Q1. WAP to remove duplicates from a string.

Ans.

```
public class Duplicates_char {
 public static void main(String args[])
    {
        String name="Mohit Jangid";
        System.out.println("Original String: "+name);
        int k=0,j=0,count=1;
        for(j=0;j<name.length();j++)
        {
            for(k=j+1;k<name.length();k++)
            {
                if(name.charAt(j)==name.charAt(k))
                {
                count++;
            }
            if(count>1)
            {
                name=name.replace(name.charAt(j),' ');

                count=1;
            }

            }
      }
      System.out.println("Deleted String: "+name);

}
}
Output:
Original String: Mohit Jangid
Deleted String: Moh t Jang d
```

Q2. WAP to print duplicates characters from the string.

Ans. `class Duplicates_char {`

```
    public static void main(String args[])
    {
        String name="great Responsibility";
        String cpy=name;
```

```
        System.out.println("Original String: "+name);
        System.out.print("Number of Duplicate Characters: ");
        int k=0,j=0,count=0;
        for(j=0;j<name.length();j++)
        {
            for(k=j+1;k<name.length();k++)
            {
                if(name.charAt(j)==name.charAt(k))
                {
                    count++;

            }

            }
            if(count>0)
            {
                System.out.print(cpy.charAt(j)+" ");
                count=0;
            }
      }


}
}
Output:
Original String: great Responsibility
Number of Duplicate Characters: e t s i i
```

Q3. WAP to check if “2552” is a palindrome or not.

Ans.

```
public class Palindrome {
        public static void main(String args[])
        {
        String s1="2552";
        String s2="";
        int length= s1.length();
        for(int i=length-1;i>=0;i--)
        {
            char s=s1.charAt(i);
            s2=s2+s;
```

```
        }
        System.out.println(s2);
        if(s1.equals(s2))
        System.out.println("PALINDROME NUMBER");
        else
        System.out.println("Not Palindrome");



        }

    }
Output:
PALINDROME NUMBER
```

Q4. WAP to count the number of consonants, vowels, special characters in a string.

Ans.

```
public class ConsVowSpec {
    public static void main(String args[])
    {
        int i = 0,length,count=0,special=0;
        String str = "~{\\/[Mohit@_Jangid]#^}`";
        System.out.println("String Length: "+str.length());
        length=str.length();
        for (i = 0; i < str.length(); i++) {
            if (str.charAt(i) == 'a'|| str.charAt(i) == 'e' ||
str.charAt(i) == 'i' || str.charAt(i) == 'o'
                    || str.charAt(i) == 'u'
                    ||str.charAt(i) == 'A'|| str.charAt(i) == 'E' ||
str.charAt(i) == 'I' || str.charAt(i) == 'O'
                    || str.charAt(i) == 'U') {
                count++;

            }
            if(str.charAt(i) =='!'||str.charAt(i) =='"'||str.charAt(i)
=='#'||str.charAt(i) =='$'||
               str.charAt(i) =='%'||str.charAt(i) =='&'|| str.charAt(i)
=='('||
```

```
                str.charAt(i) ==')'||str.charAt(i) =='*'||str.charAt(i)
=='+'||
                str.charAt(i) ==','||str.charAt(i) =='-'||str.charAt(i)
=='.'||
                str.charAt(i) =='/'||str.charAt(i) =='}'||str.charAt(i)
==';'||
                str.charAt(i) =='<'||str.charAt(i) =='='||str.charAt(i)
=='>'||
                str.charAt(i) =='_'||str.charAt(i) =='@'||str.charAt(i)
=='['||
                str.charAt(i) ==']'||str.charAt(i) =='?'||str.charAt(i)
=='{'||
                str.charAt(i) =='^'||str.charAt(i) =='\\'||str.charAt(i)
=='~'||str.charAt(i) =='`')
                special++;
            }
            System.out.println("Vowels: "+count);
            System.out.println("Special: "+special);
            System.out.println("Consonants: "+(length-count));
        }

}
OUTPUT:
String Length: 23
Vowels: 4
Special: 12
Consonants: 19
```

Q5. WAP to implement Anagram Checking with least inbuilt methods being used.

Ans.

```
import java.util.Arrays;

class Anagram {
public static void main(String args[])
{
    String s1="PEEK";//String1
    String s2="KEEP";//String2
```

```
    //Step1:Removing the spaces using replace method
    s1=s1.replace(" ","");
    s2=s2.replace(" ","");

    //Step2:Converting in either uppercase or lowercase
    s1=s1.toUpperCase();
    s2=s2.toUpperCase();

    //Step3:Converting into charactar array using tochararray
    char ch1arr[]=s1.toCharArray();
    char ch2arr[]=s2.toCharArray();

    //Step4:Sorting array using Array.sort(arrayname)
    Arrays.sort(ch1arr);
    Arrays.sort(ch2arr);

    //Check if all charactar in s1 and s2 are same or not
    if(Arrays.equals(ch1arr, ch2arr))
    {
        System.out.println("ANAGRAM STRING");

    }else
    System.out.println("NOT an ANAGRAM STRING");
}

}
Output:
ANAGRAM STRING
```

Q6. WAP to implement Pangram Checking with least inbuilt methods being used.

Ans.

```
class Pangram {
    public static void main(String args[])
    {
        String s1="ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    //Step1:Removing the spaces using replace method
    s1=s1.replace(" ","");
```

```
    //Step2:Converting in either uppercase or lowercase
    s1=s1.toUpperCase();


    //Step3:Converting into charactar array using tochararray
    char ch1arr[]=s1.toCharArray();


    //Step4:Create an empty array of total size of 26 Alphabets
    int empty[]=new int[26];

    /*Now traversing string array and USING ASCII value of char we assign
    increment the value of EMPTY ARRAY
    */
    for(int i=0;i<ch1arr.length;i++)
    {
      empty[ch1arr[i]-65]++;//-65forCapital or -90 for small
    }
    boolean flag=false;
   /*Now we are traverse the EMPTY array and checks if any
    value is zero then we assign flag to TRUE
    */
    for(int i=0;i<empty.length;i++)
    {
      if(empty[i]==0)
      {flag=true;
      }
    }
    if(flag==false)
    System.out.println("PANGRAM");
    else
    System.out.println("NOT PANGRAM");
    }

}
Output:
PANGRAM
```

Q7. WAP to find if String contains all unique characters.

Ans. `public class Unique_char_str {`

```
    public static void main(String args[])
    {
        int count=0,j=0,i=0;
        String str = "Mohit Jangid";
        for (i = 0; i < str.length(); i++) {
        for(j=i+1;j<str.length();j++)
        {
          if(str.charAt(i)==str.charAt(j))
          {
            count++;
          }

        }
        }

        if(count==0)
        System.out.println("Unique Character String");
       else
       System.out.println("Not a Unique Character String");
    }

}
Output:
Not a Unique Character String
```

Q8. WAP to find the maximum occurring character in a string.

Ans.

```
import java.util.Scanner;
public class Maximum_char_str
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        System.out.println("Finding maximum character occurred in a String
");
        int count = 1, max = 1, i, j;
        char ch=' ';
        System.out.println("Enter String with at least one maximum
character");
        String str=scan.nextLine();
```

```
        System.out.println("String: "+str);
        for (i = 0; i < str.length(); i++)
        {
            for (j = i + 1; j < str.length(); j++)
             {
                if (str.charAt(i) == str.charAt(j))
                {
                    count++;
                    if(count>max)
                    {
                        max=count;
                        ch=str.charAt(i);

                    }

                }

            }
           count=1;
        }
        if(max==1)
        System.out.println("Each characters occurred once");
        else
        System.out.println("Maximum occurred character is :
"+ch+"\n"+max+" times");
    }
}
```

```
Output:
Finding maximum character occurred in a String
Enter String with at least one maximum character
Mohit Jangid
String: Mohit Jangid
Maximum occurred character is : i
2 times
```