



IQRA NATIONAL UNIVERSITY

NAME: RAIMA ID:14321

SUBJECT: OOP DEP: BS(SE)

SUBMITTED TO: SIR AYUB KHAN

QUESTION:01

Q1. How many variables are being supported by java justify your answer with the help java coded example for each variable?

ANSWER:

Three types of variables are being supported by java ,

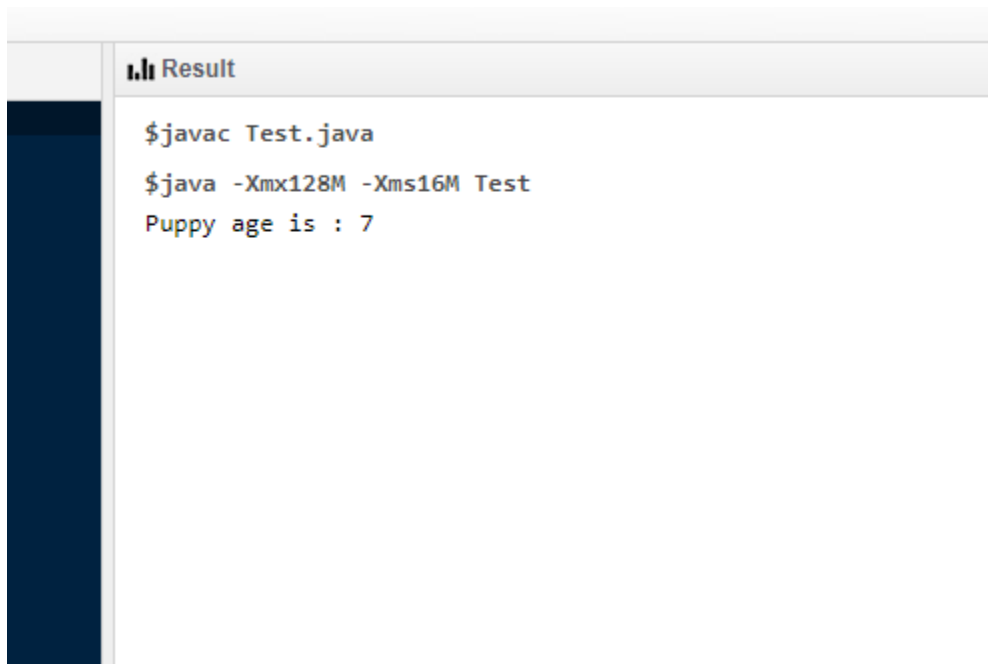
1. Local variables
2. Instant varaibale
3. Class/static variablr

LOCAL VARIABLES:

1. Local variables are declared in methods, constructors, or blocks.
2. Local variables are created when the method, constructor or block is entered and the variable will be destroyed once it exits the method, constructor, or block.
3. Access modifiers cannot be used for local variables.
4. Local variables are visible only within the declared method, constructor, or block.
5. Local variables are implemented at stack level internally.
6. There is no default value for local variables, so local variables should be declared and an initial value should be assigned before the first use.

```
1 public class Test {
2     public void pupAge() {
3         int age = 0;
4         age = age + 7;
5         System.out.println("Puppy age is : " + age);
6     }
7
8     public static void main(String args[]) {
9         Test test = new Test();
10        test.pupAge();
11    }
12 }
```

OUTPUT:



```
Result
$javac Test.java
$java -Xmx128M -Xms16M Test
Puppy age is : 7
```

INSTANT VARIABLE:

1. Instance variable in Java is used by Objects to store their states.
2. Variables that are defined without the STATIC keyword and are Outside any method declaration are Object-specific and are known as instance variables.
3. They are called so because their values are instance specific and are not shared among instances.
4. Instance variables are declared in a class, but outside a method, constructor or any block.
5. When a space is allocated for an object in the heap, a slot for each instance variable value is created.
6. Instance variables are created when an object is created with the use of the keyword 'new'; and destroyed when the object is destroyed.
7. Instance variables hold values that must be referenced by more than one method, constructor or block, or essential parts of an object's state that must be present throughout the class.
8. Instance variables can be declared in class level before or after use.
9. Access modifiers can be given for instance variables.

```

1 import java.io.*;
2 public class Employee {
3
4     public String name;
5
6     private double salary;
7
8     public Employee (String empName) {
9         name = empName;
10    }
11
12    public void setSalary(double empSal) {
13        salary = empSal;
14    }
15
16    public void printEmp() {
17        System.out.println("name : " + name );
18        System.out.println("salary :" + salary);
19    }
20
21    public static void main(String args[]) {
22        Employee empOne = new Employee("Raima");
23        empOne.setSalary(20000);
24        empOne.printEmp();
25    }
26 }

```

OUTPUT:

Result	
\$javac Employee.java	\$javac Employee.java
\$java -Xmx128M -Xms16M Employee	\$java -Xmx128M -Xms16M Employee
name : Raima	name : Raima
salary :20000.0	salary :20000.0

CLASS/STATIC VARIABLE:

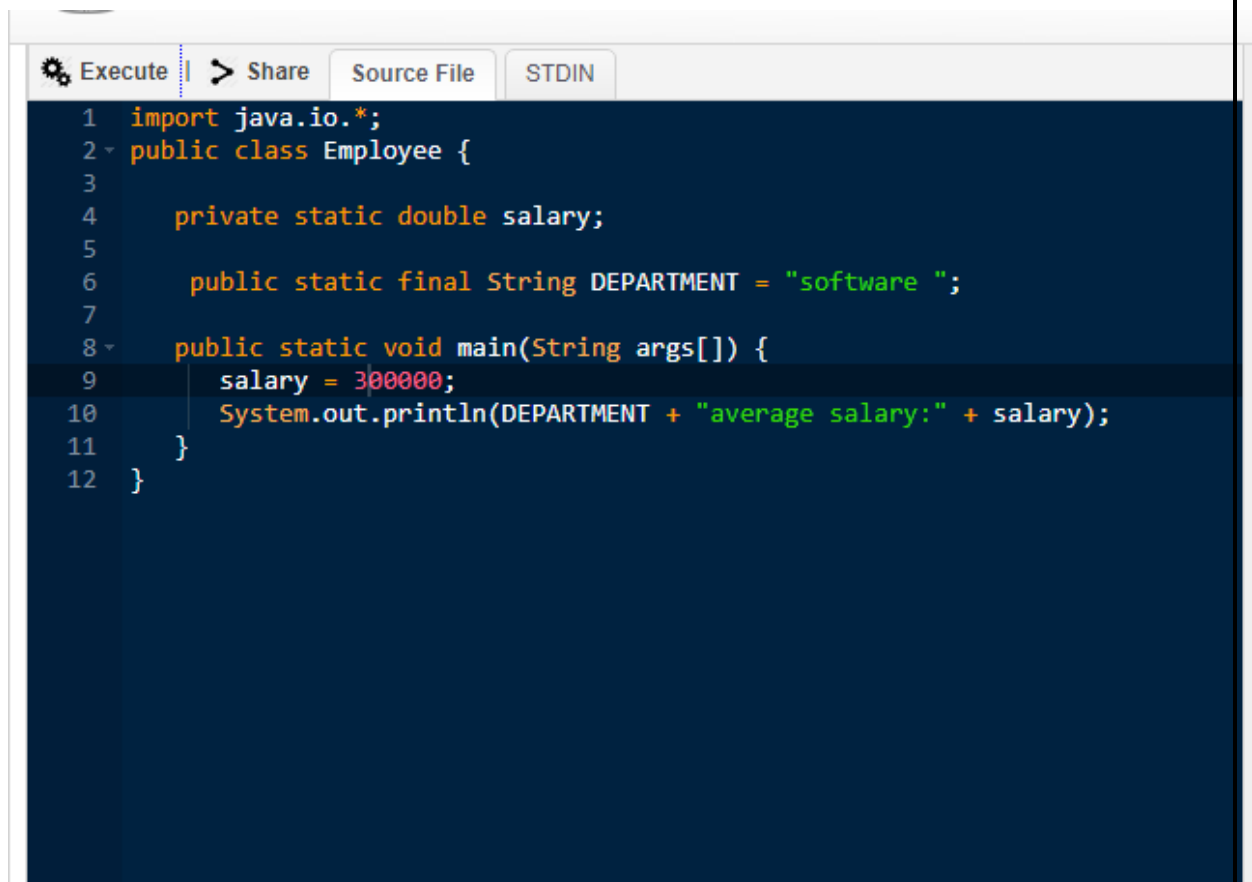
Static variables are rarely used other than being declared as constants. Constants are variables that are declared as public/private, final, and static. Constant variables never change from their initial value.

☐ Static variables are stored in the static memory. It is rare to use static variables other than declared final and used as either public or private constants.

☐ Static variables are created when the program starts and destroyed when the program stops.

☐ Visibility is similar to instance variables. However, most static variables are declared public since they must be available for users of the class. Class variables also known as static variables are declared with the static keyword in a, but outside a method, constructor or a block.

☐ There would only be one copy of each class variable per class, regardless of how many objects are created from it.



```
Execute | > Share | Source File | STDIN
1 import java.io.*;
2 public class Employee {
3
4     private static double salary;
5
6     public static final String DEPARTMENT = "software ";
7
8     public static void main(String args[]) {
9         salary = 300000;
10        System.out.println(DEPARTMENT + "average salary:" + salary);
11    }
12 }
```

OUTPUT:

```
Result
$javac Employee.java
$java -Xmx128M -Xms16M Employee
software average salary:300000.0
```

QUESTION:02

Q2. Why “If” is used in java justify your answer with the help java coded example and explain in detail?

ANSWER:

The if statement executes a certain section of code if the test expression is evaluated to true . However, if the test expression is evaluated to false , it does nothing. In this case, we can use an optional else block. This is known as the if-then-else statement in Java. java has the following conditional statements: Use if to specify a block of code to be executed, if a specified condition is true. Use else to specify a block of code to be executed, if the same condition is false. Use else if to specify a new condition to test, if the first condition is false.

Online Java Compiler IDE

For Multiple Files, Custom Library and File Read/Write, use our new - [Advanced Java IDE](#)

```
1 public class IfExample {
2 public static void main(String[] args) {
3     //defining an 'age' variable
4     int age=20;
5     //checking the age
6     if(age>18){
7         System.out.print("Age is greater than 18");
8     }
9 }
10 }
11
```

Execute Mode, Version, Inputs & Arguments

OUTPUT:

Commandline Arguments

Execute

Result
CPU Time: 0.14 sec(s), Memory: 30276 kilobyte(s) compiled and executed in 0.77 sec(s)

```
Age is greater than 18
```

EXPLANATION:

```
public static void main(String[] args) {
```

In this line of code we are defining an 'age' variable.

```
int age=20;
```

In this line of code checking the age

And we get output "age is greater than 18."

QUESTION:03

Q3. Why "if else if" is used in java justify your answer with the help java coded example and explain in detail?

ANSWER:

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.

```
Execute | Share | Source File | STDIN
1  /**
2   * abcd
3   */
4  public class abcd {
5
6      public static void main(String[] args) {
7          int time = 22;
8          if (time < 10) {
9              System.out.println("Reads boosk");
10         }
11         else if (time < 20)
12         {
13             System.out.println("Go for playing.");
14         } else
15         {
16             System.out.println("GO for shops");
17         }
18     }
19 }
20 }
```

OUTPUT:

```
Fork Project Edit Settings
Result
$javac abcd.java
$java -Xmx128M -Xms16M abcd
GO for shops
```


EXPLANATION:

In this program I am using "if else if". I have declare an integer name "time" and I have use if else if statement for it. If the time is less then "10 " it will display "read book",if this the first condition goes wrong the compiler will come to the second condition which is "else if (time <20)" which means if time is less then 20 it will display "go for playing",if these both condition goes wrong it will display "go for shops".

QUESTION:04

Q4. What are loops, why they are used in java and how many types of loops are being supported by java explain in detail?

ANSWER:

LOOP:

In computer science, a loop is a programming structure that repeats a sequence of instructions until a specific condition is met. Programmers use loops to cycle through values, add sums of numbers, repeat functions, and many other things.

WHY THEY ARE USED :

Looping in programming languages 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.

TYPES OF LOOP:

There are 3 types of loop ,following are;

1. While loop
2. For loop
3. Do while loop

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.

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.

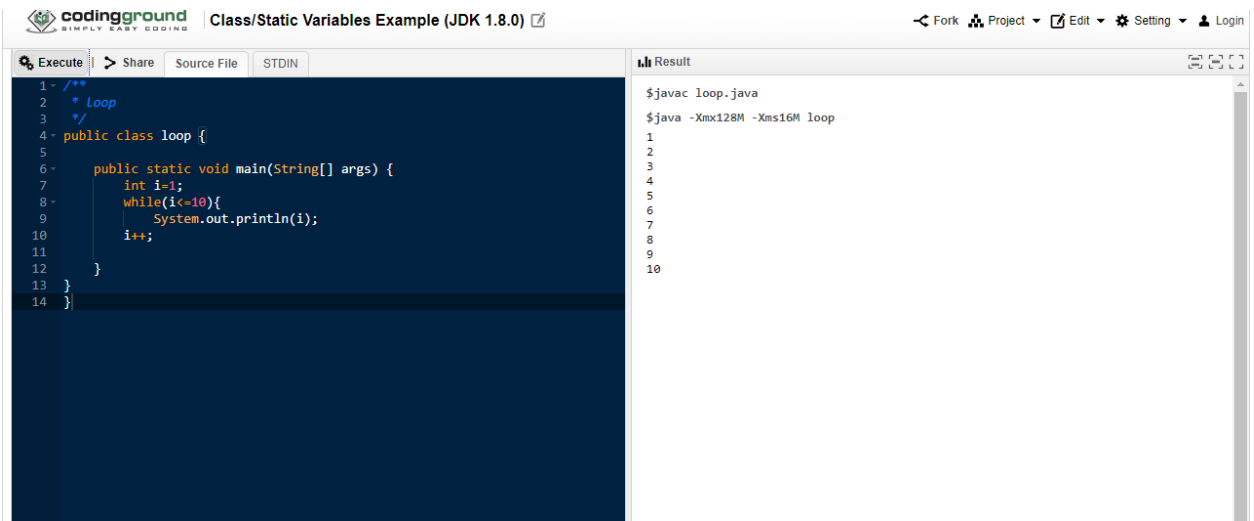
DO WHILE LOOP:

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 at least once before any condition is checked, and therefore is an example of exit control loop.



The screenshot shows a coding environment with a source code editor on the left and a result window on the right. The source code is as follows:

```
1- /*
2-  * Loop
3-  */
4- public class loop {
5-
6-     public static void main(String[] args) {
7-         int i=1;
8-         while(i<=10){
9-             System.out.println(i);
10-            i++;
11-        }
12-    }
13- }
14- }
```

The result window shows the output of the program:

```
$javac loop.java
$java -Xmx128M -Xms16M loop
1
2
3
4
5
6
7
8
9
10
```

QUESTION:05

Q5. Write 3's table in decremented form in java which takes input from user write

java coded program and explain in detail?

ANSWER:

FIRST TABLE:

```
1 import java.util.Scanner;
2
3 public class Exercise7 {
4
5     public static void main(String[] args) {
6         Scanner in = new Scanner(System.in);
7
8         System.out.print("Input a number: ");
9         int num1 = in.nextInt();
10
11        for (int i=0; i< 10; i++){
12            System.out.println(num1 + " x " + (i+1) + " = " +
13                (num1 * (i+1)));
14        }
15    }
16 }
```

```
Input a number: 8
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72
8 x 10 = 80
```

EXPLANATION:

In this program first of all I am using "scanner in" which is use to take input from the user ,then I have declare the integer n, then I have use 'for loop' which is "for (i=1;i<=10;i++); this will prent a table

2ND TABLE:

```
1 import java.util.Scanner;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Scanner in = new Scanner(System.in);
7         System.out.println("Input the Number: ");
8         int n = in.nextInt();
9         for (int i = 1; i <= 10; i++) {
10            System.out.println(n + "*" + i + " = " + (n * i));
11        }
12    }
13 }
```

OUTPUT:

```
Input the Number:
6
6*1 = 6
6*2 = 12
6*3 = 18
6*4 = 24
6*5 = 30
6*6 = 36
6*7 = 42
6*8 = 48
6*9 = 54
6*10 = 60
```

EXPLANATION:

3RD TABLE:

```
public class MultiplicationTable {  
  
    public static void main(String[] args) {  
  
        int num = 9, i = 1;  
        while(i <= 10)  
        {  
            System.out.printf("%d * %d = %d \n", num, i, num * i);  
            i++;  
        }  
    }  
}
```

OUTPUT:

Output

```
9 * 1 = 9  
9 * 2 = 18  
9 * 3 = 27  
9 * 4 = 36  
9 * 5 = 45  
9 * 6 = 54  
9 * 7 = 63  
9 * 8 = 72  
9 * 9 = 81  
9 * 10 = 90
```

EXPLANATION:

In the above program, unlike a for loop, we have to increment the value of i inside the body of the loop.

Though both programs are technically correct, it is better to use for loop in this case. It's because the number of iteration (from 1 to 10) is known.