String In Java

1.WAP(write a program)to remove duplicate from a string.(take any string example with duplicate character)

```
public class Main {
 public static void main(String[] args) {
  String str1 = "w3resource";
  System.out.println("The given string is: " + str1);
  System.out.println("After removing duplicates characters the new string is: " +
removeDuplicateChars(str1));
 }
 private static String removeDuplicateChars(String sourceStr) {
  char[] arr1 = sourceStr.toCharArray();
  String targetStr = "";
  for (char value: arr1) {
   if (targetStr.indexOf(value) == -1) {
    targetStr += value;
   }
  }
  return targetStr;
 }
}
```

2.WAP to print duplicate character from the string

```
public class DuplicateCharacters {
    public static void main(String[] args) {
      String string1 = "Great responsibility";
      int count;


           char string[] = string1.toCharArray();


      System.out.println("Duplicate characters in a given string: ");


      for(int i = 0; i <string.length; i++) {
         count = 1;
         for(int j = i+1; j <string.length; j++) {
            if(string[i] == string[j] && string[i] != ' ') {
               count++;
                             string[j] = '0';
            }
         }


         if(count > 1 && string[i] != '0')
            System.out.println(string[i]);
      }
   }
```

3.WAP to check if “2552” is palindrome or not.

```
public class Palindrome
{
   public static void main(String[] args)
   {
      String str1="2552";
      String str2="";

      for(int i=str1.length()-1;i>=0;i--)
      {
         str2=str2+str1.charAt(i);
      }
      if(str1.equals(str2))
      {
         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 ,specials characters in a string.

```
public class Main {

public static void main(String[] args) {
String line;
Scanner sc = new Scanner(System.in);
System.out.print(“\nEnter the string : “);
line = sc.nextLine();
int vowels = 0, consonants = 0, digits = 0, spaces = 0, symbols = 0;

line = line.toLowerCase();
for(int i = 0; i < line.length(); ++i)
{
char ch = line.charAt(i);
if(ch == ‘a’ || ch == ‘e’ || ch == ‘i’
|| ch == ‘o’ || ch == ‘u’) {
++vowels;
}
else if((ch >= ‘a’&& ch <= ‘z’)) {
++consonants;
}
else if( ch >= ‘0’ && ch <= ‘9’)
{
++digits;
}
else if (ch ==’ ‘)
{
++spaces;
}
else
++symbols;
}

System.out.println(“Vowels: ” + vowels);
System.out.println(“Consonants: ” + consonants);
System.out.println(“Digits: ” + digits);
System.out.println(“White spaces: ” + spaces);
System.out.println(“Symbols : ” + symbols);
}
}
```

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

```
public class Anagram
{
   public static void main(String[] args)
   {
      String str1="School Master";
      String str2="The Classroom";

      str1=str1.replace(" ", "");
      str2=str2.replace(" ", "");

      str1=str1.toLowerCase();
      str2=str2.toLowerCase();

      char []ar1=str1.toCharArray();
      char []ar2=str2.toCharArray();

      Arrays.sort(ar1);
      Arrays.sort(ar2);

      if(Arrays.equals(ar1, ar2))
      {
         System.out.println("It's an Anagram");
      }
      else
      {
         System.out.println("Its not an Anagram");
      }
   }

}
```

6.WAP to implement pangram checking least inbuilt methods being used

```
public class Pangram
{
    public static void main(String[] args)
    {
       boolean flag=false;
        String str="THE QUICK ROWN FOX JUMPS OVER LAZY DOG";
        str=str.replace(" ", "");
        char []ch=str.toCharArray();

        int ar[]=new int[26];

        for(int i=0;i<ch.length;i++)
        {
            ar[ch[i]-65]++;
        }
        for(int i=0;i<ar.length;i++)
        {
            if(ar[i]==0)
            {
                System.out.println("Its not pangram");
                flag=true;
            }
        }

        if(flag==false)
        {
            System.out.println("Its pangram");
        }



    }

}
```

7.WAP to find if a string contains all unique characters.

```
import java.util.*;

class GfG {
    /* Convert the string to character array
    for sorting */
    boolean uniqueCharacters(String str)
    {
        char[] chArray = str.toCharArray();

        Arrays.sort(chArray);

        for (int i = 0; i < chArray.length - 1; i++) {
            if (chArray[i] != chArray[i + 1])
                continue;

                        else
                return false;
        }
        return true;
    }


    public static void main(String args[])
    {
        GfG obj = new GfG();
        String input = "GeeksforGeeks";

        if (obj.uniqueCharacters(input))
            System.out.println("The String " + input
                            + " has all unique characters");
        else
            System.out.println("The String " + input
                            + " has duplicate characters");
    }
}
```

8.WAP to find the maximum occurring character in a string

```
public class Characters
{
    public static void main(String[] args) {
      String str = "grass is greener on the other side";
      int[] freq = new int[str.length()];
      char minChar = str.charAt(0), maxChar = str.charAt(0);
      int i, j, min, max;

           char string[] = str.toCharArray();

           for(i = 0; i < string.length; i++) {
         freq[i] = 1;
         for(j = i+1; j < string.length; j++) {
            if(string[i] == string[j] && string[i] != ' ' && string[i] != '0') {
               freq[i]++;


               string[j] = '0';
            }
         }
      }


      min = max = freq[0];
      for(i = 0; i <freq.length; i++) {


         if(min > freq[i] && freq[i] != '0') {
            min = freq[i];
            minChar = string[i];
         }
                 if(max < freq[i]) {
            max = freq[i];
            maxChar = string[i];
         }
      }

      System.out.println("Minimum occurring character: " + minChar);
      System.out.println("Maximum occurring character: " + maxChar);
   }
}
```