**Assignment 1**

Following things to be added in each question:

-Program

-Flow chart

-Explanation

-Output

-Time and Space complexity

Submission Date: 26/09/2024

1. Armstrong Number

Problem: Write a Java program to check if a given number is an Armstrong number.

Test Cases:

Input: 153

Output: true

Input: 123

Output: false

-->>

import java.util.Scanner;

class ArmStrongNo {

public static void main (String args[]) {

Scanner sc = new Scanner (System.in);

int a,b, d, sum = 0;

System.out.println("Enter a number :");

b = sc.nextInt();

a = b;

while (b > 0)

{

d = b % 10;

sum = sum+(d\*d\*d);

b = b / 10;

}

if (a == sum)

System.out.println(true);

else

System.out.println(false);

}

}

2. Prime Number

Problem: Write a Java program to check if a given number is prime.

Test Cases:

Input: 29

Output: true

Input: 15

Output: false

-->>

import java.util.Scanner;

class PrimeNo{

public static void main (String args[] ) {

System.out.println("Enter any number of your choice to check prime: ");

Scanner sc = new Scanner (System.in);

int num = sc.nextInt();

boolean flag = false;

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

if (num % i == 0) {

flag = true;

break;

}

}

if (!flag)

System.out.println(true);

else

System.out.println(false);

}

}

3. Factorial

Problem: Write a Java program to compute the factorial of a given number.

Test Cases:

Input: 5

Output: 120

Input: 0

Output: 1

-->>

import java.util.Scanner;

class Factorial {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter any number : ");

int fact=1;

int n = sc.nextInt();

for (int i=1; i<=n; i++)

{

fact=fact\*i;

}

System.out.println("The factorial of the number " +n+ " is " +fact);

}

}

4. Fibonacci Series

Problem: Write a Java program to print the first n numbers in the Fibonacci series.

Test Cases:

Input: n = 5

Output: [0, 1, 1, 2, 3]

Input: n = 8

Output: [0, 1, 1, 2, 3, 5, 8, 13]

-->>

import java.util.Scanner;

class Fibonacci {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number : ");

int n = sc.nextInt();

int a=0, b=1;

System.out.println("Fibonacci series till: "+n+" terms");

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

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

int c = a + b;

a = b;

b = c;

}

}

}

5. Find GCD

Problem: Write a Java program to find the Greatest Common Divisor (GCD) of two numbers.

Test Cases:

Input: a = 54, b = 24

Output: 6

Input: a = 17, b = 13

Output: 1

-->>

public class GCD {

public static int euclideanGCD(int a, int b) {

while (b != 0) {

int temp = a;

a = b;

b = temp % b;

}

return a;

}

public static void main(String[] args) {

int num1 = 54;

int num2 = 24;

int num3 = 13;

int num4 = 17;

int gcd1 = euclideanGCD(num1, num2);

System.out.println("GCD of " + num1 + " and " + num2 + " is: " + gcd1);

int gcd2 = euclideanGCD(num3, num4);

System.out.println("GCD of " + num3 + " and " + num4 + " is: " + gcd2);

}

}

6. Find Square Root

Problem: Write a Java program to find the square root of a given number (using integer approximation).

Test Cases:

Input: x = 16

Output: 4

Input: x = 27

Output: 5

-->>

import java.util.Scanner;

class SquareRoot {

public static void main (String args[]) {

Scanner sc = new Scanner (System.in);

System.out.println("Enter the number : ");

double x = sc.nextDouble();

double ans = (int) Math.sqrt(x);

System.out.println(ans);

}

}

7. Find Repeated Characters in a String

Problem: Write a Java program to find all repeated characters in a string.

Test Cases:

Input: "programming"

Output: ['r', 'g', 'm']

Input: "hello"

Output: ['l']

-->>

import java.util.Scanner;

public class RCString {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

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

String input = sc.nextLine();

int[] charCount = new int[26];

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

charCount[input.charAt(i) - 'a']++;

}

System.out.print("Repeated characters: ");

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

if (charCount[i] > 1) {

System.out.print((char) ('a' + i) + ", ");

}

}

}

}

8. First Non-Repeated Character

Problem: Write a Java program to find the first non-repeated character in a string.

Test Cases:

Input: "stress"

Output: 't'

Input: "aabbcc"

Output: null

-->>

import java.util.Scanner;

public class NRCharacter {

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

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

String str = sc.nextLine();

char[] arr = str.toCharArray();

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

{

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

{

if(arr[i] != arr[j])

{

System.out.println(arr[j]);

System.exit(0);

}

else

{

System.out.println("null");

System.exit(0);

}

}

}

}

}

9. Integer Palindrome

Problem: Write a Java program to check if a given integer is a palindrome.

Test Cases:

Input: 121

Output: true

Input: -121

Output: false

-->>

import java.util.Scanner;

public class Palindrome {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

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

String input = sc.nextLine();

boolean isPalindrome = true;

int left = 0;

int right = input.length() - 1;

while (left < right) {

if (input.charAt(left) != input.charAt(right)) {

isPalindrome = false;

break;

}

left++;

right--;

}

if (isPalindrome) {

System.out.println(input + " is a palindrome.");

} else {

System.out.println(input + " is not a palindrome.");

}

}

}

10. Leap Year

Problem: Write a Java program to check if a given year is a leap year.

Test Cases:

Input: 2020

Output: true

Input: 1900

Output: false

-->>

import java.util.Scanner;

class LeapYear {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the year : ");

int year = sc.nextInt();

if (year % 4 == 0 && year %100 != 0 || year % 400 == 0)

{

System.out.println(true);

}

else

{

System.out.println(false);

}

}

}