Q1 ->write a program to remove duplicates from the string .

Ans -> **import** java.util.Scanner;

**public** **class** duplicate {

**public** **static** **void** main(String[] args) {

System.***out***.println("Enter the String :");

Scanner str = **new** Scanner(System.***in***);

String str1 = str.nextLine();

StringBuilder str2 = **new** StringBuilder();

**char**[] arr = str1.toCharArray();

**for**(**int** i = 0; i<arr.length; i++) {

**boolean** flag = **false** ;

**for**(**int** j =i+1; j<arr.length; j++) {

**if**(arr[i] == arr[j]) {

flag = **true**;

**break**;

}

}

**if** (!flag) {

str2.append(arr[i]);

}

}

System.***out***.println("String After removing duplicates : "+ str2);

}

}

Q2 -> write a program to print duplicates character from a string .

Ans -> **import** java.util.Scanner;

**public** **class** duplicate3 {

**public** **static** **void** main(String[] args) {

Scanner sb = **new** Scanner(System.***in***);

String str = sb.next();

**int**[] count = **new** **int**[256];

**for** (**int** i=0; i<str.length() ; i++) {

count[str.charAt(i)]++;

}

**for** (**int** i =0; i<256; i++)

{

**if** (count[i]>1) {

System.***out***.println((**char**)(i) + " count = "+ count[i]);

}

}

}

}

Q3 .-> Write a program to check if a string “2552” is palindrome or not ?

Ans - **public** **class** palindrome2 {

**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("String is palindrome ");

}

**else** {

System.***out***.println("String is not palindrome");

}

}

}

Q4 - write a program to count the number of consonants , vowels and special characters in a string .

Ans - **public** **class** count {

**static** **void** countforcharacter(String str) {

**int** vowels= 0, consonants = 0, specialchar =0, digit =0;

**for** (**int** i =0; i<str.length(); i++) {

**char** ch = str.charAt(i);

**if** (ch>='a'&& ch<='z'||ch>='A'&& ch<='Z') {

ch = Character.*toLowerCase*(ch);

**if** (ch=='a'|| ch=='e'|| ch=='i'|| ch=='o'||ch=='u')

vowels++;

**else**

consonants++;

**if** (ch>='0' && ch<='9')

digit++;

**else**

specialchar++;

}

}

System.***out***.println("vowels :"+ vowels);

System.***out***.println("consonants :"+ consonants);

System.***out***.println("digits :"+ digit);

System.***out***.println("special char :"+ specialchar);

}

**public** **static** **void** main(String[] args) {

String str = "my self murtza";

*countforcharacter*(str);

}

}

Q5 ->write a program to implement anagram checking least inbuilt methods being used.

Ans -> **public** **class** anagram {

**public** **static** **void** main(String[] args) {

String str1 = "keep";

String str2 = "peek";

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

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

**char** []ar1 = str1.toCharArray();

**char**[]ar2 = str2.toCharArray();

Arrays.*sort*(ar1);

Arrays.*sort*(ar2);

**if** (Arrays.*equals*(ar1,ar2)) {

System.***out***.println(" It is a anagram");

}

**else** {

System.***out***.println("It is not an anagram");

}

}

}

Q6 - write a program to implement a panagram checking least inbuilt methods being used .

Ans -**public** **class** panagram {

**public** **static** **void** main (String[] args) {

**boolean** flag = **false**;

String str = "THE QUICK ROWEN FOX JUMPS OVER LAZY DOG ";

str = str.replace(" ", "");

**char** []ch = str.toCwrite a program to find if string contains all unique characters .

harArray();

**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("It is not a panagram");

flag = **true** ;

**if** (flag==**false**) {

System.***out***.println("It is a panagram");

}

}

}

}

}

Q7- write a program to find if string contains all unique characters or not .

Ans - **public** **class** unique {

**public** **static** **boolean** isUnique(String str) {

**int**[] charcount = **new** **int**[256];

**for** (**int** i=0; i<str.length(); i++) {

**char** c = str.charAt(i);

**if** (charcount[c] != 0) {

**return** **false** ;

}

charcount[c]++;

}

**return** **true** ;

}

**public** **static** **void** main(String[] args) {

String str ="my name is murtza";

**boolean** uni = *isUnique*(str);

**if** (uni) {

System.***out***.println("All characters are unique");

}

**else** {

System.***out***.println("All characters are not unique");

}

}

}

Q8 - write a program to find maximum occurrence character in a string .

Ans - **public** **class** maxcount {

**public** **static** **void** main(String[] args) {

String str = "Hello, World!";

**char** maxOccurringChar = *findMaxOccurrence*(str);

System.***out***.println("The maximum occurring character is: " + maxOccurringChar);

}

**public** **static** **char** findMaxOccurrence(String str) {

**int**[] charCount = **new** **int**[256];

**for** (**int** i = 0; i < str.length(); i++) {

**char** c = str.charAt(i);

charCount[c]++;

}

**char** maxChar = ' ';

**int** maxCount = 1;

**for** (**int** i = 0; i < charCount.length; i++) {

**if** (charCount[i] > maxCount) {

maxCount = charCount[i];

maxChar = (**char**) i;

}

}

**return** maxChar;

}

}