Different type of variables and their scope (local, instance, class)

In Java, variables can have different scopes depending on where they are declared. The three main types of variables and their scopes are:

Local Variables:

- Local variables are declared within a method, constructor, or a block of code, and they are only accessible within that particular block.
- Their scope is limited to the block in which they are declared.
- Local variables must be initialized before they can be used.
- They are created when the block is entered and destroyed when the block is exited.
- Local variables do not have default values and must be initialized before they can be used.

Example:

```
public void exampleMethod() {
    int localVar = 10; // local variable
    System.out.println(localVar);
}
```

Instance Variables:

- Instance variables, also known as member variables or non-static variables, are declared within a class but outside of any method, constructor, or block.
- Each instance of the class (object) has its own copy of instance variables.
- Instance variables are accessible throughout the class and can be accessed by any method, constructor, or block within the class.
- Instance variables are initialized when an object is created using the `new` keyword and exist as long as the object exists.

- They are initialized to default values (e.g., 0 for numeric types, false for boolean, null for objects) if not explicitly initialized.

Example:

```
public class DemoClass {
    int instanceVar; // instance variable

    public void DemoMethod() {
        System.out.println(instanceVar);
    }
}
```

Class Variables (Static Variables):

- Class variables, also known as static variables, are declared with the `static` keyword within a class but outside of any method, constructor, or block.
- Class variables are shared among all instances of a class. They are associated with the class itself, rather than with individual objects.
- Unlike instance variables, there is only one copy of a class variable shared by all instances of the class.
- Class variables are initialized when the class is loaded and exist as long as the program is running.
- Class variables can be accessed using the class name, without creating an instance of the class.
- Class variables are accessible to all methods, constructors, and blocks within the class and can also be accessed using the class name outside the class.
- They are initialized to default values (similar to instance variables) if not explicitly initialized.

Example:

```
public class DemoClass {
    static int classVar; // class variable

    public void DemoMethod() {
        System.out.println(classVar);
    }
}
```

It's worth mentioning that local variables take precedence over instance and class variables with the same name within their scope.

localVar >instanceVar>classVar

```
public class Demo{
    // Class variable
    static int classVariable = 10;

    // Instance variable
    int instanceVariable = 20;

    public void DemoMethod() {
        // Local variable
        int localVariable = 30;

        System.out.println("Class Variable: " + classVariable);
        System.out.println("Instance Variable: " + instanceVariable);
        System.out.println("Local Variable: " + localVariable);
    }
}
```

Differences Between the Instance Variables and the Static/class Variables

Now let us discuss the differences between the Instance variables and the Static variables:

- Each object will have its own copy of an instance variable, whereas we can only have one copy of a static variable per class, irrespective of how many objects we create. Thus, **static variables** are good for **memory management**.
- Changes made in an instance variable using one object will not be reflected in other objects as each object has its own copy of the instance variable. In the case of a static variable, changes will be reflected in other objects as static variables are common to all objects of a class.
- We can access instance variables through object references, and static variables can be accessed directly using the class name**.**
- 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.

```
What is static keyword in Java?
 Can I Declare local Variable as Static.

Can a static block exist without a main() method?
Why main() method is declared as static?
Can we Overload static methods in Java
Can we Override static methods in Java
Can we have multiple static blocks in our code?
Can constructors be static in Java?
```