Static

1. **Why do we need static keyword in Java Explain with example?**
   Answer:
   It makes our program more efficient, as every object doesn't allocate separate memory to a static variable.

   Whenever a common copy of data has to be shared among all the object of the class then we will go with static variable because during the class loading itself memory will be allocated in the heap area and same copy of data will be used among all the objects and memory is allocated only once hence it is more efficient.

   static local variables are not allowed in Java

   The static keyword in Java is mainly used for memory management. The static keyword in Java is used to share the same variable or method of a given class.

   Example:

```
package Static;

class Student {

    String name;
    int roll;

    static String clgName;

    static int counter = 0;

    Student (String name) {
        this.name = name;
        this.roll = setRoll();
    }

    static void setClgName(String name){
```

```
        clgName = name;
    }

    static int setRoll(){
        counter++;
        return counter;
    }

    void getStudentInfo(){
        System.out.println("Roll no.: "+this.roll);
        System.out.println("Name : "+this.name);
        System.out.println("College : "+ clgName);
        System.out.println("**********************************************
*");
    }


}

public class NeedStatic {
    public static void main(String[] args) {
        Student.setClgName("PW Skills");

        Student st1 = new Student("Rajan");
        Student st2 = new Student("Ajay");

        st1.getStudentInfo();
        st2.getStudentInfo();


    }

}
```

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

Answer: Loading is the process of finding the binary representation of a class or interface type with a particular name and *creating* a class or interface from that binary representation.

The Java ClassLoader is a part of the Java Runtime Environment that dynamically loads Java classes into the Java Virtual Machine. The Java run time system does not need to know about files and file systems because of classloaders. Java classes aren't loaded into memory all at once, but when required by an application. At this point, the Java ClassLoader is called by the JRE and these ClassLoaders load classes into memory dynamically.

how does the Java program actually execute; -

Java involves a two-step execution, first through an OS-independent compiler; and second, in a virtual machine (JVM) which is custom-built for every operating system.

Principle 1: Compilation

First, the source '.java' file is passed through the compiler, which then encodes the source code into a machine-independent encoding, known as Bytecode. The content of each class contained in the source file is stored in a separate '.class' file. While converting the source code into the bytecode, the compiler follows the following steps:

Principle 2: Execution

The class files generated by the compiler are independent of the machine or the OS, which allows them to be run on any system. To run, the main class file (the class that contains the method main) is passed to the JVM and then goes through three main stages before the final machine code is executed. These stages are:

These states do include:

1. ClassLoader
2. Bytecode Verifier
3. Just-In-Time Compiler

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

Answer: Java does not allow static local variables. The compiler will throw the compilation error.

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

Answer: The static blocks always execute first before the main() method in Java because the compiler stores them in memory at the time of class loading and before the object creation.

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

   Answer: A static method is a method that belongs to a class rather than an instance of a class. This means you can call a static method without creating an object of the class. For that static methods are sometimes called class methods

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

   Answer: Static block in java is used for changing the default value of static variables, initializing static variables of the class, write a set of codes that you want to execute during the class loading in memory

7. **Difference between Static and Instance variables.**

   Answer:

   Static Variable:

   - Class variables also known as static variables are declared with the static keyword in a class, but outside a method, constructor or a block.
   - Static variables are created when the program starts and destroyed when the program stops.
   - Static variables can be accessed using class name. - ClassName.VariableName.
   - Static variables can be accessed by static and non-static methods.
   - Static variables reduce the amount of memory used by a program.

   Instance Variable:

   - Instance variables are declared in a class, but outside a method, constructor or any block.
   - Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.
   - Instance Variable can be accessed only by creating objects.
   - Non static (Instance) variables cannot be accessed inside a static method.
   - Non static variables do not reduce the amount of memory used by a program.

8. Difference between static and non-static members.

Answer:

- The static methods can be accessed directly from the class, while non-static methods (or instance methods as I like to call them) have to be accessed from an instance. That is why instantiating needs to be done for instance methods, while for static methods it's just not needed.
- static members are accessed by their class name which encapsulates them, but non-static members are accessed by object reference.
- Static variable will get memory in the method area. But instance variable will get memory in the heap area.
- If the value does not change from object to object then we need to "static variables" . but If the value changes from object to object then we need to use non-static variable .
- Inside a static area we can access static variables only but Inside a non-static area we can access both static and non-static variables.