Strings in Java

1. What is Mutable String in Java Explain with an example

Ans1. **Mutable String in Java**

In Java, **mutable strings** are strings that can be modified **without creating a new object**. The primary classes used for mutable strings are:

1. `StringBuilder` (Faster, not thread-safe)
2. `StringBuffer` (Thread-safe, but slower)

Difference Between `StringBuilder` and `StringBuffer`

Feature	`StringBuilder`	`StringBuffer`
Performance	Faster	Slower (due to synchronization)
Thread Safety	Not thread-safe	Thread-safe (synchronized methods)
Use Case	Single-threaded apps	Multi-threaded apps

2. WAP to reverse a String Input:
“PWSKILLS” Output: “SLLIKSPW”

Ans2.
```
public class ReverseString {
    public static void main(String[] args) {
        // Input string
        String input = "PWSKILLS";

        // Reversed string
        String reversed = reverseString(input);

        // Output the result
        System.out.println("Input: " + input);
        System.out.println("Output: " + reversed);
    }

    // Method to reverse a string
    public static String reverseString(String str) {
        // Convert the string to a character array
        char[] charArray = str.toCharArray();

        // Initialize pointers for the start and end of the array
        int start = 0;
        int end = charArray.length - 1;

        // Swap characters from start and end until the pointers meet
        while (start < end) {
            char temp = charArray[start];
            charArray[start] = charArray[end];
            charArray[end] = temp;

            // Move the pointers towards the center
            start++;
            end--;
        }

        // Convert the character array back to a string
        return new String(charArray);
    }
}
```

3. WAP to reverse a sentence while preserving the position
Input: Think Twice Output: “kniht eciwt”

```
Ans3. public class ReverseSentencePreservePosition {
    public static void main(String[] args) {
        // Input sentence
        String input = "Think Twice";

        // Reverse each word while preserving the position
        String reversedSentence = reverseSentence(input);

        // Output the result
        System.out.println("Input: " + input);
        System.out.println("Output: " + reversedSentence);
    }
```

```
// Method to reverse each word in a sentence
public static String reverseSentence(String sentence) {
    // Split the sentence into words
    String[] words = sentence.split(" ");

    // StringBuilder to store the result
    StringBuilder result = new StringBuilder();

    // Iterate through each word
    for (String word : words) {
        // Reverse the current word
        String reversedWord = reverseWord(word);

        // Append the reversed word to the result
        result.append(reversedWord).append(" ");
    }

    // Remove the trailing space and return the result
    return result.toString().trim();
}

// Method to reverse a single word
public static String reverseWord(String word) {
    // Convert the word to a character array
    char[] charArray = word.toCharArray();

    // Initialize pointers for the start and end of the array
    int start = 0;
    int end = charArray.length - 1;

    // Swap characters from start and end until the pointers meet
    while (start < end) {
        char temp = charArray[start];
        charArray[start] = charArray[end];
        charArray[end] = temp;

        // Move the pointers towards the center
        start++;
        end--;
    }
```

```
        // Convert the character array back to a string
        return new String(charArray);
    }
}
```

4. WAP to sort a String Alphabetically .

Ans4.
```
import java.util.Arrays;

public class SortStringAlphabetically {
    public static void main(String[] args) {
        // Input string
        String input = "pwskills";

        // Sort the string alphabetically
        String sortedString = sortString(input);

        // Output the result
        System.out.println("Input: " + input);
        System.out.println("Sorted: " + sortedString);
    }

    // Method to sort a string alphabetically
    public static String sortString(String str) {
        // Convert the string to a character array
        char[] charArray = str.toCharArray();

        // Sort the character array
        Arrays.sort(charArray);

        // Convert the sorted character array back to a string
        return new String(charArray);
    }
}
```