Strings in Java

1.WAP to remove duplicates from the string

```
public class Main {

   public static void remDuplicates(char s[], int n)
   {
      String ans="";
      int i,j;
      for(i=0;i<n;i++){
         for(j=0;j<i;j++){
            if(s[i]==s[j]){
               break;
            }

         }
         if(j==i){
            ans+=s[i];
         }
      }
      System.out.print(ans);
   }
   public static void main(String[] args)
   {
      char s[] = "aaabaababbccbccd".toCharArray();
      int n = s.length;
      removeDuplicates(s, n);
   }
}
```

2.WAP to write a duplicate characters from the string

```
public class Examp {
  public static void main(String argu[]) {
    String str = "bus journey is best";
    char[] carray = str.toCharArray();
    System.out.println("The string is:" + str);
    System.out.print("Duplicate Characters in above string are: ");
    for (int i = 0; i < str.length(); i++) {
      for (int j = i + 1; j < str.length(); j++) {
        if (carray[i] == carray[j]) {
          System.out.print(carray[j] + " ");
          break;
        }
```

```
      }
    }
  }
}
```

3.WAP to check if 2552 is palindrome or not

```
class PalindromeExamp{
 public static void main(String args[]){
  int r,sum=0,temp;
  int n=454;//It is the number variable to be checked for palindrome

  temp=n;
  while(n>0){
   r=n%10;  //getting remainder
  sum=(sum*10)+r;
   n=n/10;
  }
  if(temp==sum)
  System.out.println("palindrome number ");
  else
   System.out.println("not palindrome");
}
}
```

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

```
public class JavaExamp {

   public static void main(String[] args) {
      String str = "BookReading";
      int vcount = 0, ccount = 0, cspl=0;

      //converting all the chars to lowercase
      str = str.toLowerCase();
      for(int i = 0; i < str.length(); i++)
           {
           char ch = str.charAt(i);
           if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u')
           { vcount++; } else if((ch >= 'a'&& ch <= 'z'))
           {
            ccount++;
         }
```

```
            else {
                        cspl++;
                    }
        }
        System.out.println("Number of Vowels: " + vcount);
        System.out.println("Number of Consonants: " + ccount);
        System.out.println("Number of Special Characters: " +cspl );

    }
}
```

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

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

class AnagramExam {
    static boolean areAnagram(char[] str1, char[] str2)
    {
        int n1 = str1.length;
        int n2 = str2.length;

        if (n1 != n2)
            return false;

        Arrays.sort(str1);
        Arrays.sort(str2);

         for (int i = 0; i < n1; i++)
            if (str1[i] != str2[i])
                return false;
         return true;
    }

    public static void main(String args[])
    {
        char str1[] = { 'g', 'r', 'a', 'm' };
        char str2[] = { 'a', 'r', 'm' };

        if (areAnagram(str1, str2))
            System.out.println("The two strings are"
                        + " anagram of each other");
        else
```

```
        System.out.println("The two strings are not" + " anagram of each other");
    }
}
```

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

```
public class PangramExam {
  public static void main(String[] args) {
    String str = "The quick brown fox jumps over the lazy dog";
    boolean[] alphaList = new boolean[26];
    int index = 0;
    int flag = 1;
    for (int i = 0; i < str.length(); i++) {
      if ( str.charAt(i) >= 'A' && str.charAt(i) <= 'Z') {
        index = str.charAt(i) - 'A';
      }else if( str.charAt(i) >= 'a' && str.charAt(i) <= 'z') {
        index = str.charAt(i) - 'a';
    }
    alphaList[index] = true;
  }
  for (int i = 0; i <= 25; i++) {
    if (alphaList[i] == false)
    flag = 0;
  }
  System.out.print("String: " + str);
  if (flag == 1)
    System.out.print("The above string is a pangram.");
  else
    System.out.print("The above string is not a pangram.");
  }
}
```

7.WAP to find if string contains all unique characters

```
import java.util.*;

class UniqueChar {
        boolean uniqueCharacters(String str)
        {

                for (int i = 0; i < str.length(); i++)
                        for (int j = i + 1; j < str.length(); j++)
                                if (str.charAt(i) == str.charAt(j))
                                        return false;
```

```
		return true;
	}
	public static void main(String args[])
	{
		Astr obj = new Astr();
		String input = "cloudoncloud";

		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 maximum occurring character in a string

```
public class MaxChar {
	static final int ASCII_SIZE = 256;
	static char getMaxOccurringChar(String str)
	{
		int count[] = new int[ASCII_SIZE];

		int len = str.length();
		for (int i = 0; i < len; i++)
			count[str.charAt(i)]++;

		int max = -1;
		char result = ' ';
		for (int i = 0; i < len; i++) {
			if (max < count[str.charAt(i)]) {
				max = count[str.charAt(i)];
				result = str.charAt(i);
			}
		}

		return result;
	}
	public static void main(String[] args)
	{
		String str = "sample string";
		System.out.println("Max occurring character is "+ getMaxOccurringChar(str));
	}
}
```