Name. **Bilal Ahmad**
ID. **15089**
Subject. **Object oriented programming**
Dept. **BS(SE)**

Q1:
Ans: Variable: A variable is a container which holds the value while the Java program is executed. A variable is assigned with a data type. Variable is a name of memory location. There are three types of variables in java: local, instance and static**.** **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.
int data=50;//Here data is variable

Types Of Variable:

There are three types of variables in Java:

- local variable
- instance variable
- static variable

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: public class Dog

```
{
  public void putAge()
{
    int age = 0; //local variable
    age = age + 6;
    System.out.println("Dog age is : " + age);
 }
  public static void main(String args[]){
   Dog d = new Dog();
```

```
        d.putAge();
  }
}
```

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:

```
    public class Record{


    public String name;// this instance variable is visible for any
child class.
     private int age;// this instance age variable is visible in
Record class only.
 public Record (String RecName)
    {
       name = RecName;
    }
     public void setAge(int RecSal)
    {
       age = RecSal;
    }
      public void printRec()
    {
       System.out.println("name : " + name ); //
    print the value for "name"
       System.out.println("age :" + age); //prints the
    value for "age"
    }
      public static void main(String args[])
    {
       Record r = new Record("Khan");
       r.setAge(23);
       r.printRec();
    }
}
```

3)Static Variable: The **static variable** can be used to refer to the common property of all objects (which is not unique for each object), for **example**, the company name of employees, college name of students, etc. The **static variable** gets memory only once in the class area at the time of class loading.

Example:

```
       //Java Program to demonstrate the use of an instance variable
```

```
        //which get memory each time when we create an object of th
e class.
class Counter{
int count=0;//will get memory each time when the instance is creat
ed
        Counter(){
     count++;//incrementing value
        System.out.println(count);
        }
        public static void main(String args[]){
        //Creating objects
        Counter c1=new Counter();
        Counter c2=new Counter();
        Counter c3=new Counter();
        }
        }
```

Q2:

Ans: The Java if statement is the most simple decision-making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e if a certain condition is true then a block of statement is executed otherwise not.

Control falls into the if block.

The flow jumps to Condition.

Condition is tested.

If Condition yields true, goto Step

If Condition yields false, goto Step

The if-block or the body inside the if is executed.

Flow steps out of the if block.

```
if(condition)
  statement1;
  statement2;

// Here if the condition is true, if block
// will consider only statement1 to be inside
// its block
```

Example:

```
// Java program to illustrate If statement
```

```
class IfDemo {

    public static void main(String args[])

    {
int i = 10;
if (i < 15)
System.out.println("10 is less than 15");
// This statement will be executed
// as if considers one statement by default
System.out.println("Outside if-block");

    }
}
```

Q3:

Ans:if-else-if Statement:

When we need to execute a set of statements based on a condition then we need to use control flow statements. For example, if a number is greater than zero then we want to print “Positive Number” but if it is less than zero then we want to print “Negative Number”. In this case we have two print statements in the program, but only one print statement executes at a time based on the input value. We will see how to write such type of conditions in the java program using control statements.

if-else-if statement is used when we need to check multiple conditions. In this statement we have only one “if” and one “else”, however we can have multiple “else if”. It is also known as if else if ladder. This is how it looks:

Example:

```
public class IfElseIfExample {
public static void main(String args[]){
       int num=1234;
       if(num <100 && num>=1) {
         System.out.println("Its a two digit number");
       }
       else if(num <1000 && num>=100) {
         System.out.println("Its a three digit number");
       }
       else if(num <10000 && num>=1000) {
         System.out.println("Its a four digit number");
       }
       else if(num <100000 && num>=10000) {
             System.out.println("Its a five digit number");
       }
       else {
           System.out.println("number is not between 1 & 99999");
       }
    }
}
```

Q4:

Ans: Loops are used to execute a set of statements repeatedly until a particular condition is satisfied. In Java we have three types of basic loops: for, while and do-while. In this tutorial we will learn how to use "for loop" in Java.

Looping in programming languages is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true. ... while loop: A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition.

In Java there are three primary types of loops:-

1. for loop
2. Enhanced for loop
3. while loop
4. do-while loop

1. For loop in Java

Java for loop consists of 3 primary factors which define the loop itself. These are the initialization statement, a testing condition, an increment or decrement part for incrementing/decrementing the control variable.

Example:

```
package com.dataflair.loops;
import java.io. * ;
public class ForLoop {
 public static void main(String[] args) {
   int i;
   for (i = 0; i <= 10; i++) {
    System.out.println("Studying for loops at DataFlair");
    System.out.println("Value of i = " + i);
   }
  }
 }
```

Enhanced for loop in Java:

This is similar to a for loop except that it has some enhanced features. It can be used to iterate over the elements of a collection without knowing the index of each element. You also need not know the size of the collection.

However there are some limitations to this enhanced for loop.This loop object is immutable which means the values which are retrieved during the execution of the loop are read only. You cannot change the values in the collection while using this loop which is easily done by other loops.

Example:

```
package com.dataflair.loops;

import java.io. * ;

public class EnhancedFor {

 public static void main(String[] args) {

  String array[] = {

   "DataFlair",

   "Java",

   "Python"

  };

  for (String a: array) {

   System.out.println(a);

  }

 }

}
```

While loop in Java:

While loops are very important as we cannot know the extent of a loop everytime we define one. For example if we are asked to take a dynamic collection and asked to iterate through every element, for loops would be impossible to use because we do not know the size of the collection. Then we would have to use an enhanced for loop or a while loop.

A while loop iterates through a set of statements till its boolean condition returns false. As long as the condition given evaluates to true, the loop iterates.

The condition of the loop structure is checked at first and then the control proceeds into the loop structure only if the condition evaluates to true. Hence it is called an entry-controlled loop. The body of the loop generally contains a variable which controls the boolean condition mentioned.

Example:

```
package com.dataflair.loops;

import java.io. * ;

public class WhileLoop {

 public static void main(String[] args) {

   int i = 0;

   while (i < 5) {

     System.out.println("Learning Java at DataFlair");

     System.out.println("The value of i is = " + i);

     i++;

   }

   System.out.println("The value of i became " + i + " that is why it broke out of the loop");

 }


}
```

do while loop in Java:

Java do while loop executes the statement first and then checks for the condition.Other than that it is similar to the while loop. The difference lies in the fact that if the condition is true at the starting of the loop the statements would still be executed, however in case of while loop it would not be executed at all.

This is an exit-controlled loop because of the fact that it checks the condition after the statements inside it are executed.

Example:

```
package com.dataflair.loops;

import java.io. * ;

public class DoWhileLoop {

 public static void main(String[] args) {
  int i = 0;
  do {
    i++;
    System.out.println("Learning Java at DataFlair");
    System.out.println("The value of i is " + i);
 }
  while ( i != 5 );
 }
}
```

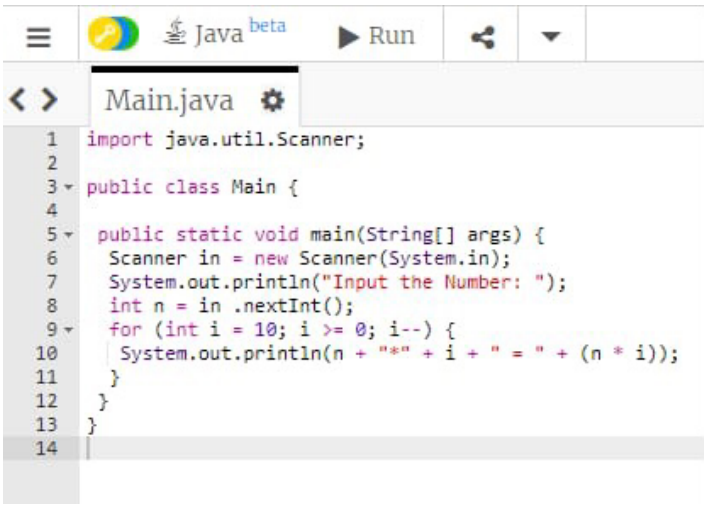
Q5:

Ans: import java.util.Scanner;

```
public class Main {

public static void main(String[] args) {
 Scanner in = new Scanner(System.in);
 System.out.println("Input the Number: ");
 int n = in .nextInt();
 for (int i = 1; i <= 10; i- -) {
  System.out.println(n + "*" + i + " = " + (n * i));
 }
}
}
```

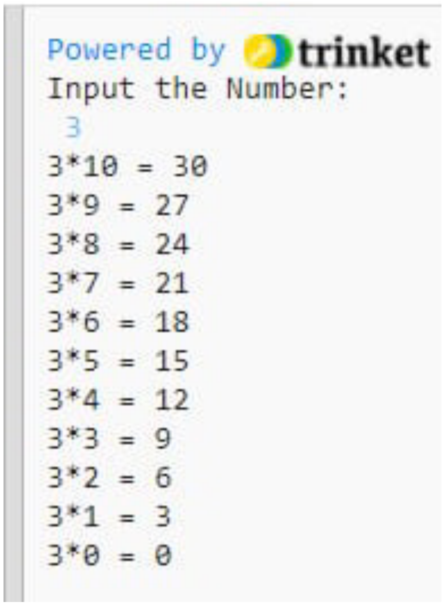
-



```
import java.util.Scanner;

public class Main {

 public static void main(String[] args) {
  Scanner in = new Scanner(System.in);
  System.out.println("Input the Number: ");
  int n = in .nextInt();
  for (int i = 10; i >= 0; i--) {
   System.out.println(n + "*" + i + " = " + (n * i));
  }
 }
}
```

- 

```
Input the Number:
 3
3*10 = 30
3*9 = 27
3*8 = 24
3*7 = 21
3*6 = 18
3*5 = 15
3*4 = 12
3*3 = 9
3*2 = 6
3*1 = 3
3*0 = 0
```