here, we have two test expressions:

- number > 0 - checks if the number is greater than 0
- number < 0 - checks if the number is less than 0

Here, the value of number is 0. So both the test expression evaluates to false. Hence the statement inside the body of else is executed.

Question No: 04

What are loops, why they are used in java and how many types of loops are being supported by java explain in detail?

Answer:

Loops in java is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true.

Java provides three ways for executing the loops. While all the ways provide similar basic functionality, they differ in their syntax and condition checking time.

1.while loop

A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.

Syntax :

```
while (boolean condition)
{
        loop statements...
}
```

- While loop starts with the checking of condition. If it evaluated to true, then the loop body statements are executed otherwise first statement following the loop is executed. For this reason it is also called Entry control loop
- Once the condition is evaluated to true, the statements in the loop body are executed. Normally the statements contain an update value for the variable being processed for the next iteration.
- When the condition becomes false, the loop terminates which marks the end of its life cycle.

Example:

```
// Java program to illustrate while loop
class whileLoopDemo
{
        public static void main(String args[])
        {
                int x = 1;
                while (x <= 4)          // Exit when x becomes greater than 4
                {
                        System.out.println("Value of x:" + x);  // Increment the value of x for
                        x++;    // next iteration
                }
        }
}
```

Output:

```
Value of x:1
Value of x:2
Value of x:3
Value of x:4
```

2. for loop: for loop provides a concise way of writing the loop structure. Unlike a while loop, a for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.

Syntax:

```
for (initialization condition; testing condition; increment/decrement)
{
        statement(s)
}
```

- **Initialization condition:** Here, we initialize the variable in use. It marks the start of a for loop. An already declared variable can be used or a variable can be declared, local to loop only.
- **Testing Condition:** It is used for testing the exit condition for a loop. It must return a boolean value. It is also an Entry Control Loop as the condition is checked prior to the execution of the loop statements.
- **Statement execution:** Once the condition is evaluated to true, the statements in the loop body are executed.
- **Increment/ Decrement:** It is used for updating the variable for next iteration.
- **Loop termination:**When the condition becomes false, the loop terminates marking the end of its life cycle.

Example:

```
// Java program to illustrate for loop.
class forLoopExample
{
        public static void main(String args[])
        {
        for (int x = 2; x <= 4; x++)                // for loop begins when x=2 and runs till x <=4
        System.out.println("Value of x:" + x);
        }
}
```

Output:

Value of x:2
Value of x:3
Value of x:4

---

3. do while

do while loop is similar to while loop with only difference that it checks for condition after executing the statements, and therefore is an example of Exit Control Loop.

Syntax:

```
do
{
        statements..
}
while (condition);
```

- do while loop starts with the execution of the statement(s). There is no checking of any condition for the first time.
- After the execution of the statements, and update of the variable value, the condition is checked for true or false value. If it is evaluated to true, next iteration of loop starts.
- When the condition becomes false, the loop terminates which marks the end of its life cycle.
- It is important to note that the do-while loop will execute its statements atleast once before any condition is checked, and therefore is an example of exit control loop.

Example:

```
// Java program to illustrate do-while loop
class dowhileloopDemo
{
        public static void main(String args[])
        {
                int x = 21;
                do
                {
                // The line will be printed even
                //if the condition is false
                System.out.println("Value of x:" + x);
                x++;
                }
                while (x < 20);
        }
}
```

Output:
Value of x: 21

Question No: 05

Write 3's table in decremented form in java which takes input from user write java coded program and explain in detail?

Answer:

- This is a Java Program to Print Multiplication Table for any Number.
- Enter any integer number as input of which you want multiplication table. After that we use for loop from one to ten to generate multiplication of that number.

```
import java.util.Scanner;
public class Multiplication_Table
{
	public static void main(String[] args)
	{
		Scanner obj = new Scanner(System.in);			// making obj name object
		System.out.print("Which table you want to print : ");		//showing Massege to the user
		int n=obj.nextInt();					// store integer value from user in n variable
		for(int i=1; i <= 10; i++)				//for loop begins when i=1 and runs till  i<=10
		{
			System.out.println(n+" * "+i+" = "+n*i);  // will display table table
		}
	}
}
```

Output:

```
Which table you want to print : 3
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```