**C-DAC Mumbai Date 25/09/2024**

**Subject: Algorithm and Data Structure**

**Assignment 1**

**Solve the assignment with following thing to be added in each question.**

-Program

-Flow chart

-Explanation

-Output

-Time and Space complexity

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

Answer -----

-----------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner;

public class ArmstrongNumber {

public static boolean isArmstrong(int num){

int original = num ;

int sum = 0 ;

int n = String.valueOf(num).length();

while (num != 0){

int digit = num % 10;

sum += Math.pow(digit,n);

num /=10;

}

return sum == original;

}

public static void main(String[] args) {

Scanner sc =new Scanner(System.in);

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

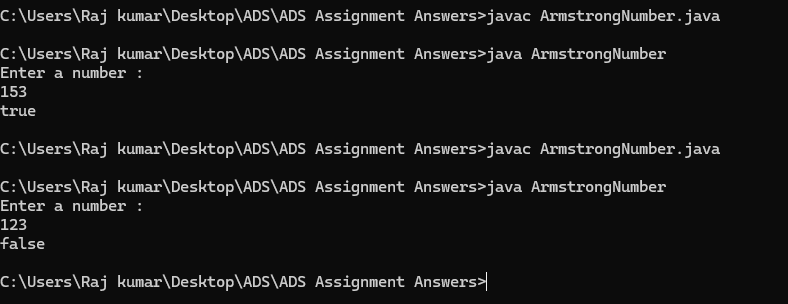
int num = sc.nextInt();

System.out.println(isArmstrong(num));

sc.close();

}

}



**Flowchart:**

1.Start

2.Input the number

3.Store the number in a variable (original)

4. Count number of digits

5. Initialize sum to 0

6. For each digit of given number :

> Extract the last digit

> Raise the digit to the power of no of digits > Add result to (sum)

> Remove the last digit from the number

7.If sum is equal to original,return TRUE otherwise return FALSE

8.End

**Time Complexity :**

O(d),where d is no of digits in the input number.

The program extracts each digit,raises it to the power of d , and sums the result.

**Space Complexity :**

O(1) : only few integer variables are used to store the result

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

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

Answer---

-------------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner;

public class PrimeNo {

public static void main(String[] args) { Scanner sc = new Scanner(System.in); System.out.println("Enter a number:"); int n=sc.nextInt();

boolean flag= false;

if(n==0 || n==1) { flag=true;

} fo r (int i = 2; i <= n / 2; ++i) {

if (n % i == 0) {

flag = true; break;

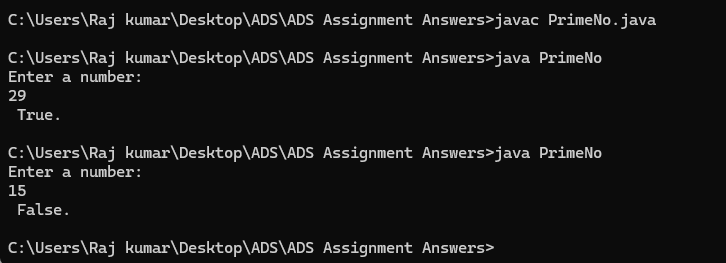
} } if (!flag)

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

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

sc.close(); }

}



**Flowchart**

1. Start

2. Input number n from user

3. Initialize flag to false

4. Check if n is 0 or 1

5. For loop from i=2 to n/2

Check if (n%i ==0)

- If yes set flag= true and break the loop

- If no continue the loop

6. Check if flag is false

If true, print n is a prime number If false, print n is not a prime number

7. end

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

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

Answer-----

-------------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner;

public class Factorial {

public static void main(String[] args) {

// Create a scanner object for input

Scanner scanner = new Scanner(System.in);

// Prompt user to enter a number

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

int number = scanner.nextInt();

// Compute factorial using a helper method

long result = factorial(number);

// Output the result

System.out.println("Factorial of " + number + " is: " + result);

// Close scanner

scanner.close();

}

// Method to calculate factorial

public static long factorial(int n) {

if (n == 0 || n == 1) {

return 1;

}

long fact = 1;

for (int i = 2; i <= n; i++) {

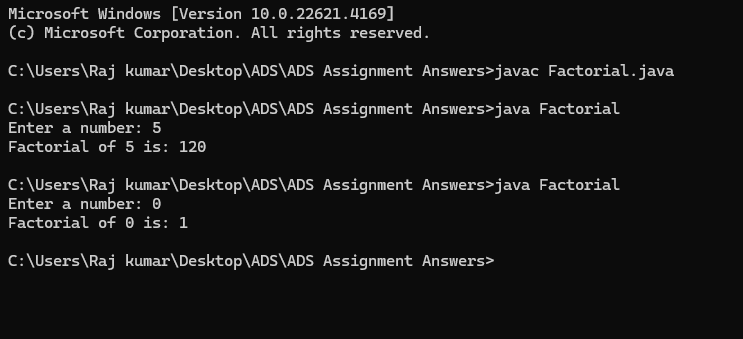
fact \*= i;

}

return fact;

}

}



**Explanation:**

The program takes an integer input from the user.

The factorial method calculates the factorial using an iterative approach.

If the input is 0, it returns 1 since 0! is defined as 1.

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

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]

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

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

Answer

import java.util.\*;

public class DSA\_Assignment\_1\_Q2 {

        public static void main(String [] args){

            Scanner sc = new Scanner(System.in);

            int ans = 0;

            boolean b = true;

            int num = sc.nextInt();

            for(int i=1; i<num; i++){

                if(num == i\*i){

                   System.out.println(i);

                   b = false;

                }

            }

            for(int i=1; i<num; i++){

                if(num < i\*i){

                    ans = i-1;

                    break;

                }

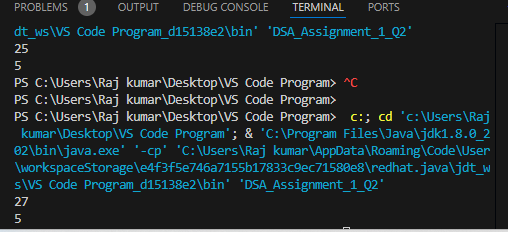
            }

            if(b)System.out.println(ans);

            sc.close();

        }

        }



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

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']

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

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

Answer

----------------------------------------------------------------------------------------------------------------------

public class IntegerPalindrome {

public static boolean isPalindrome(int x) {

// If the number is negative or ends with 0 (but isn't 0), it's not a palindrome

if (x < 0 || (x % 10 == 0 && x != 0)) {

return false;

}

int reversed = 0;

int original = x;

// Reverse the number

while (x > 0) {

int lastDigit = x % 10;

reversed = reversed \* 10 + lastDigit;

x /= 10;

}

// Check if the original number is the same as the reversed number

return original == reversed;

}

public static void main(String[] args) {

// Test cases

System.out.println(isPalindrome(121)); // true

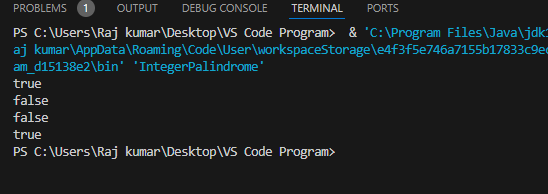
System.out.println(isPalindrome(-121)); // false

System.out.println(isPalindrome(10)); // false

System.out.println(isPalindrome(0)); // true

}

}



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

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