/\*1. How do you swap two numbers without using a third variable in Java?

package com.gqt.strings.project;

public class String1 {

public static void main(String[] args) {

int a=1; int b=2; a = a + b; b = a - b; a = a - b;

System.out.println("The value of a is: "+a);

System.out.println("The value of b is: "+b);

}

}

\*/

/\*2.Write a Java program to print a Fibonacci sequence using recursion.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the range upto which you need fibonacci series"); int n = sc.nextInt(); int prev = 0; int present = 1;

if (n==1) {

System.out.println(0);

}

else {

System.out.print("0 1 "); for (int i = 2; i < n ; i++) { int next = prev + present; prev = present; present = next;

System.out.print(next+" ");

}

}

}

}

\*/

/\*3. How do you check if an array of integers contains only odd numbers in Java?

package com.gqt.strings.project;

public class String1 {

public static void main(String[] args) { int [] a= {1,2,3,4,5,6,7,8,9}; System.out.println("ODD NUMBERS: "); for (int i = 0; i < a.length; i++) {

if (a[i]%2!=0) {

System.out.println(a[i]);

}

}

System.out.println("EVEN NUMBERS: "); for (int i = 0; i < a.length; i++) {

if (a[i]%2==0) {

System.out.println(a[i]);

}

}

}

}

\*/

/\*4. How do you remove spaces from a string in Java?

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string");

String s1 = sc.nextLine(); String s2 = "";

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

{

if(s1.charAt(i)==' ')

{

s1=s1.replaceAll(" ",s2);

}

}

System.out.println(s1);

}

}

\*/

/\*5. How can you find the factorial of an integer in Java?

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number to find the factorial"); int n = sc.nextInt(); int fact = 1 ; if (n==0) {

System.out.println(0);

}

else if (n==1) {

System.out.println(1);

}

else {

for (int i = n; i >=1; i--) { fact = fact \* i;

}

System.out.println(fact);

}

}

}

\*/

/\*6. Write a java program to find out and print the longest word present in the sentence.

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String args[])

{

Scanner scanner=new Scanner(System.in);

System.out.println("Enter the string");

String s1=scanner.nextLine();

String small="",large="",word=""; String words[]=new String[100]; int length=0; s1=s1+"";

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

{

if(s1.charAt(i)==' ')

{

words[length]=word; length++; word="";

}

else

{

word=word+s1.charAt(i);

}

}

small=large=words[0];

for(int k=0;k<length;k++)

{

if(small.length()>words[k].length())

{

small=words[k];

}

if(large.length()<words[k].length())

{

large=words[k];

}

}

System.out.println("Smallest word "+small);

System.out.println("Largest word"+large);

}

}

\*/

/\*7. Write the code in the findLongestWord() method which accepts a string and returns the longest word

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String args[])

{

Scanner scanner=new Scanner(System.in);

System.out.println("Enter the string");

String s1=scanner.nextLine();

String small="",large="",word=""; String words[]=new String[100];

int length=0;

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | s1=s1+"";  { | for(int i=0;i<s1.length();i++)  if(s1.charAt(i)==' ')  {  words[length]=word; length++; word="";  } else  {  word=word+s1.charAt(i);  } |
|  |  | }  small=large=words[0];  for(int k=0;k<length;k++)  {  if(small.length()>words[k].length())  {  small=words[k];  }  if(large.length()<words[k].length())  {  large=words[k];  }  }  System.out.println("Smallest word "+small);  System.out.println("Largest word"+large); | |
| } | } |  | |

\*/

/\*8. Write a java program to take a string as user input and returns the count of uppercase letters and lowercase letters.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the sentence of your wish"); String s1 = sc.nextLine(); int Upper = 0; int Lower = 0;

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

char ch = s1.charAt(i); if (ch>='A'&&ch<='Z') {

Upper++;

}

else if (ch>='a'&&ch<='z') {

Lower++;

}

}

System.out.println("The num of Upper case letters are: "+Upper);

System.out.println("The num of Lower case letters are: "+Lower);

}

}

\*/

/\*9. Given an array of integers and a number k, write a function that returns true if given array can be divided into pairs such that sum of every pair is divisible by k.

Example :

Input : arr = [9, 5, 7, 3], k = 6

Output: True

Explanation: {(9, 3), (5, 7)} is a possible solution. 9 + 3 = 12 is divisible by 6 and 7 + 5 = 12 is also divisible by 6.

package com.gqt.strings.project;

public class String1 {

public static boolean canDivideArray(int[] arr, int k) { // Initialize an array to store the counts of remainders int[] remainderCounts = new int[k];

// Count the remainders of elements in arr when divided by k for (int num : arr) {

int remainder = (num % k + k) % k; // Ensure remainder is non-negative remainderCounts[remainder]++;

}

// Check if remainders can be paired up if (remainderCounts[0] % 2 != 0) {

// If there is an odd count of elements with remainder 0, return false

return false;

}

for (int i = 1; i <= k / 2; i++) {

// For other remainders, check if their counts match with their complements if (remainderCounts[i] != remainderCounts[k - i]) {

return false;

}

}

return true;

}

public static void main(String[] args) { int[] arr = {9, 5, 7, 3};

int k = 6;

boolean result = canDivideArray(arr, k);

System.out.println("Output: " + result); // Output: True

}

}

\*/

/\*10. Given an unsorted array Arr of size N of positive integers. One number 'A' from set {1,

2,....,N} is missing and one number 'B' occurs twice in array. Find these two numbers.

Example 1 :

Input:N = 2

Arr[] = {2, 2}

Output: 2 1

Explanation: Repeating number is 2 and smallest positive missing number is 1.

Example 2:

Input:N = 3

Arr[] = {1, 3, 3}

Output: 3 2

Explanation: Repeating number is 3 and smallest positive missing number is 2.

package com.gqt.strings.project;

public class String1 {

public static void findNumbers(int[] arr) { int n = arr.length; int repeating = -1; int missing = -1;

// Traverse the array for (int i = 0; i < n; i++) { int index = Math.abs(arr[i]) - 1;

// If the current element at index is negative, it means it has been seen before

(repeating)

if (arr[index] < 0) { repeating = Math.abs(arr[i]);

} else {

arr[index] = -arr[index];

}

}

// Find the missing number for (int i = 0; i < n; i++) {

if (arr[i] > 0) { missing = i + 1; break;

}

}

System.out.println("Repeating Number: " + repeating);

System.out.println("Missing Number: " + missing);

}

public static void main(String[] args) { int[] arr1 = {2, 2};

findNumbers(arr1); // Output: Repeating Number: 2, Missing Number: 1

int[] arr2 = {1, 3, 3};

findNumbers(arr2); // Output: Repeating Number: 3, Missing Number: 2

}

}

\*/

/\*11. Given a string S. The task is to print all unique permutations of the given string in lexicographically sorted order.

Example1: Input: ABC

Output:

ABC ACB BAC BCA CAB CBA

Explanation:

Given string ABC has permutations in 6 forms as ABC, ACB, BAC, BCA, CAB and CBA .

Example 2: Input: ABSG Output:

ABGS ABSG AGBS AGSB ASBG ASGB BAGS

BASG BGAS BGSA BSAG BSGA GABS GASB GBAS GBSA GSAB GSBA SABG SAGB SBAG

SBGA SGAB SGBA

Explanation:

Given string ABSG has 24 permutations.

package com.gqt.strings.project; import java.util.\*; public class String1 {

public static void main(String[] args) {

String input = "ABSG"; generateUniquePermutations(input);

}

public static void generateUniquePermutations(String input) { char[] chars = input.toCharArray();

Arrays.sort(chars); // Sort the input string for lexicographic order boolean[] used = new boolean[chars.length]; char[] output = new char[chars.length]; generatePermutations(chars, output, used, 0);

}

public static void generatePermutations(char[] chars, char[] output, boolean[] used, int level) { if (level == chars.length) {

System.out.println(new String(output));

return;

}

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

if (!used[i]) { output[level] = chars[i]; used[i] = true;

generatePermutations(chars, output, used, level + 1);

used[i] = false;

// Skip duplicates

while (i < chars.length - 1 && chars[i] == chars[i + 1]) { i++;

}

}

}

}

}

\*/

/\*12. Write a Java Program to Check Leap Year?

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

int year;

System.out.println("Enter the Year :"); Scanner sc = new Scanner(System.in); year = sc.nextInt();

if (((year % 4 == 0) && (year % 100!= 0)) || (year%400 == 0)) System.out.println("Specified year is a leap year");

else

System.out.println("Specified year is not a leap year");

}

}

\*/

/\*13. Write Java Program to Display Armstrong Number Between Two Intervals?

package com.gqt.strings.project; import java.util.Scanner; public class String1 {

public static void main(String[] args) { int num1, num2;

Scanner sc = new Scanner(System.in); System.out.println("Enter the first number :"); num1 = sc.nextInt();

System.out.println("Enter the second number :"); num2 = sc.nextInt();

for (int i = num1; i<num2; i++){ int check, rem, sum = 0;

check = i;

while(check != 0) { rem = check % 10;

sum = sum + (rem \* rem \* rem); check = check / 10;

}

if(sum == i){

System.out.println(i+" is an Armstrong number.");

}

}

}

}

\*/

/\*14. Write a program to find the first non-repeating character in a given String?

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter a string"); String s1 = sc.next(); for (char i: s1.toCharArray()) { if (s1.indexOf(i)==s1.lastIndexOf(i)) {

System.out.println("The first non repeating character is: "+i);

break;

}

else {

System.out.println("There is no non repeating character in this string");

break;

}

}

}

}

\*/

/\*15. Write a program to remove the character in a string

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string");

String s1 = sc.nextLine();

System.out.println(s1);

System.out.println("Enter the character u want to remove"); String s2 = sc.next(); s1 = s1.replace(s2, "");

System.out.println("The string after removing the character");

System.out.println(s1);

}

}

\*/

/\*16. Determine the array is subset of another array

package com.gqt.strings.project; import java.util.Arrays; public class String1 {

public static void main(String[] args) { int[] array1 = {1, 2, 3, 4, 5}; int[] array2 = {3, 1, 4};

boolean isSubset = isSubset(array1, array2);

if (isSubset) {

System.out.println("Array 2 is a subset of Array 1.");

} else {

System.out.println("Array 2 is not a subset of Array 1.");

}

}

public static boolean isSubset(int[] array1, int[] array2) {

// Sort both arrays to simplify the comparison

Arrays.sort(array1);

Arrays.sort(array2);

int i = 0, j = 0;

while (i < array1.length && j < array2.length) { if (array1[i] < array2[j]) { i++;

} else if (array1[i] == array2[j]) {

i++; j++;

} else {

// If an element in array1 is greater than the current element in array2, // array2 is not a subset of array1.

return false;

}

}

// If we reached the end of array2, it means array2 is a subset of array1. return j == array2.length;

}

}

\*/

/\* 17. Find the Smallest and largest element in an array ?

* 18. Find Second Smallest Element in an Array?
* 19. Find Second largest Element in an Array?

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the size of the array"); int n = sc.nextInt(); int [] a = new int [n]; //storing the values

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

System.out.println("Enter the number"+(i+1)); a[i] = sc.nextInt();

}

Arrays.sort(a);

//soritng

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

System.out.print(a[i]+" ");

}

System.out.println();

System.out.println("The second largest element is: "+a[n-2]);

System.out.println("The second smallest element is: "+a[1]);

System.out.println("The smallest element is: "+a[0]);

System.out.println("The largest element is: "+a[n-1]);

}

}

\*/

/\*20. You are working as a software developer at a weather forecasting company. The company is developing a new feature for their app that allows users to convert temperatures from Fahrenheit to Celsius. Your task is to create a method that takes a temperature in Fahrenheit and converts it to Celsius.

Hint: The formula to convert temperature from Fahrenheit to Celsius is

C = (F - 32) \* 5/9

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the Surrounding temperature in Fahrenheit"); double Fahrenheit = sc.nextDouble(); double Celcius = (Fahrenheit-32)\*5/9;

System.out.println("The Surrounding temperature in Degree Celcius is "+Celcius);

/\*21. You are creating a student portal for your school. The portal needs to display the average marks of a student for 8 semesters. Your task is to write a Java function that takes the marks of each semester individually and returns the average.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of subjects"); int n = sc.nextInt(); int sum = 0;

int [] sub = new int [n]; //for storing marks

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

System.out.println("Enter the marks of the subject number"+(i+1)); sub[i]=sc.nextInt();

}

//for taking the marks for average

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

sum = sum+sub[i];

}

System.out.println("The average of the all sem is "+(sum/n));

/\*22. Write a Java program that categorizes people based on their age: Child (0-

12), Teen (13-19), Adult (20-59), Senior (60+)

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter your age"); int age = sc.nextInt(); if (age>=0&&age<=12) {

System.out.println("You are a child");

}

else if (age>=13&&age<=19) {

System.out.println("You are a teen");

}

else if (age>=20&&age<=59) {

System.out.println("You are a adult");

}

else if (age>=60) {

System.out.println("You are a senior");

} else {

System.out.println("Invalid age");

}

/\*23. Write a java program to display all even prime numbers?

\*24. Write a java program to display all odd prime numbers?

package com.gqt.strings.project;

public class String1 {

public static void main(String[] args) { int start = 2; // Starting number for the range int end = 100; // Ending number for the range

System.out.println("Even Prime Numbers:"); for (int i = start; i <= end; i++) { if (isPrime(i) && i % 2 == 0) {

System.out.print(i + " ");

}

}

System.out.println("\nOdd Prime Numbers:"); for (int i = start; i <= end; i++) { if (isPrime(i) && i % 2 != 0) { System.out.print(i + " ");

}

}

}

// Function to check if a number is prime

public static boolean isPrime(int num) { if (num <= 1) { return false;

}

for (int i = 2; i <= Math.sqrt(num); i++) { if (num % i == 0) { return false;

} } return true;

}

}

\*/

/\*25. Write a program to find the cube root of a number. package com.gqt.strings.project;

public class String1 {

public static void main(String[] args) { double s = 27;

System.out.println(Math.cbrt(s));

}

}

\*/

/\*26. Write a java program to take a string as user input and returns the count of uppercase letters and lowercase letters.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the sentence of your wish"); String s1 = sc.nextLine(); int Upper = 0; int Lower = 0;

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

char ch = s1.charAt(i); if (ch>='A'&&ch<='Z') {

Upper++;

}

else if (ch>='a'&&ch<='z') {

Lower++;

}

}

System.out.println("The num of Upper case letters are: "+Upper);

System.out.println("The num of Lower case letters are: "+Lower);

}

}

\*/

/\*27. WAP TO PRINT

* \* \*
* \* \*
* \* \* \*
* \* \* \* \*

package com.gqt.strings.project; import java.util.Scanner; public class String1 {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in); System.out.println("Enter the size: "); int n=sc.nextInt(); for (int i = 1; i <=n; i++) {

//for spces

for (int j = n-1; j >=i; j--) {

System.out.print(" ");

}

//for upper triangle for (int j = 1; j <=i; j++) {

System.out.print("\* ");

}

System.out.println();

}

}

}

\*/

/\*28. Write a Program to Print the Hollow Diamond Star Pattern.

\*

* \*
* \* \* \*
* \*
* \*
* package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the size of the pattern you want"); int n = sc.nextInt(); for (int i = 0; i < n; i++) { for (int j = 0; j < n; j++) { if (i+j==n/2 || j-i==n/2 || i-j==n/2 || i+j==(n-1)+ (n/2)) {

System.out.print("\*");

} else {

System.out.print(" ");

}

}

System.out.println();

}

}

}

\*/

/\*29. Write a program to find the frequency of each digit in a number.

package com.gqt.strings.project; import java.io.\*;

import java.util.\*;

public class String1{

public static void main (String[] args)

{

int n = 898989, count = 0;

int d = 9;

while(n>0){ int rem = n%10; if(rem == d) count++; n /= 10;

}

System.out.println(count);

}

}

\*/

/\*30. Java Program to Multiply two Matrices by Passing Matrix to a Function

package com.gqt.strings.project;

public class String1{ public static void main(String args[]){ //creating two matrices int a[][]={{1,1,1},{2,2,2},{3,3,3}}; int b[][]={{1,1,1},{2,2,2},{3,3,3}};

//creating another matrix to store the multiplication of two matrices int c[][]=new int[3][3]; //3 rows and 3 columns

//multiplying and printing multiplication of 2 matrices for(int i=0;i<3;i++){ for(int j=0;j<3;j++){ c[i][j]=0; for(int k=0;k<3;k++)

{

c[i][j]+=a[i][k]\*b[k][j];

}//end of k loop

System.out.print(c[i][j]+" "); //printing matrix element

}//end of j loop

System.out.println();//new line

}

}

}

\*/

/\*31. write a program to find duplicate characters in a string

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter a string"); String s1 = sc.nextLine(); char [] a = s1.toCharArray();

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

for (int i = 0; i < s1.length(); i++) { for (int j = i + 1; j < s1.length(); j++) {

if (a[i] == a[j]) { System.out.print(a[j] + " ");

break;

}

}

}

}

}

\*/

/\*32. Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.[Example Input: nums = [1,3,5,6], target = 5

Output: 2]

package com.gqt.strings.project; public class String1 {

public static int searchInsert(int[] nums, int target) { int left = 0;

int right = nums.length - 1;

while (left <= right) { int mid = left + (right - left) / 2;

if (nums[mid] == target) { return mid; // Target found

} else if (nums[mid] < target) { left = mid + 1;

} else {

right = mid - 1;

}

}

return left; // Target not found, return the insert position

}

public static void main(String[] args) {

int[] nums = {1, 3, 5, 6}; int target = 5;

int result = searchInsert(nums, target);

System.out.println("Output: " + result);

}

}

\*/

/\*33. program to split a string where an upper case letter occured in a string in java?

package com.gqt.strings.project;

import java.util.regex.\*; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string ");

String s1 = sc.nextLine();

// Define a regular expression pattern to match uppercase letters

Pattern p = Pattern.compile("(?=[A-Z])");

// Use the pattern to split the string

String[] parts = p.split(s1);

// Print the split parts for (String part : parts) {

System.out.println(part);

}

}

}

\*/

/\*34. Given a string, s, and two indices, start and end, print a substring consisting of all characters in the inclusive range from start to end - 1.

Sample Input:Helloworld

3

7

Output:lowo

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input the string

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

String inputString = scanner.nextLine();

// Input the start and end indices

System.out.print("Enter the start index: ");

int start = scanner.nextInt();

System.out.print("Enter the end index: "); int end = scanner.nextInt();

// Ensure the indices are valid

if (start >= 0 && end >= start && end <= inputString.length()) { String result = inputString.substring(start, end);

System.out.println("Substring: " + result);

} else {

System.out.println("Invalid indices.");

}

}

}\*/

/\*35. Write a java program to take a string as user input and returns the count of

uppercase letters

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the sentence of your wish"); String s1 = sc.nextLine(); int Upper = 0; int Lower = 0;

for (int i = 0; i < s1.length(); i++) { char ch = s1.charAt(i); if (ch>='A'&&ch<='Z') {

Upper++;

}

else if (ch>='a'&&ch<='z') {

Lower++;

}

}

System.out.println("The num of Upper case letters are: "+Upper); System.out.println("The num of Lower case letters are: "+Lower);

}

}

\*/

/\*36. Write a java program to find out and print the longest word present in the

sentence.Write the code in the findLongestWord() method which accepts a string and returns the longest word

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String args[])

{

Scanner scanner=new Scanner(System.in);

System.out.println("Enter the string");

String s1=scanner.nextLine();

String small="",large="",word=""; String words[]=new String[100]; int length=0; s1=s1+"";

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

{

if(s1.charAt(i)==' ')

{

words[length]=word; length++; word="";

}

else

{

word=word+s1.charAt(i);

}

}

small=large=words[0];

for(int k=0;k<length;k++)

{

if(small.length()>words[k].length())

{

small=words[k];

}

if(large.length()<words[k].length())

{

large=words[k];

}

}

System.out.println("Smallest word "+small);

System.out.println("Largest word"+large);

}

}

\*/

/\*37. Write a program that takes your full name as input and displays the abbreviations of the first and middle names except the last name which is displayed as it is. For example, if your name is Robert Brett Roser, then the output should be R.B.Roser.

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String args[])

{

Scanner scanner = new Scanner(System.in);

System.out.print("Enter your full name: "); String fullName = scanner.nextLine(); scanner.close();

String[] nameParts = fullName.split(" "); int numNameParts = nameParts.length;

if (numNameParts >= 2) {

StringBuilder abbreviation = new StringBuilder();

// Abbreviate the first name

abbreviation.append(nameParts[0].charAt(0)).append(".");

// Abbreviate the middle names (if any) for (int i = 1; i < numNameParts - 1; i++) { abbreviation.append(nameParts[i].charAt(0)).append(".");

}

// Append the last name abbreviation.append(nameParts[numNameParts - 1]);

System.out.println("Abbreviated name: " + abbreviation.toString());

} else {

System.out.println("Invalid input. Please enter your full name with at least two parts.");

}

}

}

\*/

/\*38. Write a program to delete all consonents from the string "Hello, have a good day".

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

String s; int j=0;

System.out.println("Enter a string"); Scanner so=new Scanner(System.in); s= so.nextLine(); char ch[]=new char[20];

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

if(s.charAt(i)=='a'|| s.charAt(i)=='A'||s.charAt(i)=='e'|| s.charAt(i)=='E'||

s.charAt(i)=='i'|| s.charAt(i)=='I'||s.charAt(i)=='o'|| s.charAt(i)=='O'||

s.charAt(i)=='U'|| s.charAt(i)=='u')

{

continue;

}

else{

ch[j++]=s.charAt(i);

}

}

for(int i=0;i<j;i++)

{

System.out.print(ch[i]);

}

System.out.println();

}

}

\*/

/\*39. write a java program to capitalize each word in a string . input: "hello this is java program" then the output should be : : "Hello This Is Java Program".

package com.gqt.strings.project; public class String1 {

public static void main(String[] args) {

// create a string

String message = "hello this is java program";

// stores each characters to a char array char[] charArray = message.toCharArray(); boolean foundSpace = true;

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

// if the array element is a letter if(Character.isLetter(charArray[i])) {

// check space is present before the letter if(foundSpace) {

// change the letter into uppercase

charArray[i] = Character.toUpperCase(charArray[i]); foundSpace = false; }

}

else {

// if the new character is not character foundSpace = true;

}

}

// convert the char array to the string message = String.valueOf(charArray);

System.out.println("Message: " + message);

}

}

\*/

/\*40. WAP to reverse each word in a string. input : "i love my country" output: "i evol ym yrtnuoc".

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("enter the string");

String s1 =sc.nextLine(); String s2 = ""; int sp\_count=0; //counting words

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

sp\_count++;

}

}

int word\_count=sp\_count+1;

//creating the array

String str[]=new String[word\_count];

int count=0;

//traversing string in reverse order for (int i = s1.length()-1; i>=0; i--) {

if (s1.charAt(i)==' ') {

str[count]=s2;

s2=""; count++;

}

else {

s2=s2+s1.charAt(i);

}

}

str[count]=s2;

//printing the array

for(int i =str.length-1; i>=0; i--) {

System.out.print(str[i]+" ");

}

}

}

\*/

/\*41. Take 10 integer inputs from user and store them in an array. prompt user to give a number.Check whether that number is present in array or not and if present print index of the number.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); int [] a = new int[10]; //storing all the 10 integer values for (int i = 0; i < a.length; i++) {

System.out.println("Enter the number"+(i+1)); a[i]=sc.nextInt();

}

//For searching the number in the array System.out.println("Enter the number to search"); int search = sc.nextInt(); for (int i = 0; i < a.length; i++) {

if (a[i]==search) {

System.out.println("The number present in the index of "+i);

}

}

}

}

\*/

/\*42. Take 20 integer inputs from user and print the following:

number of positive numbers number of negative numbers number of odd numbers number of even numbers number of 0s.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); int [] a = new int[20]; int positive = 0; int negative = 0; int odd = 0; int even = 0; int zero = 0;

//storing all the 10 integer values for (int i = 0; i < a.length; i++) {

System.out.println("Enter the number"+(i+1)); a[i]=sc.nextInt();

}

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

if (a[i]>=0) {

positive++;

}

if (a[i]<0) {

negative++;

}

if (a[i]%2==0&&a[i]%2>=0) {

even++;

}

if (a[i]%2!=0&&a[i]%2>=0) { odd++;

}

if(a[i]==0) {

zero++;

}

}

System.out.println("The number of positive numbers are: "+(positive));

System.out.println("The number of negative numbers are: "+(negative));

System.out.println("The number of even numbers are: "+(even));

System.out.println("The number of odd numbers are: "+(odd));

System.out.println("The number of zeros are: "+(zero));

}

}

\*/

/\*43. write a program to check whether the integer array contains only odd numbers?

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the size of the array"); int n = sc.nextInt(); int [] a = new int[n]; //storing the values

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

System.out.println("Enter the num"+(i+1)); a[i]=sc.nextInt();

}

//to check weather the array contains only odd numbers or not for (int i = 0; i < a.length; i++) { if (a[i]%2!=0) {

System.out.println("This array is containing odd numbers");

continue;

}

else if (a[i]%2==0) {

System.out.println("This array is also containing the even numbers");

break;

}

else {

System.out.println("This array is containing odd numbers");

continue;

}

}

}

}

\*/

/\*44. Take 10 integer inputs from user and store them in an array. and copy all the elements into an another array in reverse order.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("enter the size of the array"); int n = sc.nextInt();

int [] a = new int[n];

int [] b = new int[n];

//To store the elements for (int i = 0; i < a.length; i++) { System.out.println("enter the element num"+(i+1)); a[i] = sc.nextInt();

}

//To reverse the array int temp = 0;

for (int i = a.length-1; i >=0 ; i--) {

b[temp]=a[i]; temp++;

}

//To print the reversed array for (int i = 0; i <a.length ; i++) {

System.out.print(b[i]+" ");

}

}

}

\*/

/\*45. Find largest and smallest elements of an array.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the size of the array"); int n = sc.nextInt(); int [] a = new int [n];

//storing the values for (int i = 0; i < a.length; i++) {

System.out.println("Enter the number"+(i+1)); a[i] = sc.nextInt();

}

Arrays.sort(a);

//soritng

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

System.out.print(a[i]+" ");

}

System.out.println();

System.out.println("The smallest element is: "+a[0]);

System.out.println("The largest element is: "+a[n-1]);

}

}

\*/

/\*46. program to remove a particular element from an array.

package com.gqt.strings.project;

import java.lang.reflect.Array; import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the size of the array"); int n = sc.nextInt();

int [] a = new int[n]; int [] new\_arr = new int[n-1];

//for storing the elements for (int i = 0; i < a.length; i++) {

System.out.println("Enter the element num"+(i+1)); a[i]=sc.nextInt();

}

//to print the elements for (int i = 0; i < a.length; i++) {

System.out.print(a[i]);

}

System.out.println();

System.out.println("Enter the number you want to remove"); int remove =sc.nextInt();

for(int i=0, k=0;i<a.length;i++){

if(i!=remove){

new\_arr[k]=a[i];

k++;

}

}

System.out.println("Before deletion :" + Arrays.toString(a));

System.out.println("After deletion :" + Arrays.toString(new\_arr));

}

}

\*/

/\*47. program to find the second largest integer in an array.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the size of the array"); int n = sc.nextInt(); int [] a = new int [n]; //storing the values

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

System.out.println("Enter the number"+(i+1)); a[i] = sc.nextInt();

}

Arrays.sort(a);

//soritng

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

System.out.print(a[i]+" ");

}

System.out.println();

System.out.println("The largest element is: "+a[n-2]);

}

}

\*/

/\*48. program sort an array of 0 and 1? input:[0,1,1,1,0,1,0] then the output should be

[0,0,0,1,1,1,1] without using

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); int [] a = {0,1,1,1,0,1,0};

Arrays.sort(a);

System.out.println(Arrays.toString(a));

}

}

\*/

/\*49. Consider an integer array, the number of elements in which is determined by the user. The elements are also taken as input from the user. Write a program to find those pair of elements that has the maximum and minimum difference among all element pairs.

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void findMaxMinDifferencePairs(int[] nums) { if (nums.length < 2) {

System.out.println("The array must have at least two elements.");

return;

}

int maxDifference = Integer.MIN\_VALUE; int minDifference = Integer.MAX\_VALUE;

int maxElement1 = 0, maxElement2 = 0, minElement1 = 0, minElement2 = 0;

for (int i = 0; i < nums.length - 1; i++) { for (int j = i + 1; j < nums.length; j++) {

int difference = Math.abs(nums[i] - nums[j]);

if (difference > maxDifference) { maxDifference = difference; maxElement1 = nums[i]; maxElement2 = nums[j];

}

if (difference < minDifference) { minDifference = difference; minElement1 = nums[i]; minElement2 = nums[j];

}

}

}

System.out.println("Pair with Maximum Difference: " + maxElement1 + " and " + maxElement2);

System.out.println("Maximum Difference: " + maxDifference);

System.out.println("Pair with Minimum Difference: " + minElement1 + " and " + minElement2);

System.out.println("Minimum Difference: " + minDifference);

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements in the array: "); int n = scanner.nextInt(); int[] nums = new int[n];

System.out.println("Enter the elements of the array:");

for (int i = 0; i < n; i++) {

nums[i] = scanner.nextInt();

}

findMaxMinDifferencePairs(nums);

}

}\*/

/\*50. program to find the missing integer in an array of range 1 to 10.

package com.gqt.strings.project;

import java.util.Arrays;

public class String1 {

public static void main(String[] args) { int [] a = {1,2,3,5,6,7,8,9,10};

//to get the sum from 1 to 10

int sum =0;

for (int i = 1; i <= 10; i++) { sum += i;

}

System.out.println("The sum of 1 to 10 range is: "+(sum));

//to get the actual sum of the defined array int act\_sum = 0;

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

act\_sum += a[i];

}

System.out.println("The actual sum is: "+(act\_sum));

//To print the missing element in range of 1 to 10 if (a.length==10) {

System.out.println("There is no missing element in the range of 1 to 10"); } else {

System.out.println("The missing number in the 1 to 10 range is: "+(sum - act\_sum));

}

}

}

\*/

/\*51. How to Find Common Element Between Two Arrays In Java?

package com.gqt.strings.project;

import java.util.ArrayList; import java.util.List;

public class String1 {

public static int[] findCommonElements(int[] array1, int[] array2) {

List<Integer> commonElementsList = new ArrayList<>();

for (int i = 0; i < array1.length; i++) { for (int j = 0; j < array2.length; j++) { if (array1[i] == array2[j]) { commonElementsList.add(array1[i]); break; // Break to avoid duplicates

}

}

}

// Convert the list to an array

int[] commonElementsArray = new int[commonElementsList.size()]; for (int i = 0; i < commonElementsList.size(); i++) { commonElementsArray[i] = commonElementsList.get(i);

}

return commonElementsArray;

}

public static void main(String[] args) { int[] array1 = {1, 2, 3, 4, 5}; int[] array2 = {3, 4, 5, 6, 7};

int[] commonElements = findCommonElements(array1, array2);

System.out.println("Common Elements: "); for (int element : commonElements) {

System.out.print(element + " ");

}

}

}

\*/

/\*52. Find the highest palindrome number in a given array which contains both palindrome and non-palindrome numbers?

package com.gqt.strings.project;

public class String1 {

public static boolean isPalindrome(int num) {

int originalNum = num; int reversedNum = 0;

while (num > 0) { int digit = num % 10;

reversedNum = reversedNum \* 10 + digit; num /= 10;

}

return originalNum == reversedNum;

}

public static int findHighestPalindrome(int[] nums) {

int highestPalindrome = -1;

for (int num : nums) {

if (isPalindrome(num) && num > highestPalindrome) {

highestPalindrome = num;

}

}

return highestPalindrome;

}

public static void main(String[] args) { int[] array = {121, 345, 1331, 555, 101}; int highestPalindrome = findHighestPalindrome(array);

if (highestPalindrome != -1) {

System.out.println("Highest Palindrome: " + highestPalindrome);

} else {

System.out.println("No palindrome found in the array.");

}

}

}

\*/

/\*53. Move all negative numbers to beginning and positive to end.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Iterator; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in); System.out.println("Enter the size of an array"); int n = scanner.nextInt(); int [] a = new int[n]; //to store the elements for (int i = 0; i < a.length; i++) {

System.out.println("Enter the element"+(i+1)); a[i]=scanner.nextInt();

}

Arrays.sort(a);

System.out.println("The sorted array is "+Arrays.toString(a));

}

}

\*/

/\*54. Given an integer array nums and an integer val, remove all occurrences of val in nums in-place. The order of the elements may be changed. Then return the number of elements in nums which are not equal to val.

package com.gqt.strings.project; public class String1 { public static int removeElement(int[] nums, int val) { int n = nums.length;

int left = 0; // Pointer for elements not equal to val

for (int right = 0; right < n; right++) { if (nums[right] != val) { nums[left] = nums[right];

left++; // Move the element not equal to val to the left

}

}

return left;

}

public static void main(String[] args) { int[] nums = {3, 2, 2, 3}; int val = 3;

int result = removeElement(nums, val);

System.out.println("Output: " + result);

}

}

\*/

/\*55. Consider the number of elements in nums which are not equal to val be k, to get accepted, you need to do the following things:

Change the array nums such that the first k elements of nums contain the elements which are not equal to val. The remaining elements of nums are not important as well as the size of nums.

Return k.

Example

Input: nums = [3,2,2,3], val = 3

Output: 2, nums = [2,2,,]

Explanation: Your function should return k = 2, with the first two elements of nums being 2

package com.gqt.strings.project; public class String1 {

public static int removeElement(int[] nums, int val) { int k = 0; // Initialize the count of elements not equal to val for (int i = 0; i < nums.length; i++) { if (nums[i] != val) { nums[k] = nums[i]; k++; // Increment k for each element not equal to val

} } return k;

}

public static void main(String[] args) { int[] nums = {3, 2, 2, 3}; int val = 3; int result = removeElement(nums, val);

System.out.println("Output: " + result); System.out.print("nums = ["); for (int i = 0; i < result; i++) { System.out.print(nums[i]); if (i < result - 1) {

System.out.print(", ");

}

}

System.out.println("]");

}

}

\*/

/\*56. find the sum of the digits of a number

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

int num, digit, sum=0;

Scanner s = new Scanner(System.in);

System.out.print("Enter a Number: "); num = s.nextInt();

while(num!=0)

{

digit = num%10; sum = sum + digit; num = num/10;

}

System.out.println("\nSum of Digits = " +sum);

}

}

\*/

/\*57. print below pattern

1

1. 1
2. 0 1

0 1 0 1

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("enter the size"); int n = sc.nextInt(); for (int i = 1; i <=n; i++) { for (int j = 1; j<=i ; j++) {

if ((i+j)%2==0) {

System.out.print(1+" ");

}

else {

System.out.print(0+" ");

}

}System.out.println();

}

}

}

\*/

/\*58. Write a program to remove characters from the first string which are present in second string

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the string");

String s1 = scanner.nextLine();

System.out.println("Enter the character you want to replace"); String s2 = scanner.next(); s1=s1.replace(s2, "");

System.out.println(s1);

}

}

\*/

/\*59. write a program to find the given string is sorted with a specified string or not .Example(String1="JavaScript" ,specified string is"java" the output is "true")

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the string");

String s1 = scanner.nextLine();

System.out.println("Enter the string you want to search"); String s2 = scanner.next(); if (s1.contains(s2)) {

System.out.println("The string contains the word you entered");

} else {

System.out.println("The string doesn't contains the word u entered");

}

}

}

\*/

/\*60. print below pattern

A B C

A B

A

A

A B

A B C

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the size"); int n = sc.nextInt();

for (int i = 0; i < n; i++) { for (int j = 0; j < n; j++) { if (j==0) {

System.out.print("A ");

}

else if (j==1&&i<=1 || j==1&&i>=4) {

System.out.print("B ");

}

else if (j==2&&i==0 || j==2&&i==n-1) {

System.out.print("C ");

}

}

System.out.println();

}

}

}

\*/

/\*61. Write a program in Java to calculate the number of times a digit ‘D’ appears in

\* a number N. You have to take N and D as inputs from the user.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the number"); long n = sc.nextLong();

System.out.println("Enter the digit to find"); int d = sc.nextInt(); int count = 0;

while(n>0){ long rem = n%10;

if(rem == d)

count++;

n /= 10;

}

System.out.println("The count of the digit "+d+" is "+ count);

}

}

\*/

/\*62. Write a program in Java to Toggle the case of every character of a string.

\* For instance, if the input string is “ApPLe”, the output should be “aPplE”.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input the string

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

String inputString = scanner.nextLine();

// Toggle the case of each character and build the result string StringBuilder result = new StringBuilder(); for (int i = 0; i < inputString.length(); i++) { char currentChar = inputString.charAt(i); if (Character.isUpperCase(currentChar)) { result.append(Character.toLowerCase(currentChar));

}

else if (Character.isLowerCase(currentChar)) { result.append(Character.toUpperCase(currentChar));

}

else {

// If the character is not a letter, keep it unchanged result.append(currentChar);

}

}

// Output the result

System.out.println("Toggled case string: " + result.toString());

}

}

\*/

/\*63. write a java program to Merge the two Arrays.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) { int [] a = {1,2,3,4}; int [] b = {5,6,7,8,9,10}; int [] res = new int [a.length+b.length];

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

res[i]=a[i];

}

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

res[i+a.length]=b[i];

}

System.out.println(Arrays.toString(res));

}

}

\*/

/\*64. print 1 to n numbers using foreach loop

* In Java, you can't directly use a foreach loop to print numbers from 1 to n, \* as foreach loops are primarily used for iterating through arrays or collections.
* However, you can use a regular for loop to achieve this task.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the size"); int n = sc.nextInt(); for (int i = 0; i <= n; i++) {

System.out.print(i+" ");

}

}

}

\*/

/\*65. Write a Java Program to reverse a string with using String inbuilt function reverse(). package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the Name");

String s = sc.nextLine();

StringBuilder s1 = new StringBuilder(s); s1.reverse();

System.out.println(s1);

}

}

\*/

/\*66. find the given element that is present in an array or not, by using Binary Search.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the element to search");

int n = sc.nextInt();

int [] a = {1,2,3,4,5,6}; int res = Arrays.binarySearch(a, n);

if (res<0) {

System.out.println("Element not found");

} else {

System.out.println("The element found at the index: "+res);

}

}

}

\*/

/\*67. convert the given integer into a binary number format.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the number"); int n = sc.nextInt();

String binary = Integer.toBinaryString(n);

System.out.println("Binary representation of " + n + " is: " + binary);

}

}

\*/

/\*68. convert the given integer into a binary number format.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

int[] array = { -40, -5, 1, 3, 6, 7, 8, 20 };

int targetSum = 15;

findPairsWithSum(array, targetSum);

}

public static void findPairsWithSum(int[] array, int targetSum) { for (int i = 0; i < array.length - 1; i++) { for (int j = i + 1; j < array.length; j++) { if (array[i] + array[j] == targetSum) {

System.out.println("Pair of elements whose sum is equal to " + targetSum + ": " + array[i] + ", " + array[j]);

}

}

}

}

}

\*/

/\*69. convert the given integer into a binary number format.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

int[][] matrix = {

{5, 2, 9, 1},

{8, 3, 7, 6},

{4, 5, 1, 2}

};

findSmallestInEachRow(matrix);

}

public static void findSmallestInEachRow(int[][] matrix) {

|  |  |  |
| --- | --- | --- |
|  | for (int row = 0; row < matrix.length; row++) { | |
| row |  | int smallest = matrix[row][0]; // Initialize the smallest with the first element of the |
|  |  | for (int col = 1; col < matrix[row].length; col++) { |
|  |  | if (matrix[row][col] < smallest) { |
| found |  | smallest = matrix[row][col]; // Update smallest if a smaller element is |
|  |  | } |
|  |  | } |
|  |  | System.out.println("Smallest element in row " + (row + 1) + ": " + smallest); |
|  | } |  |

}

}

\*/

/\*70. perfor XOR operation on two integers.

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); System.out.println("Enter the num1 value"); int num1 = sc.nextInt(); // First integer

System.out.println("Enter the num2 value");

int num2 = sc.nextInt(); // Second integer

int result = num1 ^ num2; // Perform XOR operation

System.out.println("Result of " + num1 + " XOR " + num2 + " is: " + result);

}

}

\*/

/\*71. From the given input separate numbers and characters

Example- input: j34784ha

output:

jha

3478

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

String s1 = "j34784ha";

String numbers = "";

String characters = "";

for (int i = 0; i < s1.length(); i++) { char ch = s1.charAt(i); if (Character.isDigit(ch)) { numbers += ch;

} else if (Character.isLetter(ch)) { characters += ch;

}

}

System.out.println("Numbers: " + numbers);

System.out.println("Characters: " + characters);

}

}

\*/

/\*72. write a program to find the sum of odd integers in an array and prduct of even integers in array.finally find sum of the both of the results. example:

input:[3,4,5,2,1,7]

output:

odd sum=3+5+1+7=16 even product=4+2=6 final sum=22

package com.gqt.strings.project;

import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) { int [] a = {3,4,5,2,1,7}; int even = 0; int odd = 0;

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

if (a[i]%2==0) { even += a[i];

}

else { odd += a[i];

}

}

System.out.println("The even sum in the array is: "+even);

System.out.println("The odd sum in the array is: "+odd);

System.out.println("The addition of both odd and even sum is: "+(odd+even));

}

}

\*/

/\*73.find the given input number is perfect number or not.The perfect number is the

\* number where the sum of its divisors is equal to the number itself.

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in); long n, sum=0;

System.out.print("Enter the number: "); n=sc.nextLong(); int i=1; while(i <= n/2)

{

if(n % i == 0)

{

//calculates the sum of factors

sum = sum + i;

}

i++;

}

//compares sum with the number if(sum==n)

{

//prints if sum and n are equal

System.out.println(n+" is a perfect number.");

}

else

//prints if sum and n are not equal

System.out.println(n+" is not a perfect number.");

}

}

\*/

/\*74.Replace all zero's with one's and one's with zero's.

package com.gqt.strings.project; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

String input = "010101001011";

String result = "";

for (int i = 0; i < input.length(); i++) { if (input.charAt(i) == '0') { result += '1'; // Replace '0' with '1' } else if (input.charAt(i) == '1') { result += '0'; // Replace '1' with '0'

}

}

System.out.println("Original: " + input);

System.out.println("Replaced: " + result);

}

}

\*/

/\*75.Sort the first half of the array elements in ascending order second half of the array elements in descending order.

package com.gqt.strings.project; import java.util.Arrays; import java.util.Scanner;

public class String1 { static void printOrder(int[] arr, int n)

{

// sorting the array

Arrays.sort(arr);

// printing first half in ascending order for (int i = 0; i < n / 2; i++)

System.out.print(arr[i]+" ");

// printing second half in descending order for (int j = n - 1; j >= n / 2; j--)

System.out.print(arr[j]+" ");

}

// Main

public static void main(String[] args)

{

int[] arr = { 5, 4, 6, 2, 1, 3, 8, 9, 7,10 }; int n = arr.length; printOrder(arr, n);

}

}

\*/

/\*76.find the most repeated word in a sentence.

package com.gqt.strings.project; import java.util.Arrays; import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string"); String s1 = sc.nextLine(); char[] str = s1.toCharArray(); int len[] = new int [s1.length()]; for (int i = 0; i < s1.length(); i++) {

len[i]=1;

for (int j = i+1; j < s1.length(); j++) {

if (str[i]==str[j]) {

len[i]++; str[j]=0;

}

}

}

for (int i = 0; i < len.length; i++) { if (str[i]==' '||str[i]==0) {

} else {

System.out.println(str[i]+"---->"+len[i]);

}

}

}

}

\*/

/\*77.check number belongs to Fibonacci series or not.

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter a number"); // read the number to be checked int numberToCheck = sc.nextInt();

int firstNumber = 0, secondNumber = 1, fibonacciNumber = 0;

// loop till the current fibonacci number is less than the number to

// check

while (fibonacciNumber < numberToCheck) { // calculate the next fibonacci number fibonacciNumber = firstNumber + secondNumber;

// move the fibonacci series ahead firstNumber = secondNumber; secondNumber = fibonacciNumber;

}

// compare the current fibonacci number with number to check if (numberToCheck == fibonacciNumber) {

System.out.println("Number belongs to Fibonacci series");

} else {

System.out.println("Number does not belong to Fibonacci series");

}

}

}

\*/

/\*78.Remove an element at specific index from an array.

package com.gqt.strings.project;

import java.util.\*;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); int[] arr = {1,2,3,4,5};

System.out.println("Before deletion :" + Arrays.toString(arr)); System.out.println("Enter the index to delete the element"); int remove = sc.nextInt(); int[] arr\_new = new int[arr.length-1]; for(int i=0, k=0;i<arr.length;i++){

if(i!=remove){ arr\_new[k]=arr[i];

k++;

}

}

System.out.println("After deletion :" + Arrays.toString(arr\_new));

}

}

\*/

/\*79.check whether the given string is panagram or not.

package com.gqt.strings.project;

import java.util.\*;

public class String1 {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

System.out.println("Enter Your String:");

String str=sc.nextLine();

str=str.replaceAll("","").toLowerCase();

// empty string

String s="";

// checking characters (a-z or A-Z) for(char i='a';i<='z';i++){

//indexOf(char i)--> This method returns '-1' substring not found, if the position of substrings 'i' in 'str'

if(str.indexOf(i)!=-1){

s=s+i;// empty string+character

}

}

if(s.length()==26){

System.out.println("Pangram");

}

else{

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

}

}

}

\*/

/\*80. find sum of prime numbers in a given range.

package com.gqt.strings.project;

import java.util.\*;

public class String1 {

public static void main(String args[])

{

int number = 1, count, sum = 0;

while(number <= 100)

{

count = 0; int i = 2; while(i <= number/2 ) { if(number % i == 0)

{

count++; break;

}

i++;

}

if(count == 0 && number != 1 )

{

sum = sum + number;

}

number++;

}

System.out.println("The Sum of Prime Numbers from 1 to 100 is: " + sum);

}

}

\*/

/\*81. Write a Java code to solve the Dining Philosophers problem?

package com.gqt.strings.project;

import java.util.concurrent.Semaphore; import java.util.concurrent.ThreadLocalRandom; public class String1

{

//defining the number of philosophers static int philosopher = 5;

//initializing an array of philosophers with the number of philosophers static Philosopher philosophers[] = new Philosopher[philosopher]; //initializing an array of chosticks with the number of philosophers static Chopstick chopsticks[] = new Chopstick[philosopher]; static class Chopstick

{

//creating a constructor of the Semaphore class that accepts the number permits public Semaphore mutex = new Semaphore(1);

//the method grabs the chopstick void grab()

{ try

{

//acquires a permit from the semaphore mutex.acquire();

}

catch (Exception e)

{

e.printStackTrace(System.out);

}

}

//release the chopstick

void release()

{

//releases an acquire a permit and increases the number of available permits by one mutex.release();

}

//checks if the chopstick is free or not boolean isFree()

{

//the method returns the current number of permits available in the semaphore

//returns true if available permits is greater than 0, else returns false return mutex.availablePermits() > 0;

}

} //end of Chopstick class static class Philosopher extends Thread

{

public int number;

//represents left chopstick public Chopstick leftchopstick; //represents right chopstick public Chopstick rightchopstick;

//creating a constructor of the Philosopher class

Philosopher(int num, Chopstick left, Chopstick right)

{

number = num;

leftchopstick = left; rightchopstick = right;

}

public void run()

{

while (true)

{

//philosopher grabs the chopsticks if both are free leftchopstick.grab();

System.out.println("Philosopher " + (number+1) + " grabs left chopstick."); rightchopstick.grab();

System.out.println("Philosopher " + (number+1) + " grabs right chopstick.");

//hunger philosopher starts eating

eat();

//releases left and right chopsticks when philosopher is not hunger

leftchopstick.release();

System.out.println("Philosopher " + (number+1) + " releases left chopstick."); rightchopstick.release();

System.out.println("Philosopher " + (number+1) + " releases right chopstick.");

} //end of while loop

} //end of run() method

//the method invokes after grabbing both the chopsticks (left and right) void eat()

{ try

{

//determines the pseudorandom number between 0 to 1000 that represents the sleep time in milli seconds

int sleepTime = ThreadLocalRandom.current().nextInt(0, 1000); System.out.println("Philosopher " + (number+1) + " eats for " + sleepTime

+"ms"); //sleeps the thread for a specified time

Thread.sleep(sleepTime);

}

catch (Exception e)

{

e.printStackTrace(System.out);

}

}// end of eat() method

}

public static void main(String args[])

{

//loop iterates over chopsticks for (int i = 0; i < philosopher; i++)

{

chopsticks[i] = new Chopstick();

} //end of for loop

//loop iterates over philosopher for (int i = 0; i < philosopher; i++)

{

philosophers[i] = new Philosopher(i, chopsticks[i], chopsticks[(i + 1) % philosopher]);

//begins the execution of the thread philosophers[i].start(); } //end of for loop while (true)

{ try

{

//thread sleep for 1 sec or 1000ms

Thread.sleep(1000);

//check for deadlock condition boolean deadlock = true;

//for each loop iterates over chopsticks for (Chopstick cs : chopsticks)

{

//checks if chopstick is free or not if (cs.isFree())

{

deadlock = false; break;

} //end of if

} //end of for loop

//deadlock occurs if sleep time is 1000ms it means each philosopher is eating if (deadlock)

{

Thread.sleep(1000);

System.out.println("Everyone Eats");

break;

} //end of if

}

catch (Exception e)

{

e.printStackTrace(System.out);

}

}

System.out.println("Exit The Program!");

System.exit(0);

}

}

\*/

/\*82.Solve the 0/1 Knapsack Problem using dynamic programming in Java.

package com.gqt.strings.project;

public class String1 {

public static int knapsack(int[] weights, int[] values, int capacity) {

int n = weights.length;

int[][] dp = new int[n + 1][capacity + 1];

for (int i = 0; i <= n; i++) { for (int w = 0; w <= capacity; w++) { if (i == 0 || w == 0) { dp[i][w] = 0;

} else if (weights[i - 1] <= w) {

dp[i][w] = Math.max(values[i - 1] + dp[i - 1][w - weights[i - 1]], dp[i -

1][w]);

} else {

dp[i][w] = dp[i - 1][w];

}

}

}

return dp[n][capacity];

}

public static void main(String[] args) {

int[] weights = {2, 3, 4, 5}; int[] values = {3, 4, 5, 6}; int capacity = 5;

int maxValue = knapsack(weights, values, capacity);

System.out.println("Maximum Value: " + maxValue);

}

}

\*/

/\*83.Write a program to solve the classic "N-Queens" problem, where you need to place N queens on an N×N chessboard such that no two queens threaten each other?(N-Queens Problem)

package com.gqt.strings.project;

public class String1 { private static int N;

public static void solveNQueens(int n) {

N = n;

int[][] board = new int[N][N]; if (solve(board, 0)) { printSolution(board);

} else {

System.out.println("No solution exists.");

}

}

private static boolean solve(int[][] board, int col) {

if (col >= N) {

return true;

}

for (int i = 0; i < N; i++) {

if (isSafe(board, i, col)) {

board[i][col] = 1;

if (solve(board, col + 1)) {

return true;

}

board[i][col] = 0; // Backtrack

}

}

return false;

}

private static boolean isSafe(int[][] board, int row, int col) {

int i, j;

// Check this row on the left side for (i = 0; i < col; i++) { if (board[row][i] == 1) { return false;

}

}

// Check upper diagonal on the left side for (i = row, j = col; i >= 0 && j >= 0; i--, j--) {

if (board[i][j] == 1) {

return false;

}

}

// Check lower diagonal on the left side for (i = row, j = col; i < N && j >= 0; i++, j--) {

if (board[i][j] == 1) {

return false;

}

}

return true;

}

private static void printSolution(int[][] board) { for (int i = 0; i < N; i++) { for (int j = 0; j < N; j++) {

System.out.print(board[i][j] + " ");

}

System.out.println();

}

}

public static void main(String[] args) {

int n = 8; // Change this value for different board sizes solveNQueens(n);

}

}

\*/

/\*84.Define a base class "Shape" with a method "calculateArea." Create two subclasses, "Circle" and "Triangle," that inherit from "Shape" and provide their own implementations of "calculateArea." Demonstrate polymorphism by calculating the area of various shapes using an array of "Shape" objects.

package com.gqt.strings.project;

class Shape { public double calculateArea() { return 0.0; // Default area for a generic shape

}

}

class Circle extends Shape { private double radius;

public Circle(double radius) { this.radius = radius;

}

@Override

public double calculateArea() { return Math.PI \* radius \* radius;

}

}

class Triangle extends Shape { private double base; private double height;

public Triangle(double base, double height) { this.base = base; this.height = height;

}

@Override

public double calculateArea() { return 0.5 \* base \* height;

}

}

public class String1 {

public static void main(String[] args) { Shape[] shapes = new Shape[3]; shapes[0] = new Circle(5.0); shapes[1] = new Triangle(4.0, 6.0); shapes[2] = new Circle(3.0);

for (Shape shape : shapes) {

System.out.println("Area: " + shape.calculateArea());

}

}

}

\*/

/\*85.Implement a class hierarchy for a zoo simulation. Create a base class "Animal" with a method "makeSound." Then, create subclasses for different animals like "Lion," "Elephant," and "Monkey" that override the "makeSound" method to produce their specific sounds. Use polymorphism to make animals in the zoo make their sounds.

package com.gqt.strings.project; class Animal {

public void makeSound() {

System.out.println("The animal makes a sound");

}

}

class Lion extends Animal {

@Override

public void makeSound() {

System.out.println("The lion roars");

}

}

class Elephant extends Animal {

@Override

public void makeSound() {

System.out.println("The elephant trumpets");

}

}

class Monkey extends Animal {

@Override

public void makeSound() {

System.out.println("The monkey chatters");

}

}

public class String1 {

public static void main(String[] args) {

// Create instances of different animals

Animal lion = new Lion();

Animal elephant = new Elephant();

Animal monkey = new Monkey();

// Use polymorphism to make animals in the zoo make their sounds makeAnimalSound(lion); makeAnimalSound(elephant); makeAnimalSound(monkey);

}

public static void makeAnimalSound(Animal animal) {

animal.makeSound();

}

}

\*/

/\*86.Develop a banking system with a base class "Account" and subclasses "SavingsAccount" and "CheckingAccount." Each account type should have a method "calculateInterest" that calculates interest differently. Demonstrate polymorphism by calling the "calculateInterest" method on both account types.

package com.gqt.strings.project;

class Account { private double balance;

public Account(double balance) { this.balance = balance;

}

public double getBalance() {

return balance;

}

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) { if (balance >= amount) {

balance -= amount;

} else {

System.out.println("Insufficient funds");

}

}

public void calculateInterest() {

System.out.println("Interest calculation for a generic account");

}

}

class SavingsAccount extends Account { private double interestRate;

public SavingsAccount(double balance, double interestRate) {

super(balance);

this.interestRate = interestRate;

}

@Override

public void calculateInterest() {

double interest = getBalance() \* (interestRate / 100);

System.out.println("Interest calculated for Savings Account: $" + interest);

}

}

class CheckingAccount extends Account { private double monthlyFee;

public CheckingAccount(double balance, double monthlyFee) {

super(balance); this.monthlyFee = monthlyFee;

}

@Override

public void calculateInterest() {

System.out.println("No interest is calculated for Checking Account");

}

}

public class String1 {

public static void main(String[] args) {

SavingsAccount savingsAccount = new SavingsAccount(1000.0, 5.0);

CheckingAccount checkingAccount = new CheckingAccount(1500.0, 10.0); // Demonstrate polymorphism by calling calculateInterest on both account types Account[] accounts = {savingsAccount, checkingAccount}; for (Account account : accounts) {

System.out.println("Account balance: $" + account.getBalance()); account.calculateInterest();

System.out.println();

}

}

}

\*/

/\*87.Write a java code to verify performance of StringBuffer and StringBuilder classes.

package com.gqt.strings.project;

public class String1 {

public static void main(String[] args) {

int iterations = 1000000; // Number of iterations for the test

// Using StringBuffer

long startTime = System.currentTimeMillis(); StringBuffer stringBuffer = new StringBuffer(); for (int i = 0; i < iterations; i++) {

stringBuffer.append("Hello, ");

}

long endTime = System.currentTimeMillis(); long stringBufferTime = endTime - startTime;

System.out.println("StringBuffer time: " + stringBufferTime + "ms");

// Using StringBuilder startTime = System.currentTimeMillis();

StringBuilder stringBuilder = new StringBuilder();

for (int i = 0; i < iterations; i++) {

stringBuilder.append("Hello, ");

}

endTime = System.currentTimeMillis(); long stringBuilderTime = endTime - startTime;

System.out.println("StringBuilder time: " + stringBuilderTime + "ms");

}

}

\*/

/\*88.Build a simple media player application with a base class "MediaPlayer" and subclasses "AudioPlayer" and "VideoPlayer." Each subclass should have a method "play" to display messages like "Playing audio" or "Playing video."

Utilize polymorphism to play different media types.

package com.gqt.strings.project;

class MediaPlayer { public void play() {

System.out.println("Playing media");

}

}

class AudioPlayer extends MediaPlayer {

@Override public void play() {

System.out.println("Playing audio");

}

}

class VideoPlayer extends MediaPlayer {

@Override public void play() {

System.out.println("Playing video");

}

}

public class String1 { public static void main(String[] args) {

MediaPlayer mediaPlayer = new MediaPlayer();

AudioPlayer audioPlayer = new AudioPlayer();

VideoPlayer videoPlayer = new VideoPlayer();

mediaPlayer.play(); // Output: Playing media audioPlayer.play(); // Output: Playing audio videoPlayer.play(); // Output: Playing video

}

}

\*/

/\*89.Create a Java class hierarchy for geometric shapes, including a base class "Shape" with methods for calculating area and perimeter. Implement two subclasses, "Circle" and "Rectangle," that inherit from the "Shape" class and provide their own implementations of area and perimeter calculations.(Inheritance)

package com.gqt.strings.project; //Base class "Shape" class Shape {

// Common properties and constructor (e.g., color)

// ...

// Method to calculate area (to be overridden by subclasses) public double calculateArea() {

return 0.0;

}

// Method to calculate perimeter (to be overridden by subclasses)

public double calculatePerimeter() {

return 0.0;

}

}

//Subclass "Circle" class Circle extends Shape { private double radius;

public Circle(double radius) { this.radius = radius;

}

@Override

public double calculateArea() { return Math.PI \* radius \* radius;

}

@Override

public double calculatePerimeter() { return 2 \* Math.PI \* radius;

}

}

//Subclass "Rectangle" class Rectangle extends Shape { private double length; private double width;

public Rectangle(double length, double width) { this.length = length; this.width = width;

}

@Override public double calculateArea() { return length \* width;

}

@Override

public double calculatePerimeter() { return 2 \* (length + width);

}

}

public class String1 {

public static void main(String[] args) { Circle circle = new Circle(5.0);

Rectangle rectangle = new Rectangle(4.0, 6.0);

// Calculate and print area and perimeter for Circle

System.out.println("Circle - Area: " + circle.calculateArea() + ", Perimeter: " + circle.calculatePerimeter());

// Calculate and print area and perimeter for Rectangle

System.out.println("Rectangle - Area: " + rectangle.calculateArea() + ", Perimeter: " + rectangle.calculatePerimeter());

}

}

\*/

/\*90. Define a class "Animal" with properties like name, age, and sound. Create two subclasses, "Dog" and "Cat," that inherit from the "Animal" class. Add methods to both subclasses to make them produce their respective sounds. Demonstrate polymorphism by creating instances of each subclass and calling the sound methods. (Inheritance)

package com.gqt.strings.project;

class Animal {

private String name; private int age; private String sound;

public Animal(String name, int age, String sound) {

this.name = name; this.age = age; this.sound = sound;

}

public void makeSound() {

System.out.println(name + " makes a sound: " + sound);

}

}

class Dog extends Animal { public Dog(String name, int age) {

super(name, age, "bowbow");

}

}

class Cat extends Animal { public Cat(String name, int age) {

super(name, age, "Meow");

}

}

public class String1 {

public static void main(String[] args) {

Animal dog = new Dog("Buddy", 3);

Animal cat = new Cat("Whiskers", 2); dog.makeSound(); // Output: Buddy makes a sound: Woof

cat.makeSound(); // Output: Whiskers makes a sound: Meow

}

}

\*/

/\*91.Design a class hierarchy for a banking system. Create a base class "Account" with fields for account number, account holder name, and balance. Implement two subclasses, "SavingsAccount" and

"CheckingAccount," that inherit from the "Account" class. Add methods for deposit, withdrawal, and account-specific behaviors. (Inheritance)

package com.gqt.strings.project;

//Base class Account class Account {

private int accountNumber; private String accountHolderName; private double balance;

public Account(int accountNumber, String accountHolderName, double balance) { this.accountNumber = accountNumber; this.accountHolderName = accountHolderName; this.setBalance(balance);

}

public int getAccountNumber() { return accountNumber;

}

public String getAccountHolderName() { return accountHolderName;

}

public double getBalance() {

return balance;

}

public void deposit(double amount) {

if (amount > 0) {

setBalance(getBalance() + amount);

System.out.println("Deposited $" + amount);

} else {

System.out.println("Invalid deposit amount.");

}

}

public void withdraw(double amount) { if (amount > 0 && amount <= getBalance()) { setBalance(getBalance() - amount);

System.out.println("Withdrawn $" + amount);

} else {

System.out.println("Invalid withdrawal amount or insufficient balance.");

}

}

public void displayAccountInfo() {

System.out.println("Account Number: " + accountNumber);

System.out.println("Account Holder: " + accountHolderName);

System.out.println("Balance: $" + getBalance());

}

public void setBalance(double balance) { this.balance = balance;

}

}

//Subclass SavingsAccount

class SavingsAccount extends Account { private double interestRate;

public SavingsAccount(int accountNumber, String accountHolderName, double balance, double interestRate) {

super(accountNumber, accountHolderName, balance); this.interestRate = interestRate;

}

public void applyInterest() {

double interest = getBalance() \* interestRate; deposit(interest);

System.out.println("Interest applied: $" + interest);

}

}

//Subclass CheckingAccount

class CheckingAccount extends Account { private double overdraftLimit;

public CheckingAccount(int accountNumber, String accountHolderName, double balance, double overdraftLimit) {

super(accountNumber, accountHolderName, balance); this.overdraftLimit = overdraftLimit;

}

@Override

public void withdraw(double amount) {

if (amount > 0 && amount <= (getBalance() + overdraftLimit)) { setBalance(getBalance() - amount);

System.out.println("Withdrawn $" + amount);

} else {

System.out.println("Invalid withdrawal amount or exceeded overdraft limit.");

}

}

}

public class String1 {

public static void main(String[] args) {

|  |  |
| --- | --- |
|  | // Create a savings account |
|  | SavingsAccount savingsAccount = new SavingsAccount(12345, "John Doe", 1000.0, 0.05); |
|  | savingsAccount.displayAccountInfo(); |
|  | savingsAccount.deposit(500.0); |
|  | savingsAccount.applyInterest(); |
|  | savingsAccount.withdraw(200.0); |
|  | savingsAccount.displayAccountInfo(); |
|  | // Create a checking account |
| 100.0); | CheckingAccount checkingAccount = new CheckingAccount(67890, "Jane Smith", 500.0, |
|  | checkingAccount.displayAccountInfo(); |
|  | checkingAccount.deposit(200.0); |
|  | checkingAccount.withdraw(800.0); // Attempt to exceed overdraft limit |
| }  }  \*/ | checkingAccount.displayAccountInfo(); |

/\*92.Create a "Person" class with properties like name and address. Implement a subclass "Student" that inherits from "Person" and includes additional properties like student ID and GPA. Then, create another subclass "Teacher" that inherits from "Person" and has properties like employee ID and subject taught.

package com.gqt.strings.project;

//Person class class Person { private String name; private String address;

public Person(String name, String address) {

this.name = name; this.address = address;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) { this.address = address;

}

}

//Student class (subclass of Person) class Student extends Person { private int studentId; private double gpa;

public Student(String name, String address, int studentId, double gpa) {

super(name, address); this.studentId = studentId; this.gpa = gpa;

}

public int getStudentId() {

return studentId;

}

public void setStudentId(int studentId) { this.studentId = studentId;

}

public double getGpa() {

return gpa;

}

public void setGpa(double gpa) {

this.gpa = gpa;

}

}

//Teacher class (subclass of Person) class Teacher extends Person { private int employeeId; private String subjectTaught;

public Teacher(String name, String address, int employeeId, String subjectTaught) { super(name, address); this.employeeId = employeeId; this.subjectTaught = subjectTaught;

}

public int getEmployeeId() { return employeeId;

}

public void setEmployeeId(int employeeId) { this.employeeId = employeeId;

}

public String getSubjectTaught() { return subjectTaught;

}

public void setSubjectTaught(String subjectTaught) { this.subjectTaught = subjectTaught;

}

}

//Sample program to demonstrate usage public class String1 {

public static void main(String[] args) {

Student student = new Student("John Doe", "123 Main St", 101, 3.8);

Teacher teacher = new Teacher("Jane Smith", "456 Elm St", 201, "Mathematics");

// Display student information

System.out.println("Student Information:");

System.out.println("Name: " + student.getName());

System.out.println("Address: " + student.getAddress());

System.out.println("Student ID: " + student.getStudentId());

System.out.println("GPA: " + student.getGpa());

// Display teacher information

System.out.println("\nTeacher Information:");

System.out.println("Name: " + teacher.getName());

System.out.println("Address: " + teacher.getAddress());

System.out.println("Employee ID: " + teacher.getEmployeeId());

System.out.println("Subject Taught: " + teacher.getSubjectTaught());

}

}

\*/

/\*93.Create a Java class to represent a "Book" with private fields for title, author, and price. Provide encapsulated methods to get and set these fields. Write a sample program to demonstrate its usage.

package com.gqt.strings.project;

public class String1 { private String title; private String author; private double price;

// Constructor

public String1(String title, String author, double price) { this.title = title; this.author = author; this.price = price;

}

// Getter for title public String getTitle() { return title;

}

// Setter for title public void setTitle(String title) { this.title = title;

}

// Getter for author

public String getAuthor() {

return author;

|  |  |
| --- | --- |
|  | }  // Setter for author  public void setAuthor(String author) { this.author = author;  }  // Getter for price public double getPrice() {  return price;  }  // Setter for price  public void setPrice(double price) { this.price = price;  }  // Sample program to demonstrate usage public static void main(String[] args) {  // Create a Book object  String1 myBook = new String1("The Great Gatsby", "F. Scott Fitzgerald", 12.99);  // Access and print the book's information  System.out.println("Title: " + myBook.getTitle());  System.out.println("Author: " + myBook.getAuthor());  System.out.println("Price: $" + myBook.getPrice());  // Update the book's price myBook.setPrice(14.99);  // Display the updated price  System.out.println("Updated Price: $" + myBook.getPrice()); |
| } | } |

\*/

/\*94.Implement a class called "Employee" with private fields for name, salary, and employee ID. Ensure encapsulation and provide a method to give a yearly bonus to the employee. Write a program to test this class.

package com.gqt.strings.project; public class String1 {

private String name; private double salary; private int employeeID;

// Constructor to initialize name, salary, and employeeID public String1(String name, double salary, int employeeID) {

this.name = name; this.salary = salary; this.employeeID = employeeID;

}

// Getter method to get the name public String getName() { return name;

}

// Setter method to set the name public void setName(String name) {

this.name = name;

}

// Getter method to get the salary public double getSalary() { return salary;

}

|  |  |
| --- | --- |
|  | // Setter method to set the salary public void setSalary(double salary) { this.salary = salary;  }  // Getter method to get the employee ID public int getEmployeeID() { return employeeID;  }  // Method to give a yearly bonus to the employee public void giveYearlyBonus(double bonus) {  salary += bonus;  }  public static void main(String[] args) {  // Create an Employee object  String1 employee = new String1("John Doe", 50000.0, 12345);  // Display employee information  System.out.println("Employee Name: " + employee.getName());  System.out.println("Employee Salary: " + employee.getSalary());  System.out.println("Employee ID: " + employee.getEmployeeID());  // Give a yearly bonus of $2000 employee.giveYearlyBonus(2000.0);  // Display updated salary after the bonus  System.out.println("Updated Salary after Bonus: " + employee.getSalary()); |
| }  \*/ | } |

/\*95.Create a "Circle" class with a private field for radius. Implement encapsulated methods to set and get the radius and calculate the area of the circle. Write a program to calculate the area of a circle using this class.

package com.gqt.strings.project;

public class String1 {

private double radius;

// Constructor to initialize the radius public String1(double radius) { this.radius = radius;

}

// Getter method to get the radius public double getRadius() {

return radius;

}

// Setter method to set the radius public void setRadius(double radius) { this.radius = radius;

}

// Method to calculate the area of the circle public double calculateArea() { return Math.PI \* radius \* radius;

}

public static void main(String[] args) {

// Create a Circle object with a radius of 5.0

String1 circle = new String1(5.0);

// Get the radius double radius = circle.getRadius();

// Calculate and display the area double area = circle.calculateArea();

System.out.println("Radius: " + radius);

System.out.println("Area: " + area);

}

}

\*/

/\*96. Design a class called "Person" with private fields for name, age, and gender. Ensure encapsulation and provide a method to check if a person is eligible to vote (age >= 18). Write a program to test this class.

package com.gqt.strings.project; public class String1 {

private String name; private int age; private String gender;

// Constructor

public String1(String name, int age, String gender) { this.name = name; this.age = age; this.gender = gender;

}

// Getter methods public String getName() {

return name;

}

public int getAge() {

return age;

}

public String getGender() {

return gender;

}

// Method to check if the person is eligible to vote public boolean isEligibleToVote() {

return age >= 18;

}

public static void main(String[] args) {

// Creating a Person object

String1 person1 = new String1("John", 25, "Male");

// Testing eligibility to vote if (person1.isEligibleToVote()) {

System.out.println(person1.getName() + " is eligible to vote.");

} else {

System.out.println(person1.getName() + " is not eligible to vote.");

}

}

}

\*/

/\*97.Implement a "Bank" class that manages customer accounts. Each account should have a private balance field. Provide methods for deposit, withdrawal, and checking the balance, ensuring that the balance cannot go below zero. Write a program to simulate banking operations.

package com.gqt.strings.project; import java.util.Scanner; class BankDetails { private String accno; private String name; private String acc\_type; private long balance;

Scanner sc = new Scanner(System.in); //method to open new account public void openAccount() {

System.out.print("Enter Account No: "); accno = sc.next();

System.out.print("Enter Account type: "); acc\_type = sc.next(); System.out.print("Enter Name: "); name = sc.next();

System.out.print("Enter Balance: "); balance = sc.nextLong();

}

//method to display account details public void showAccount() {

System.out.println("Name of account holder: " + name);

System.out.println("Account no.: " + accno);

System.out.println("Account type: " + acc\_type);

System.out.println("Balance: " + balance);

}

//method to deposit money public void deposit() {

long amt;

System.out.println("Enter the amount you want to deposit: "); amt = sc.nextLong(); balance = balance + amt;

}

//method to withdraw money

public void withdrawal() {

long amt;

System.out.println("Enter the amount you want to withdraw: "); amt = sc.nextLong(); if (balance >= amt) {

balance = balance - amt;

System.out.println("Balance after withdrawal: " + balance);

} else {

System.out.println("Your balance is less than " + amt + "\tTransaction failed...!!" );

}

}

//method to search an account number public boolean search(String ac\_no) { if (accno.equals(ac\_no)) {

showAccount(); return (true);

}

return (false);

}

}

public class String1 {

public static void main(String arg[]) {

Scanner sc = new Scanner(System.in);

//create initial accounts

System.out.print("How many number of customers do you want to input? "); int n = sc.nextInt();

BankDetails C[] = new BankDetails[n]; for (int i = 0; i < C.length; i++) {

C[i] = new BankDetails();

C[i].openAccount();

}

// loop runs until number 5 is not pressed to exit int ch; do {

System.out.println("\n \*\*Banking System Application\*\*");

System.out.println("1. Display all account details \n 2. Search by Account number\n

3. Deposit the amount \n 4. Withdraw the amount \n 5.Exit ");

System.out.println("Enter your choice: "); ch = sc.nextInt(); switch (ch) { case 1:

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

C[i].showAccount();

}

break; case 2:

System.out.print("Enter account no. you want to search: "); String ac\_no = sc.next(); boolean found = false;

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

found = C[i].search(ac\_no); if (found) {

break;

}

}

if (!found) {

System.out.println("Search failed! Account doesn't exist..!!");

}

break; case 3:

System.out.print("Enter Account no. : "); ac\_no = sc.next(); found = false;

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

found = C[i].search(ac\_no); if (found) { C[i].deposit();

break;

}

}

if (!found) {

System.out.println("Search failed! Account doesn't exist..!!");

}

break; case 4:

System.out.print("Enter Account No : "); ac\_no = sc.next(); found = false;

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

found = C[i].search(ac\_no); if (found) { C[i].withdrawal();

break;

}

}

if (!found) {

System.out.println("Search failed! Account doesn't exist..!!");

}

break; case 5:

System.out.println("See you soon..."); break;

}

}

while (ch != 5);

}

}

\*/

/\*98.Problem Statement –

A chocolate factory is packing chocolates into the packets. The

chocolate packets here represent an array of N number of integer values. The task is to find the empty packets(0) of chocolate and push it to the end of the conveyor belt(array).

Example 1 :

N=8 and arr = [4,5,0,1,9,0,5,0].

There are 3 empty packets in the given set. These 3 empty packets represented as O should be pushed towards the end of the array Input :

8 – Value of N

[4,5,0,1,9,0,5,0] – Element of arr[O] to arr[N-1],While input each element is separated by newline Output: 4 5 1 9 5 0 0 0 Example 2:

Input:

6 — Value of N.

[6,0,1,8,0,2] – Element of arr[0] to arr[N-1], While input each element is separated by newline Output:

6 1 8 2 0 0

package com.gqt.strings.project;

import java.util.Scanner;

public class String1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("enter the size"); // Read the value of N int N = sc.nextInt(); sc.nextLine(); // Consume the newline // Read the array elements int[] arr = new int[N]; for (int i = 0; i < N; i++) {

arr[i] = sc.nextInt();

}

// Process the array to push empty packets to the end int emptyPacketCount = 0; for (int i = 0; i < N; i++) { if (arr[i] != 0) { arr[emptyPacketCount++] = arr[i];

}

}

// Fill the remaining positions with 0s while (emptyPacketCount < N) { arr[emptyPacketCount++] = 0;

}

// Print the modified array for (int i = 0; i < N; i++) {

System.out.print(arr[i] + " ");

}

sc.close();

}

}

\*/

/\*99. Selection of MPCS exams include a fitness test which is conducted on ground. There will be a batch of 3 trainees, appearing for running test in track for 3 rounds. You need to record their oxygen level after every round. After trainee are finished with all rounds, calculate for each trainee his average oxygen level over the 3 rounds and select one with highest oxygen level as the most fit trainee. If more than one trainee attains the same highest average level, they all need to be selected. Display the most fit trainee (or trainees) and the highest average oxygen level.

Note:



95

92

95

92

90

92

90

92

90



The oxygen value entered should not be accepted if it is not in the range between 1 and 100.

If the calculated maximum average oxygen value of trainees is below 70 then declare the trainees as unfit with meaningful message as “All trainees are unfit.

Average Oxygen Values should be rounded.

Example1:

INPUTVALUES

OUTPUTVALUES

TraineeNumber:1 Trainee Number : 3 Note:

Input should be 9 integer values representing oxygen levels entered in order as Round 1

Oxygen value of trainee 1

Oxygen value of trainee 2

Oxygen value of trainee 3

Round 2

Oxygen value of trainee 1

Oxygen value of trainee 2



Oxygen value of trainee 3

Round 3

Oxygen value of trainee 1

Oxygen value of trainee 2

Oxygen value of trainee 3

Output must be in given format as in above example. For any wrong input final output should display “INVALID INPUT”

package com.gqt.strings.project; import java.util.Scanner; public class String1 { public static void main(String[] args) { int[][] trainee = new int[3][3]; int[] average = new int[3]; int max = 0;

Scanner sc = new Scanner(System.in); for(int i = 0; i < 3; i++) { for(int j = 0; j < 3; j++) { trainee[i][j] = sc.nextInt(); if(trainee[i][j] < 1 || trainee[i][j] > 100) { trainee[i][j] = 0;

}

}

}

for(int i = 0; i < 3; i++) { for(int j = 0; j < 3; j++) { average[i] = average[i] + trainee[j][i];

}

average[i] = average[i] / 3;

}

for(int i = 0; i < 3; i++) {

if(average[i] > max) {

max = average[i];

}

}

for(int i = 0; i < 3; i++) {

if(average[i] == max) {

System.out.println("Trainee Number : " + (i + 1));

}

if(average[i] <70) {

System.out.print("Trainee is Unfit");

}

}

}

}

\*/

/\*100.The Caesar cipher is a type of substitution cipher in which each alphabet in the plaintext or messages is shifted by a number of places down the alphabet.

For example,with a shift of 1, P would be replaced by Q, Q would become R, and so on.

To pass an encrypted message from one person to another, it is first necessary that both parties have the ‘Key’ for the cipher, so that the sender may encrypt and the receiver may decrypt it.

Key is the number of OFFSET to shift the cipher alphabet. Key can have basic shifts from 1 to 25 positions as there are 26 total alphabets. As we are designing custom Caesar Cipher, in addition to alphabets, we are considering numeric digits from 0 to 9. Digits can also be shifted by key places.

For Example, if a given plain text contains any digit with values 5 and keyy =2, then 5 will be replaced by 7, “-”(minus sign) will remain as it is.

Key value less than 0 should result into “INVALID INPUT” Example 1:

Enter your PlainText: All the best

Enter the Key: 1

The encrypted Text is: Bmm uif Cftu

Write a function CustomCaesarCipher(int key, String message) which will accept plaintext and key as input parameters and returns its cipher text as output.\*/

package com.gqt.strings.project; import java.util.Scanner; public class String1 { public static String CustomCaesarCipher(int key, String message) {

if (key < 0) { return "INVALID INPUT";

}

StringBuilder cipherText = new StringBuilder();

for (char character : message.toCharArray()) { if (Character.isLetter(character)) { char base = Character.isLowerCase(character) ? 'a' : 'A'; cipherText.append((char) ((character - base + key) % 26 + base));

} else if (Character.isDigit(character)) { int digit = Character.getNumericValue(character); int shiftedDigit = (digit + key) % 10; cipherText.append(shiftedDigit);

} else {

cipherText.append(character); // Keep non-alphanumeric characters unchanged

}

}

return cipherText.toString();

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter your PlainText: ");

String plainText = scanner.nextLine(); System.out.print("Enter the Key: "); int key = scanner.nextInt(); scanner.close();

String cipherText = CustomCaesarCipher(key, plainText);

System.out.println("The encrypted Text is: " + cipherText);

}

}