1. What is a String in Java?

In Java, a String is a sequence of characters. It's an object that represents a series of characters. Strings are widely used in Java for storing and manipulating text. They are immutable, meaning once created, their values cannot be changed. You can create a String in Java using double quotes, like this:

String myString = "Hello, World!";

Or by creating a new String object:

String myString = new String("Hello, World!");

1. Types of String in Java are?

In java String are classified into 2 types

1. Mutable String

2. Immutable String

Mutable String: - Once if we create a String, on that String if we try to perform any operation and if those changes get reflected in the same object, then such String are called Mutable String. Example- StringBuffer, StringBuilder.

Immutable String: - Once if we create a String, on that String if we try to perform any operation then those changes won’t be reflected in the same object, rather a new object will be created. Such type of String is called as immutable String. Example- String.

1. In how many ways can you create string objects in Java?

1. Using String literals:

String str1 = "Hello";

2. Using the new keyword and the String class constructor:

String str2 = new String("World");

3. Using the StringBuilder or StringBuffer classes to build strings dynamically:

StringBuilder sb = new StringBuilder();

sb.append("Dynamic").append(" ").append("String");

String dynamicString = sb.toString();

1. What is a string constant pool?

In Java, the string constant pool is a special area of memory where string literals are stored. When you create a string using a literal (e.g., `"Hello"`), Java checks if an equivalent string already exists in the constant pool. If it does, it returns a reference to that string; otherwise, it creates a new string in the pool. This mechanism helps conserve memory by avoiding duplicate string objects with the same value. Additionally, string literals in Java are interned by default, meaning that identical string literals within the same classloader share the same memory location in the string constant pool.

1. What do you mean by mutable and immutable objects?

In Java, mutable objects are objects whose state can be changed after they are created. This means you can modify their properties or contents without creating a new object.

Immutable objects, on the other hand, are objects whose state cannot be changed after they are created. Any attempt to modify an immutable object results in the creation of a new object with the modified state.

Examples of mutable objects in Java include StringBuilder, StringBuffer, ArrayList, and HashMap.

Examples of immutable objects in Java include String, Integer, Double, and other wrapper classes, as well as instances of classes with final fields and no mutator methods.

Immutable objects are often preferred for their thread safety, simplicity, and ease of reasoning about their state. They are particularly useful in concurrent programming and functional programming paradigms.

1. Where exactly is the string constant pool located in the memory?

In Java, the string constant pool is located in the method area of the JVM's memory. The method area is a part of the JVM's runtime data area and is shared among all threads. It stores class structures, method code, runtime constant pool, and other static data. The string constant pool is a special section within the method area where string literals are stored. This allows for efficient memory management and ensures that duplicate string literals are not unnecessarily duplicated, saving memory resources.