**STATIC KEYWORD ASSIGNMENT**

**1. Why do we need static keyword in Java Explain with an example?**

**Ans:** The static keyword in Java is used to indicate that a variable, method, or inner class is a class variable (belongs to the class, rather than to an instance of the class).

Example: class Counter {

static int count = 0;

static void increment() {

count++;

}

}

public class Main {

public static void main(String[] args) {

Counter.count = 5;

System.out.println(Counter.count); // Output: 5

Counter.increment();

System.out.println(Counter.count); // Output: 6

}

}

**2. What is class loading and how does the Java program actually executes?**

**Ans:** Class loading is the process of loading the Java class files into memory, so that the Java virtual machine (JVM) can execute the code contained in the class files.

When a Java program is executed, the JVM loads the class files in a particular order, and performs several tasks, including:

1. Verifying the bytecode to ensure that it is valid and follows the Java language specifications.
2. Resolving symbolic references to other classes and methods.
3. Allocating memory for class variables and initializing them to default values.
4. Executing the code contained in the class's static initializer blocks.

**3. Can we make a local variable as static?**

**Ans:** No, you cannot make a local variable as static.

**4. Why is the static block executed before the main method in Java?**

**Ans:** The static block in Java is executed before the main method because the static block is executed when the class is first loaded into memory by the Java Virtual Machine (JVM). The main method, on the other hand, is executed when the program is run from the command line.

The static block is used to initialize class variables and perform any other setup tasks that need to be done before any instances of the class are created

**5. Why is a static method also called a class method?**

**Ans:** A static method in Java is also called a class method because it is a method that belongs to the class, rather than to an instance of the class.

**6. What is the use of static blocks in java?**

**Ans:** The static blocks in Java are used to initialize class variables and perform any other setup tasks that need to be done before any instances of the class are created.

**7. Difference between Static and instance variable.**

**Ans**:

1. Scope: A static variable is associated with the class as a whole, while an instance variable is associated with a specific instance of the class. This means that a static variable can be accessed directly from the class, while an instance variable can only be accessed through an instance of the class.
2. Memory allocation: Static variables are stored in the class-level memory, while instance variables are stored in the heap memory associated with each instance of the class. This means that there is only one copy of a static variable shared by all instances of the class, while each instance of the class has its own copy of instance variables.
3. Initialization: Static variables are initialized when the class is first loaded into memory, while instance variables are initialized when an instance of the class is created.
4. Visibility: Static variables are accessible from any instance of the class, while instance variables are only accessible from the instance of the class in which they are defined.

**8. Difference between static and non static members.**

**Ans:**

1. Scope: A static member is associated with the class as a whole, while a non-static member is associated with a specific instance of the class. This means that a static member can be accessed directly from the class, while a non-static member can only be accessed through an instance of the class.
2. Memory allocation: Static members are stored in the class-level memory, while non-static members are stored in the heap memory associated with each instance of the class. This means that there is only one copy of a static member shared by all instances of the class, while each instance of the class has its own copy of non-static members.
3. Initialization: Static members are initialized when the class is first loaded into memory, while non-static members are initialized when an instance of the class is created.
4. Visibility: Static members are accessible from any instance of the class, while non-static members are only accessible from the instance of the class in which they are defined.