//Assignment file1

1.Write a Java program to reverse a string without using the inbuilt method.

Ans:
```
public class ReverseString {
    public static void main(String[] args) {
        String str = "Hello, World!";
        String reversedStr = reverseString(str);
        System.out.println("Original string: " + str);
        System.out.println("Reversed string: " + reversedStr);
    }

    public static String reverseString(String str) {
        char[] charArray = str.toCharArray();
        int left = 0;
        int right = charArray.length - 1;

        while (left < right) {
            // Swap characters at left and right indices
            char temp = charArray[left];
            charArray[left] = charArray[right];
            charArray[right] = temp;

            // Move to the next characters
            left++;
            right--;
        }

        return new String(charArray);
    }
}
```

2.Write a java program to know whether the given string is palindrome

Ans:
```
public class Palindrome {
    public static void main(String[] args) {
        String str = "madam";
        boolean isPalindrome = checkPalindrome(str);
        System.out.println("Input string: " + str);
        System.out.println("Is Palindrome? " + isPalindrome);
    }

    public static boolean checkPalindrome(String str) {
        int left = 0;
        int right = str.length() - 1;

        while (left < right) {
            if (str.charAt(left) != str.charAt(right)) {
```

```
            return false;
        }

        left++;
        right--;
    }

    return true;
  }
}
```

3.Write a java program to convert upper case to lower case and vice versa.

Ans:
```
public class CaseConverter {
  public static void main(String[] args) {
    String str = "Hello, World!";
    String convertedStr = convertCase(str);
    System.out.println("Original string: " + str);
    System.out.println("Converted string: " + convertedStr);
  }

  public static String convertCase(String str) {
    char[] charArray = str.toCharArray();

    for (int i = 0; i < charArray.length; i++) {
      if (Character.isUpperCase(charArray[i])) {
        charArray[i] = Character.toLowerCase(charArray[i]);
      } else if (Character.isLowerCase(charArray[i])) {
        charArray[i] = Character.toUpperCase(charArray[i]);
      }
    }

    return new String(charArray);
  }
}
```

4.Write a java program to remove a particular character from a string.

Ans:
```
public class CharacterRemover {
  public static void main(String[] args) {
    String str = "Hello, World!";
    char charToRemove = 'o';
    String removedStr = removeCharacter(str, charToRemove);
    System.out.println("Original string: " + str);
    System.out.println("Character to remove: " + charToRemove);
    System.out.println("String after removal: " + removedStr);
  }
```

```
    public static String removeCharacter(String str, char charToRemove) {
        StringBuilder sb = new StringBuilder();

        for (char c : str.toCharArray()) {
            if (c != charToRemove) {
                sb.append(c);
            }
        }

        return sb.toString();
    }
}
```

5.Write a java program to find the index of a substring.

Ans:
```
public class SubstringIndexFinder {
    public static void main(String[] args) {
        String str = "Hello, World!";
        String substring = "World";
        int index = findSubstringIndex(str, substring);

        System.out.println("Original string: " + str);
        System.out.println("Substring to find: " + substring);

        if (index != -1) {
            System.out.println("Substring found at index: " + index);
        } else {
            System.out.println("Substring not found in the string.");
        }
    }

    public static int findSubstringIndex(String str, String substring) {
        int index = str.indexOf(substring);
        return index;
    }
}
```

//Assignment file2

1.What is a string in java

Ans: In Java, a string is a sequence of characters. It is represented by the `String` class, which is part of the Java standard library. Strings are used to store and manipulate text-based data in Java programs.

2.Types of strings in java are?

Ans: In Java, there are two types of strings:
a. String Literal: A string created using double quotes, such as `"Hello, World!"`. String literals are stored in the string constant pool (explained in question 4) and are immutable (cannot be changed).
b. String Object: A string created using the `new` keyword and the `String` constructor, such as `new String("Hello, World!")`. String objects are stored in the heap memory and can be mutable (can be changed).

3.In how many ways can you create string objects in java?
Ans:There are several ways to create string objects in Java:
a. String Literal: Using double quotes to create a string literal, such as `"Hello, World!"`.
b. Using the `new` keyword and the `String` constructor, such as `new String("Hello, World!")`.
c. Using the `new` keyword and character array initialization, such as `new String(char[] {'H', 'e', 'l', 'l', 'o'})`.
d. Using the `new` keyword and byte array initialization for specific character encoding, such as `new String(byte[] {72, 101, 108, 108, 111}, "UTF-8")`

4.What is a string constant pool?
Ans: The string constant pool is a special memory area in Java where string literals are stored. It is a part of the Java Runtime Environment's method area. String literals, such as `"Hello, World!"`, are stored in the constant pool to optimize memory usage. The string constant pool ensures that multiple string literals with the same value are stored as a single copy, saving memory. This is possible because string literals are immutable and cannot be modified.

5.What do you mean by mutable and immutable objects?
Ans:In Java, mutable objects can be modified or changed after they are created, while immutable objects cannot be modified once they are created. In the context of strings:
a. Immutable Objects: String literals and the strings created using the `String` class are immutable. Once a string is created, its value cannot be changed. If you perform any operation on an immutable string, it creates a new string object with the modified value. This immutability provides advantages like thread-safety and caching.
b. Mutable Objects: If you need a mutable string, you can use the `StringBuilder` or `StringBuffer` classes in Java. These classes allow you to modify the contents of the string without creating new objects. Mutable strings are useful when you need to perform frequent modifications to the string, such as concatenation or appending.

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

Ans:The string constant pool is located in the Java Runtime Environment's method area, which is a part of the JVM's heap memory. The string constant pool is shared among all the string objects in a Java program. It is created when the Java class is loaded by the JVM, and it contains all the unique string literals encountered in the program. The constant pool is a memory optimization technique that allows strings with the same value to be stored as a single copy, reducing memory usage.

//Assignment file3

1.What is Mutable string in java explain with an example

Ans: n Java, a mutable string refers to a string that can be modified or changed after it is created. Unlike immutable strings (string literals and strings created using the `String` class), which cannot be modified once created, mutable strings allow you to modify their contents without creating new objects.

2.WAP to reverse a String

Input: “PWSKILLS”

Ans:
```
public class StringReverser {
    public static void main(String[] args) {
        String input = "PWSKILLS";
        String reversed = reverseString(input);
        System.out.println("Input: " + input);
        System.out.println("Output: " + reversed);
    }

    public static String reverseString(String input) {
        StringBuilder sb = new StringBuilder(input);
        sb.reverse();
        return sb.toString();
    }
}
```

3.WAP to reverse a sentence while preserving the position

input:Think Twice

Output:”kniht eciwt”

Ans:
```
public class SentenceReverser {
    public static void main(String[] args) {
        String input = "Think Twice";
        String reversed = reverseSentence(input);
        System.out.println("Input: " + input);
        System.out.println("Output: " + reversed);
    }

    public static String reverseSentence(String input) {
        String[] words = input.split(" ");
        StringBuilder sb = new StringBuilder();
```

```
        for (int i = words.length - 1; i >= 0; i--) {
            sb.append(reverseString(words[i])).append(" ");
        }

        return sb.toString().trim();
    }

    public static String reverseString(String input) {
        StringBuilder sb = new StringBuilder(input);
        sb.reverse();
        return sb.toString();
    }
}
```

4.WAP to sort a String Alphabetically
Ans:

```
import java.util.Arrays;

public class StringSorter {
    public static void main(String[] args) {
        String input = "openai";
        String sorted = sortStringAlphabetically(input);
        System.out.println("Input: " + input);
        System.out.println("Sorted Output: " + sorted);
    }

    public static String sortStringAlphabetically(String input) {
        char[] charArray = input.toCharArray();
        Arrays.sort(charArray);
        return new String(charArray);
    }
}
```