String

🔷 String:

-> A String is a sequence of characters used to represent text. It's one of the most commonly used data types in the language.

-> A String is an object, not a primitive type.

-> Strings are immutable, meaning once a String object is created, its value cannot be changed.

-> Strings are defined in the java.lang package (imported by default).

🔷 Declaring and Using Strings:

```
String greeting = "Hello, World!";

System.out.println(greeting);
```

🔷 String Methods:

1. Length & Character Access:

Method	Description	Example
length()	Returns the number of characters in the string	"hello".length() → 5
charAt(int index)	Returns the character at a specific index	"hello".charAt(1) → 'e'
codePointAt(int index)	Returns the Unicode value at the index	"A".codePointAt(0) → 65

2. Comparison & Checking:

Method	Description	Example
equals(String s)	Checks if two strings are equal (case-sensitive)	"Java".equals("java") → false
equalsIgnoreCase(String s)	Checks if two strings are equal (case-insensitive)	"Java".equalsIgnoreCase("java") → true
compareTo(String s)	Lexicographically compares two strings	"abc".compareTo("abd") → -1
contains(CharSequence s)	Checks if the string contains a substring	"hello".contains("ell") → true
startsWith(String prefix)	Checks if the string starts with prefix	"hello".startsWith("he") → true
endsWith(String suffix)	Checks if the string ends with suffix	"hello".endsWith("lo") → true
isEmpty()	Checks if the string is empty	"".isEmpty() → true
isBlank() (Java 11+)	Checks if the string is empty or whitespace only	" ".isBlank() → true

3.Substring & Searching

Method	Description	Example
indexOf(String s)	Finds first occurrence of a substring	"hello".indexOf("l") → 2
lastIndexOf(String s)	Finds last occurrence of a substring	"hello".lastIndexOf("l") → 3
substring(int start)	Returns substring from start index	"hello".substring(2) → "llo"
substring(int start, int end)	Returns substring between two indices	"hello".substring(1, 4) → "ell"

4. Modification & Formatting

Method	Description	Example
toUpperCase()	Converts all characters to uppercase	"java".toUpperCase() → "JAVA"
toLowerCase()	Converts all characters to lowercase	"JAVA".toLowerCase() → "java"
trim()	Removes leading/trailing whitespace	" hello ".trim() → "hello"
replace(char old, char new)	Replaces characters	"cat".replace('c', 'b') → "bat"

Method	Description	Example
replaceAll(String regex, String replacement)	Replaces using regex	"abc123".replaceAll("\\d", "") → "abc"
repeat(int count) (Java 11+)	Repeats the string	"ha".repeat(3) → "hahaha"

5. Splitting and Joining

Method	Description	Example
split(String regex)	Splits the string into an array using regex	"a,b,c".split(",") → ["a", "b", "c"]
join(CharSequence delimiter, CharSequence... elements)	Joins elements into a string	String.join("-", "a", "b", "c") → "a-b-c"

🔷**StringBuffer :**

-> StringBuffer is a **mutable sequence of characters**. Unlike String, it allows you to modify the contents (e.g., append, insert, delete) **without creating a new object**.

Features:

->Mutable (unlike String).

-> **Thread-safe** (synchronized methods).

-> Slower than StringBuilder in single-threaded applications.

🔷 StringBuilder :

-> StringBuilder is **almost identical** to StringBuffer, but it is **not thread-safe**.

Features:

-> Mutable

-> Not synchronized → faster than StringBuffer for single-threaded code.

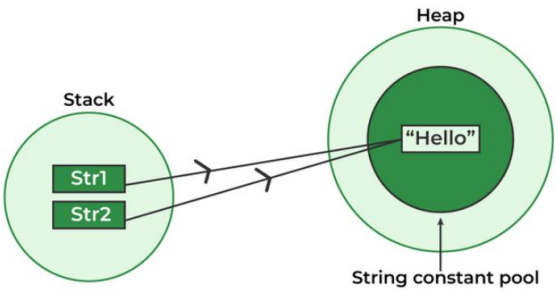
Comparison: String, StringBuffer, and StringBuilder

Feature	**String**	**StringBuffer**	**StringBuilder**
Mutability	Immutable	Mutable	Mutable
Thread-safe	Yes (immutable)	Yes (synchronized)	No
Performance	Fast (read only)	Slower (thread-safe)	Faster (no sync)
Use case	Constant strings	Multi-threaded edits	Single-threaded edits

🔷 String Pool in Java

-> The **String Pool** (also called the **interned string pool**) is a special area in the **Java heap memory** where **unique string literals** are stored.

-> It helps **optimize memory usage** and **improve performance** by reusing immutable String objects.



✅ How It Works

-> Example: When you create a string like this:

```
String s1 = "Hello";

String s2 = "Hello";
```

-> Java stores the string "Java" in the string pool only **once**. Both s1 and s2 refer to the **same object** in memory.

🔍 Why Use the String Pool?

-> Saves memory: avoids storing duplicate string literals.

-> Improves speed: string comparisons using == are faster when referencing the same object.

Important Coding Questions On String

1. Write a Java program to get the character at the given index within the string.

Day6Q1String.java

```java
import java.util.Scanner;
public class Day6Q1String {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the string
        System.out.print(s:"Enter a string: ");
        String input = scanner.nextLine();

        // Input the index
        System.out.print(s:"Enter the index: ");
        int index = scanner.nextInt();

        // Check if index is valid
        if (index >= 1 && index < input.length()) {
            char result = input.charAt(index);
            System.out.println("Character at index " + index + ": " + result);
        } else {
            System.out.println("Invalid index. Must be between 0 and " + (input.length() - 1));
        }

        scanner.close();
    }
}
```

Output:

Enter a string: Ankita

Enter the index: 3

Character at index 1: k

2. Write a Java program to concatenate a given string to the end of another string.

Day6Q2String.java > ...

```java
import java.util.Scanner;
public class Day6Q2String
Run | Debug
{    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the first string
        System.out.print(s:"Enter the first string: ");
        String str1 = scanner.nextLine();

        // Input the second string
        System.out.print(s:"Enter the second string: ");
        String str2 = scanner.nextLine();

        // Concatenate the strings
        String result = str1.concat(str2);  // or str1 + str2

        // Output the result
        System.out.println("Concatenated string: " + result);

        scanner.close();
    }
}
```

Output:

Enter the first string: Ankita

Enter the second string: Mitra

Concatenated string: Ankita Mitra

3. Write a Java program to compare a given string to a specified string buffer.

Day6Q3String.java > ...

```java
import java.util.Scanner;
public class Day6Q3String
{
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input a string
        System.out.print(s:"Enter a String: ");
        String str = scanner.nextLine();

        // Input a string for the buffer
        System.out.print(s:"Enter a StringBuffer: ");
        String bufferInput = scanner.nextLine();
        StringBuffer stringBuffer = new StringBuffer(bufferInput);

        // Convert StringBuffer to String and compare
        boolean isEqual = str.equals(stringBuffer.toString());

        // Output result
        if (isEqual) {
            System.out.println(x:"The String and StringBuffer are equal.");
        } else {
            System.out.println(x:"The String and StringBuffer are NOT equal.");
        }

        scanner.close();
    }
}
```

Output:

Enter a String: Hello

Enter a StringBuffer: Hello

The String and StringBuffer are equal.

4. Write a Java program to create a String object with a character array.

Day6Q4String.java > ...

```java
public class Day6Q4String
{
    Run | Debug
    public static void main(String[] args) {
        // Define a character array
        char[] charArray = { 'H', 'e', 'l', 'l', 'o', ' ', 'J', 'a', 'v', 'a' };

        // Create a String object from the character array
        String result = new String(charArray);

        // Print the result
        System.out.println("The string is: " + result);
    }
}
```

Output:

The string is: Hello Java

5. Write a Java program to get the length of a given string.

Day6Q5String.java > ...

```java
import java.util.Scanner;
public class Day6Q5String
{
    Run | Debug
    public static void main(String[] args) {
     Scanner scanner = new Scanner(System.in);

     // Input the string
     System.out.print(s:"Enter a string: ");
     String input = scanner.nextLine();

     // Get the length of the string
     int length = input.length();

     // Display the result
     System.out.println("Length of the string: " + length);

     scanner.close();
    }
}
```

Output:

Enter a string: Hello Java

Length of the string: 10

6. Write a Java program to replace a specified character with another character.

Day6Q6String.java

```java
import java.util.Scanner;

public class Day6Q6String {
    Run | Debug
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);

        // Input the string
        System.out.print(s:"Enter the original string: ");
        String original = scanner.nextLine();

        // Input the character to be replaced
        System.out.print(s:"Enter the character to replace: ");
        char oldChar = scanner.next().charAt(index:0);

        // Input the replacement character
        System.out.print(s:"Enter the new character: ");
        char newChar = scanner.next().charAt(index:0);

        // Replace the character
        String result = original.replace(oldChar, newChar);

        // Display the result
        System.out.println("Modified string: " + result);

        scanner.close();
    }
}
```

Output:

Enter the original string: hello world

Enter the character to replace: o

Enter the new character: a

Modified string: hella warld

7. Write a Java program to convert all characters in a string to lowercase.

```java
import java.util.Scanner;
public class Day6Q7String
{
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the string
        System.out.print(s:"Enter a string: ");
        String input = scanner.nextLine();

        // Convert to lowercase
        String lowercase = input.toLowerCase();

        // Display the result
        System.out.println("Lowercase string: " + lowercase);

        scanner.close();
    }
}
```

Output:

Enter a string: ANKITA

Lowercase string: ankita

8. Write a java program to check whether a given string is palindrome or not.

Day6Q7String.java Day6Q8String.java

Day6Q8String.java > Day6Q8String

```java
import java.util.Scanner;
public class Day6Q8String
{
    Run | Debug
    public static void main(String[] args) {
     Scanner sc = new Scanner(System.in);
     System.out.print(s:"Enter a string: ");
     String str = sc.nextLine().replaceAll(regex:"\\s+", replacement:"").toLowerCase();
     String rev = new StringBuilder(str).reverse().toString();

     if (str.equals(rev))
         System.out.println(x:"Palindrome");
     else
         System.out.println(x:"Not a palindrome");

     sc.close();
    }
}
```

Output:

Enter a string: Racecar

Palindrome