Assignment(Module-25)
Mutable String in Java

Q.1 In Java, a mutable string refers to a string-like object whose content can be changed after it is created. Java provides the `StringBuilder` and `StringBuffer` classes for mutable strings. Unlike the `String` class, which creates new objects whenever a string is modified, these classes allow you to modify the content of the string directly, improving performance when frequent modifications are required.

Characteristics of Mutable Strings in Java

- **Efficient:** Modifications such as appending, deleting, or inserting characters do not create new objects.
- **Thread-safety:**
    - `StringBuilder` is not thread-safe, meaning it is not synchronized.
    - `StringBuffer` is thread-safe, meaning it is synchronized for multi-threaded environments**.**

Example of Mutable String Using `StringBuilder`

```
public class MutableStringExample {

    public static void main(String[] args) {

        // Create a StringBuilder object

        StringBuilder sb = new StringBuilder("Hello");


        // Append a string to the original content

        sb.append(" World");

        System.out.println("After append: " + sb);


        // Insert a string at a specific index

        sb.insert(5, ",");

        System.out.println("After insert: " + sb);
```

```
        // Replace a part of the string
        sb.replace(6, 11, "Java");
        System.out.println("After replace: " + sb);

        // Delete a part of the string
        sb.delete(5, 6);
        System.out.println("After delete: " + sb);

        // Reverse the string
        sb.reverse();
        System.out.println("After reverse: " + sb);
    }
}
```

Q.2

```
package pw_java;

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

       // Using a loop to reverse the string
       String reversed = "";
       for (int i = original.length() - 1; i >= 0; i--) {
           reversed += original.charAt(i);
       }
       // Output the reversed string
       System.out.println("Original String: " + original);
```

```
        System.out.println("Reversed String: " + reversed);
    }
}
```

Q.3

```
package pw_java;

public class ReverseSentence {
    public static void main(String[] args) {
        // Input sentence
        String sentence = "Think Twice";
        // Split the sentence into words
        String[] words = sentence.split(" ");
        // Reverse each word and store the result
        StringBuilder reversedSentence = new StringBuilder();
        for (String word : words) {
            String reversedWord = new StringBuilder(word).reverse().toString();
            reversedSentence.append(reversedWord).append(" ");
        }
        // Trim the trailing space and print the result
        System.out.println("Original Sentence: " + sentence);
        System.out.println("Reversed Sentence: " +
reversedSentence.toString().trim());
    }
}
```

Q.4

```
package pw_java;
public class SortStringAlphabetically {
      public static void main(String[] args) {
        // Input string
        String original = "PWSKILLS";
        // Convert the string to a character array
        char[] charArray = original.toCharArray();
        // Implement Bubble Sort to sort the character array
        for (int i = 0; i < charArray.length - 1; i++) {
            for (int j = 0; j < charArray.length - 1 - i; j++) {
                if (charArray[j] > charArray[j + 1]) {
                    // Swap characters if they are out of order
                    char temp = charArray[j];
                    charArray[j] = charArray[j + 1];
                    charArray[j + 1] = temp;
                }
            }
        }
```

```
        // Convert the sorted character array back to a string
        String sortedString = new String(charArray);
        // Print the original and sorted strings
        System.out.println("Original String: " + original);
        System.out.println("Sorted String: " + sortedString);
    }
}
```