Strings Assignment

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

```
public class RemoveDuplicates1 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String str = "AaBcDdeeFfGghhiijkloxxyyz";
        str = str.replace(" ", "");
        String str1 = "";

        int arr[] = new int[26];
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) >= 97 && str.charAt(i) <=
122) {
                str1 = str1 + (char) (str.charAt(i) -
32);

            } else {
                str1 = str1 + str.charAt(i);
            }
        }
        for (int i = 0; i < str1.length(); i++) {
            System.out.print(str1.charAt(i) + " ");
        }

        char ch[] = str1.toCharArray();

        for (int i = 0; i < ch.length; i++) {
            int index = ch[i] - 65;
            arr[index]++;
        }

        System.out.println();
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();

        int num = 0;
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == 1) {
                num = i;
                char ch1 = (char) (num + 65);
//                 System.out.println(arr[i]+" "+ch1+"
");
                System.out.print(ch1 + " ");
            }
        }
        System.out.print("These are not the  duplicate
characters in string");

    }

}
```

2. WAP to print Duplicates characters from the String.

```
package stringsAssignment;

public class printDuplicateCharacters2 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String str = "AaBcDdeeFfGghhiijkloxxyyz";
        String str1 = "";
        int arr[] = new int[26];
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) >= 97 && str.charAt(i) <=
122) {
                str1 = str1 + (char) (str.charAt(i) -
32);

            } else {
                str1 = str1 + str.charAt(i);
            }
        }

        char ch[] = str1.toCharArray();

        for (int i = 0; i < ch.length; i++) {
            int index = ch[i] - 65;
            arr[index]++;
        }

        System.out.println();
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
        System.out.println();

        int num = 0;
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] > 1) {
                num = i;
                char ch1 = (char) (num + 65);
//                       System.out.println(arr[i]+"
"+ch1+" ");
                System.out.print(ch1 + " ");
            }
        }
        System.out.print("These are all the duplicate
characters in string");


    }

}
```

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

```java
package stringsAssignment;

public class StringPalindrome3 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String str = "2552";
        String str1 = "";

        for (int i = str.length() - 1; i >= 0; i--) {
            str1 = str1 + str.charAt(i);
        }
        System.out.println(str1);

        if (str.equals(str1)) {
            System.out.println("It is a palindrome");
        } else {
            System.out.println("It's not a palindrome");
        }

    }

}
```

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

```java
package stringsAssignment;

public class SpecialCharacters4 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
       String str="123456aabcDEABCDE#$%";
     // String str1="";
```

```
       int count=0;

       for(int i=0;i<str.length();i++)
       {
    if((str.charAt(i)>=65 && str.charAt(i)<=90) ||
(str.charAt(i)>=97 && str.charAt(i)<=122) ||
            (str.charAt(i)>=32 &&
str.charAt(i)<=47)) {
        count++;
       }
       }
       System.out.println(str.length());
       System.out.println(count);
    }

}
```

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

```
  package stringsAssignment;

import java.util.Arrays;

public class AnagramString5 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String s1="Race";
        String s2="Care";
        String s3="";
        String s4="";
        for(int i=0;i<s1.length();i++) {
            if(s1.charAt(i)>=65 && s1.charAt(i)<=90) {
                s3=s3+(char) (s1.charAt(i)+32);
            }
            else
            {
                s3=s3+s1.charAt(i);
            }
```

```java
}

for(int i=0;i<s2.length();i++) {
    if(s2.charAt(i)>=65 && s2.charAt(i)<=90)
    {
        s4=s4+(char) (s2.charAt(i)+32);
    }
    else
    {
        s4=s4+s2.charAt(i);
    }
}

char ch1[]=s3.toCharArray();
char ch2[]=s4.toCharArray();


for(int i=0;i<ch1.length;i++) {
    System.out.print(ch1[i]+" ");
}

System.out.println();

Arrays.sort(ch1);
Arrays.sort(ch2);

for(int i=0;i<ch1.length;i++) {
    System.out.print(ch1[i]+" ");
}
System.out.println();
for(int i=0;i<ch2.length;i++) {
    System.out.print(ch2[i]+" ");
}
System.out.println();

if(Arrays.equals(ch1, ch2)) {
    System.out.println("It's an anagram");
}
else {
    System.out.println("It's not an anagram ");
}
```

```
         System.out.println();
         //System.out.println(s3);
         //System.out.println(s4);



     }

}
```

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

```
package stringsAssignment;

public class PangramString5 {

     public static void main(String[] args) {
         // TODO Auto-generated method stub
    String s1="the quick brown fox jumps over the lazy 
dog";
    s1=s1.replace(" ","");
    boolean flag=false;

    char ch[]=s1.toCharArray();
    int [] ar=new int[26];

    for(int i=0;i<ch.length;i++) {
     int index=ch[i]-97;
     ar[index]++;
     System.out.print(ar[index]+" ");
    }
    System.out.println();
    for(int i=0;i<ar.length;i++) {
     if(ar[i]==0) {
         System.out.println("It,s not pangram");
         flag=true;
     }
    }
    System.out.println();
    if(flag==false) {
     System.out.println("It's a pangram");
    }
     }

}
```

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

```
package stringsAssignment;

public class UniqueCharacters7 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        String str = "abcdefghi";
        String str1 = "";
        boolean flag=false;
        int arr[] = new int[26];
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) >= 97 && str.charAt(i) <=
122) {
                str1 = str1 + (char) (str.charAt(i) -
32);

            } else {
                str1 = str1 + str.charAt(i);
            }
        }

        char ch[] = str1.toCharArray();

        for (int i = 0; i < ch.length; i++) {
            int index = ch[i] - 65;
            arr[index]++;
        }

        System.out.println();
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
        int num = 0;
        for (int i = 0; i < arr.length;i++) {
            if (arr[i]>1) {

                System.out.print("It's not  a unique 
characters string ");
                flag=true;
            }
        }

            if(flag==false) {
                System.out.print("It's a string which 
contains unique characters ");

            }

    }

}
```

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

```
package stringsAssignment;

public class MaximumOccuringCharcater8 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        String str = 
"AaaBbbbbbbbcCCdefghiaXXxxxxxxxxYYYYZZZZZZZzZ";
        String str1 = "";
        boolean flag=false;
        int arr[] = new int[26];
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) >= 97 && str.charAt(i) <= 
122) {
                str1 = str1 + (char) (str.charAt(i) - 
32);

            } else {
                str1 = str1 + str.charAt(i);
            }
        }

        char ch[] = str1.toCharArray();
        for(int i=0;i<ch.length;i++) {
            System.out.print(ch[i]+" ");
        }

        for (int i = 0; i < ch.length; i++) {
            int index = ch[i] - 65;
            arr[index]++;
        }

        System.out.println();
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();

        int num=0;
        int max=0;

        for(int i=0;i<arr.length;i++) {
            if(arr[i]>max) {
                max=arr[i];
                num=i;
            }
        }
        char ch1=(char) (num+65);
        System.out.println(ch1+" is repeating "+max+"
times");


    }

}
```