Static Keyboard

Assignment Questions

Assignment Questions

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

   Ans :

- **In Java, the static keyword is used to create variables or methods that belong to a class rather than to instances (objects) of the class.**
- **This means that the static members are shared among all instances of the class and can be accessed using the class name itself, without needing to create an object. The static keyword is often used for utility methods, constants, or variables that should be common to all instances of the class.**

```
public class Demo {
 static int a;
 static{
     a=15;
     System.out.println ("control in the static block");
     System.out.println (a);
 }
 static void disp (){
     System.out.println ("control in the static method");
 }

 public static void main(String[]args){
     System.out.println ("control in the main method");
     disp ();
 }
}
```

- **Output :**

```
control in the static block
15
control in the main method
control in the static method
```

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

Ans :

- **Class loading is a fundamental concept in Java and other programming languages that use a similar execution model. It refers to the process of loading classes (which are the building blocks of Java programs) into memory so that they can be executed by the Java Virtual Machine (JVM).**

- **When a Java program is executed, it goes through several steps, including class loading, linking, and initialization, which together make up the process of loading and executing Java code**

- **Compilation: Java source code is first compiled into bytecode using the Java compiler (javac). Bytecode is a platform-independent representation of the source code that can be executed by the JVM.**

- **Class Loading: When a Java program starts, the JVM's class loader subsystem is responsible for loading classes into memory as they are needed. Classes are loaded from various sources, such as the local file system, network, or even dynamically generated.**

- **Linking: After a class is loaded, the JVM performs linking, which consists of three sub-steps:**

  a. **Verification: Ensures that the loaded bytecode is well-formed and follows the Java language and JVM rules.**

  b. **Preparation: Allocates memory for class variables and sets their default values.**

  c. **Resolution: Replaces symbolic references (such as method and field names) with direct references.**

- **Initialization: This step involves executing the static initialization blocks and assigning values to static variables. Static initialization occurs only once per class, and it's guaranteed to be thread-safe by the JVM.**

- **Execution: Once the classes are loaded, linked, and initialized, the JVM starts executing the program by invoking the main method of the designated class. From there, the program's execution follows the flow of its code, invoking methods, creating objects, and interacting with data.**

- **Throughout the execution process, the JVM manages memory, handles exceptions, performs garbage collection to free up unused memory, and provides various runtime services to the executing program.**

3. Can we mark a local variable as static

Ans :

- No, we cannot mark a local variable as static in Java. The static keyword is used to define a class-level (static) variable or method that belongs to the class itself rather than to instances of the class. It is not applicable to local variables.

- Local variables are declared within methods, constructors, or blocks and have a limited scope within that block of code. They are created and destroyed as the flow of execution enters and exits the block. Therefore, it wouldn't make sense to declare a local variable as static since it contradicts the concept of a local variable having a limited scope and being specific to a particular instance of the method or block.

- If you want a variable to be shared among all instances of a class, you would declare a static class-level variable. If you want a variable to be specific to an instance of a class, you would declare an instance variable (also known as a non-static member variable).

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

Ans :

- In Java, the static block is executed before the main method because of the order of class loading and initialization.

- When a Java program is executed, the Java Virtual Machine (JVM) loads and initializes classes as needed. The order of execution can be summarized as follows:

a. Class Loading: The JVM loads the classes required by the program into memory. This includes loading the class that contains the main method.

b. Static Block Execution: If a class contains static blocks, they are executed as part of the class initialization process. static blocks are used to initialize static variables or perform other class-level initialization tasks.

c. main Method Execution: After the class is loaded and its static blocks are executed, the main method is called and executed.

```
public class StaticBlock {
    static {
        System.out.println("Static block is executed.");
    }

    public static void main(String[] args) {
        System.out.println("Main method is executed.");
    }
}
```

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

Ans :

- **A static method is also referred to as a "class method" because it is associated with the class itself rather than with instances of the class. This means that you can call a static method on the class itself, without needing to create an instance of the class.**

- **Belongs to the Class: A static method is defined at the class level, and it is not tied to any particular instance of the class. It operates on class-level data and does not have access to instance-specific data.**

- **No Instance Required: Since a static method is not associated with instances, you can call it using the class name, without creating an object of the class.**

- **Shared among All Instances: static methods are shared among all instances of the class. Any changes made by a static method are visible to all instances.**

- **Cannot Access Non-Static Members: Inside a static method, you cannot directly access non-static (instance) variables or methods of the class. You can only access other static members.**

- **Use Case: static methods are often used for utility methods that perform tasks related to the class but do not require access to instance-specific data.**

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

Ans :

- **Static blocks in Java are used for class-level initialization. They provide a way to perform actions that need to be executed once when the class is loaded into memory, regardless of whether any objects of the class are created. Static blocks are defined using the static keyword and enclosed within curly braces {}. They are executed in the order they appear in the code.**

a. **Initializing Static Variables: Static blocks are often used to initialize static variables. This is especially useful when the initialization requires more than a simple assignment and might involve complex calculations or loading resources.**

b. **Loading Resources: If a class needs to load resources from external sources, such as reading configuration files or initializing database connections, a static block can be used to perform these actions during class loading.**

c. **Singleton Pattern Initialization: In the Singleton design pattern, where a class ensures that only one instance is created, a static block can be used to create the instance and perform any necessary initialization.**

d. **Caching: Static blocks can be used to initialize and populate data caches that are shared among all instances of the class.**

e. Setting Up Constants: Static blocks can be used to initialize constant values or lookup tables that are required for the class.

7. Difference between Static and Instance variables

Ans :

- **Sure, here is a more detailed answer about the difference between static and instance variables:**
- **Static variables are declared with the static keyword. They are created when the class is loaded into memory and destroyed when the program terminates. Static variables are shared by all objects of a class, which means that they have the same value for all objects. For example, a static variable could be used to store the number of objects that have been created of a particular class.**
- **Instance variables are declared without the static keyword. They are created when each object is created and destroyed when the object is destroyed. Instance variables are unique to each object, which means that they can have different values for different objects. For example, an instance variable could be used to store the name of a particular customer.**
- **Here are some other key differences between static and instance variables:**
- **Scope: The scope of a static variable is the entire class. This means that it can be accessed from any method or block in the class. The scope of an instance variable is only the object in which it is declared. This means that it can only be accessed from methods or blocks in that object.**
- **Initialization: Static variables can be initialized in a static block or a static method. Instance variables can be initialized in a constructor or a method.**
- **Access: Static variables can be accessed using the class name or an object reference. Instance variables can only be accessed using an object reference.**
- **Here are some examples of when to use static and instance variables:**
- **Static variables should be used to store information that is common to all objects of a class, such as the class name, version number, or a counter. For example, you could use a static variable to keep track of the number of objects that have been created of a particular class.**
- **Instance variables should be used to store information that is specific to each object, such as the object's name, age, or weight. For example, you would use an instance variable to store the name of a particular customer.**

8. Difference between static and non static members

Ans :

Sure, here are the key differences between static and non-static members:

- **Scope: The scope of a static member is the entire class. This means that it can be accessed from any method or block in the class. The scope of a non-static member is only the object in which it is declared. This means that it can only be accessed from methods or blocks in that object.**
- **Initialization: Static members can be initialized in a static block or a static method. Non-static members can be initialized in a constructor or a method.**
- **Access: Static members can be accessed using the class name or an object reference. Non-static members can only be accessed using an object reference.**
- **Memory allocation: Static members are allocated once when the class is loaded into memory. Non-static members are allocated each time an object of the class is created.**
- **Number of copies: There is only one copy of each static member, regardless of how many objects of the class are created. There is one copy of each non-static member for each object of the class.**
- **Inheritance: Static members are not inherited by subclasses. Non-static members are inherited by subclasses.**

Here are some examples of when to use static and non-static members:

- **Static members should be used to store information that is common to all objects of a class, such as the class name, version number, or a counter. For example, you could use a static variable to keep track of the number of objects that have been created of a particular class.**
- **Non-static members should be used to store information that is specific to each object, such as the object's name, age, or weight. For example, you would use an instance variable to store the name of a particular customer.**

Here are some other examples:

- **A static method can be used to print the class name or version number.**
- **A static method can be used to calculate a mathematical formula that is independent of any particular object.**
- **A non-static method can be used to set or get the value of an instance variable.**
- **A non-static method can be used to perform an operation that is specific to a particular object.**