1. Why do we need static keyword in java Explain with an example?
   - We need static keyword in java to sharing data across instances.
   - For an example:

```
public class Counter {
    private static int count = 0;

    public Counter() {
        count++;
    }

    public static int getCount() {
        return count;
    }
}

public class Main {
    public static void main(String[] args) {
        Counter c1 = new Counter();
        Counter c2 = new Counter();
        Counter c3 = new Counter();

        System.out.println(Counter.getCount()); // Output: 3
    }
}
```

2. What is class loading and how does the java program actually execute?

   Class loading is the process by which the Java Virtual Machine (JVM) loads the byte code of a Java class into memory.
   When a Java program is executed, it goes through several stages, including class loading. Here's a step-by-step explanation of how a Java program is executed:
   - Compilation
   - Class Loading
   - Linking
   - Initialization
   - Execution

3. Can we mark a local variable as static?

   No, it is not possible to mark a local variable as static in Java. The **static** keyword is used to define members (variables or methods) that belong to a class itself

rather than to an instance of the class. Static members are associated with the class and are shared among all instances of that class.

Therefore, the **static** keyword is not applicable to local variables in Java. It is used for members of a class, such as static variables and static methods, to provide class-level behavior and shared data.

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

In Java, the static block is executed before the **main** method because it is part of the class initialization process. When a class is loaded and initialized by the Java Virtual Machine (JVM), it goes through a series of steps, and one of those steps is the execution of static blocks.

Here's the order of execution when a class is loaded and initialized:

- Class Loading
- Linking
- Initialization
- Execution

So, the static block is executed as part of the class initialization process before the **main** method because it allows you to set up any necessary initialization logic or perform additional actions before the program starts executing. This ensures that the class's static state is properly initialized before the program begins its execution.

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

A static method is also called a class method because it belongs to the class itself rather than to an instance of the class. It is associated with the class and can be accessed using the class name, without the need to create an object or instance of the class.

```
public class MathUtils {
    public static int square(int num) {
        return num * num;
    }
}

public class Main {
    public static void main(String[] args) {
        int result = MathUtils.square(5); // Accessing the static method using the class name
        System.out.println(result); // Output: 25
    }
}
```

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

   In Java, static blocks are used to initialize static variables or to perform other one-time setup operations that should be executed before the class is used. They are executed only once, during the initialization of the class, before any other code is executed.

   Some use cases for static blocks:

   - **Initializing static variables**

     ```
     public class MyClass {
        public static int someStaticVariable;

        static {
           // Complex initialization logic
           someStaticVariable = computeInitialValue();
        }
     }
     ```

   - **Resource initialization**

     ```
     public class ResourceHandler {
        private static Resource resource;

        static {
           resource = new Resource();
           resource.initialize();
        }
     }
     ```

7. Difference between Static and instance variables.

   Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed. Static variables are created when the program starts and destroyed when the program stops. Instance variables can be accessed directly by calling the variable name inside the class.

8. Difference between Static and non static members.

   Static variables are shared among all instances of a class. Non static variables are specific to that instance of a class. Static variable is like a global variable and is available to all methods. Non static variable is like a local variable and they can be accessed through only instance of a class.