**Day-10**

1. How to create an object in java?

To create an object, we have to use the “new” operator to create an object.

There is no “delete” operator in java because destruction of useless objects is the responsibility of the garbage collector.

class ToCreateObj{

    int n;

    String name;

    public static void main(String[] args) {

        int n = 9;

        ToCreateObj obj = new ToCreateObj();

    }

}

1. What is the use of a new keyword in Java?

The use of a new keyword in java is creating object.

1. What is the difference between instance variable and local variable?

Instance Variable

* Instance variables will be created at the time of object creation and destroyed at the time of object destruction hence the scope of instance variables is exactly the same as scope of objects.
* Instance variables will be stored on the heap as the part of the object.
* Instance variables should be declared within the class directly but outside of any method or block or constructor.

Local Variable

* Local variables will be stored inside the stack.
* The local variables will be created as part of the block execution in which it is declared and destroyed once that block execution completes. Hence the scope of the local variables is exactly the same as the scope of the block in which we declared.

1. In which area memory is allocated for instance variable and local variable?

Memory for instance variables (also known as member variables) is allocated in the heap memory. Each instance of a class has its own set of instance variables, and memory for these variables is allocated when the object is created using the `new` keyword.

Local variables, on the other hand, are allocated memory on the stack. Local variables are variables declared within a method, constructor, or block, and their memory allocation is handled by the Java Virtual Machine (JVM) during the execution of the program. Once the method, constructor, or block in which the local variable is declared completes its execution, the memory allocated for the local variable is reclaimed by the JVM.

1. What is method overloading?

Method overloading enables several methods to use the same name but have distinct signatures, where the signature might vary based on the quantity, nature, or combination of input arguments.

In the 'C' language we can't take 2 methods with the same name and different types. If there is a change in argument type compulsory we should go for a new method name.

Example :

abs() for int datatype

labs() for long datatype

fabs() for float datatype

Lack of overloading in "C" increases complexity of the programming.

But in java we can take multiple methods with the same name and different

argument types.

abs(int) for int datatype

abs(long) for long datatype

abs(float) for float datatype