1. Write a program to remove duplicates from a String.

Ans: import java.util.Scanner;

public class Question1{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter a string:");

String duplicateString = sc.nextLine();

String result = "";

result = result + duplicateString.charAt(0);

for(int i = 1; i < duplicateString.length(); i++){

String characterLower = "" + duplicateString.charAt(i);

characterLower = characterLower.toLowerCase();

String characterUpper = characterLower.toUpperCase();

if(!(result.contains(characterLower) || result.contains(characterUpper))){

result = result + duplicateString.charAt(i);

}

}

System.out.println(result);

}

}

1. Write a program to print duplicate characters from a string.

Ans: import java.util.Scanner;

public class Question2{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter a string:");

String duplicateString = sc.nextLine();

String result = "";

result = result + duplicateString.charAt(0);

System.out.println("The duplicate characters in the String: ");

for(int i = 1; i < duplicateString.length(); i++){

String characterLower = "" + duplicateString.charAt(i);

characterLower = characterLower.toLowerCase();

String characterUpper = characterLower.toUpperCase();

if(!(result.contains(characterLower) || result.contains(characterUpper))){

result = result + duplicateString.charAt(i);

}

else{

System.out.println(duplicateString.charAt(i));

}

}

}

}

1. Write a program to check if “2552” is palindrome or not.

Ans: public class Question3 {

public static boolean palindrome(String value){

int low = 0;

int high = value.length() - 1;

while (low < high) {

if(value.charAt(low) != value.charAt(high)){

return false;

}

low++;

high--;

}

return true;

}

public static void main(String[] args) {

String value = "2552";

if(palindrome(value)){

System.out.println("Palindrome.");

}

else{

System.out.println("Not a palindrome.");

}

}

}

1. Write a program to count number of consonents, vowels, special characters in a String.

Ans: import java.util.Scanner;

public class Question4 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String vowels = "aeiou";

String consonents = "bcdfghjklmnpqrstvwxyz";

int countVowels = 0;

int countConsonents = 0;

int countSpecial = 0;

System.out.print("Enter a string: ");

String value = sc.nextLine().toLowerCase();

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

String characterAt = "" + value.charAt(i);

if(vowels.contains(characterAt)){

countVowels++;

}

else if(consonents.contains(characterAt)){

countConsonents++;

}

else{

countSpecial++;

}

}

System.out.println("Number of vowels: " + countVowels);

System.out.println("Number of consonents: " + countConsonents);

System.out.println("Number of special characters: " + countSpecial);

}

}

1. Write a program to implement anagram checking using least number of built-in methods.

And: import java.util.Scanner;

public class Question5 {

public static boolean Anagram(String s1, String s2){

if (s1.length() == s2.length()) {

for (int i = 0; i < s1.length(); i++) {

String ch = "" + s1.charAt(i);

if(s2.contains(ch)){

continue;

}

else{

return false;

}

}

return true;

}

else {

return false;

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter string1: ");

String s1 = sc.nextLine();

System.out.print("Enter string2: ");

String s2 = sc.nextLine().replace(" ", "");

if(Anagram(s1.toLowerCase(), s2.toLowerCase())){

System.out.println("Anagram");

}

else{

System.out.println("Not anagram");

}

}

}

1. Write a program to implement Pangram checking with least built-in methods used.

Ans: import java.util.Scanner;

public class Question6 {

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

boolean flag = false;

System.out.print("Enter string: ");

String str = sc.nextLine().toUpperCase().replace(" ", "");

char ch[] = str.toCharArray();

int arr[] = new int[26];

for(int i = 0; i < ch.length; i++)

{

arr[ch[i] - 65]++;

}

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

{

if(arr[i] == 0)

{

System.out.println("Its not pangram");

flag = true;

break;

}

}

if(flag == false)

{

System.out.println("Its pangram");

}

}

}

1. Write program to find maximum occurring character in a String.

Ans: import java.util.Scanner;

public class Question7 {

public static boolean checkAllUnique(String s){

String temp = "" + s.charAt(0);

for (int i = 1; i < s.length(); i++) {

String chAt = "" + s.charAt(i);

if(temp.contains(chAt)){

return true;

}

temp = temp + s.charAt(i);

}

return false;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter a string: ");

String str = sc.nextLine().toLowerCase().replace(" ", "");

if(checkAllUnique(str)){

System.out.println("Contains Duplicates.");

}

else{

System.out.println("Don't contains duplicates");

}

}

}