05 Feb'23 String in Java Assignment-13

1. WAP(Write a Program) to remove Duplicates from a String.(Take any String example with duplicates character)

Ans:

```
 public static void main(String[] args) {
        String str = "Ballooneenn";
        String str2 = " ";
        char[] arr = str.toCharArray();
        for (int i = 0; i < arr.length; i++) {
            if(str2.indexOf(arr[i]) < 0){
                str2 += arr[i];
            };
        }
        System.out.println(str2);
```

2. WAP to print Duplicates characters from the String

Ans:

```
package Assignment_revision;

public class PrintDuplicates {
    public static void main(String[] args) {

        String str = "Universal Productions";
        int length = str.length();
        char ch[] = str.toCharArray();
        for (int i = 0; i < ch.length; i++) {
            for (int j = i + 1; j < ch.length; j++) {
                if (ch[i] == ch[j]) {
                    System.out.print(ch[j]);
                    break;
                }
            }
        }
    }
}
```

05 Feb'23 String in Java Assignment-13

3. WAP to check if "2552" is palindrome or not.

Ans:

```
package Assignment_revision;

public class Palindrome {
    public static void main(String[] args) {
        String str = "2552";
        String str1 = "";

        for (int i = str.length() - 1; i >= 0; i--) {  //reversing string loop
            str1 += str.charAt(i);   //Adding every character in str1
        }
        if (str.equals(str1)) {   //comparing whether it is a Palindrome or not.
            System.out.println("Given string is PALINDROME");
        } else {
            System.out.println("Given string is not PALINDROME");
        }
    }
}
```

4. WAP to count the number of consonants, vowels, special characters in a String.

Ans:

```
package Assignment_revision;

public class count_ConsonantsVowelsSpecialCharacters {
    public static void main(String[] args) {
        String str = "Abdulaamir9496@gmail.com";

        // Declare the variable vowels, consonant, digit and special characters
        int vowels = 0, consonants = 0, specialCharacters = 0, digits = 0;

        // str.length() function to count number of character in given string.
        for (int i = 0; i < str.length(); i++) {
            char ch = str.charAt(i);
```

05 Feb'23 String in Java Assignment-13

```
            if (ch >= 'a' && ch <= 'z' || ch >= 'A' && ch <= 'Z') {
                str.toLowerCase();   //convert all alphabets into lowercase

                if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
                    vowels++;
                } else
                    consonants++;
            } else if (ch >= '0' && ch <= '9') {
                digits++;
            } else
                specialCharacters++;
        }
        System.out.println("vowels: " + vowels);
        System.out.println("consonants: " + consonants);
        System.out.println("digits: " + digits);
        System.out.println("specialCharacters: " + specialCharacters);
    }
}
```

5. WAP to implement Anagram checking least inbuilt methods being used.

Ans:

```
public class Anagram_LeastInbuiltMethodsUsed {
    public static void main(String[] args) {

        String str = "School Master".replace(" ", " ");
        String str1 = "The classroom".replace(" ", " ");

        char arr1[] = str.toLowerCase().toCharArray();
        char arr2[] = str1.toLowerCase().toCharArray();

        Arrays.sort(arr1);
        Arrays.sort(arr2);
```

05 Feb'23 String in Java Assignment-13

```
        if (Arrays.equals(arr1, arr2)) {
            System.out.println("Given string is ANAGRAM");
        } else {
            System.out.println("Given string is not a ANAGRAM");
        }
    }
```

6. WAP to implement Pangram checking with least inbuilt methods being used.

Ans:

```
package Assignment_revision;

public class pangram_LeastInbuiltMethodsUsed {
    public static void main(String[] args) {

        boolean flag = false;
        String str = "THE QUICK BROWN FOX JUMPS OVER LAZY
DOG".replace(" ", " ");
        char ch[] = str.toCharArray();

        int arr[] = new int[26];

        for (int i = 0; i < ch.length; i++) {
            arr[ch[i] - 65]++;
        }
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == 0) {
                System.out.println("It is not a PANGRAM");
                flag = true;
            }
        }
        if (!flag) {
            System.out.println("It is a PANGRAM");
        }
    }
```

05 Feb'23 String in Java Assignment-13

```
}
```

7. WAP to find if String contains all unique characters.
Ans:

```
package Assignment_revision;

import java.util.Arrays;

public class AllUniqueCharacters {
    public static boolean is_Unique_str(String str) {
        char[] chars = str.toCharArray();
        Arrays.sort(chars);
        for (int i = 1; i < chars.length; ++i) {
            if (chars[i] == chars[i - 1]) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        String str = "xyz";
//        String str = "xyyz";
        System.out.println("Original String : " + str);
        System.out.println("String has all unique characters: " +
is_Unique_str(str));
    }
}
```

8. WAP to find the maximum occurring character in a String.
Ans:

```
package Assignment_revision;

import java.util.*;
import java.util.Arrays;
```

05 Feb'23 String in Java Assignment-13

```
public class MaximumOccuringCharacterIn_String {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String k = in.nextLine();
        char tempArray[] = k.toCharArray();
        Arrays.sort(tempArray);
        String s = new String(tempArray);
        int n = s.length();
        int max_count = 0;
        int count = 1;
        char ans = '-';
        for (int i = 1; i <= n; i++) {
            if ((i == n) || (s.charAt(i) != s.charAt(i - 1))) {
                if (max_count < count) {
                    max_count = count;
                    ans = s.charAt(i - 1);
                }
                count = 1;
            } else {
                count++;
            }
        }
        System.out.println("Maximum occurring character is " + ans);
    }
}
```