**ADS Assignment 1**

1. Amstrong Number

import java.util.\*;

public class AmstrongNumber {

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

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

int num = sc.nextInt();

sc.close();

int originalNum = num;

int sum = 0;

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

while(num > 0){

int digit = num % 10;

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

num /= 10;

}

String res = (sum == originalNum)? (originalNum + " is an AmstrongNumber."):(originalNum + " is not an AmstrongNumber.");

System.out.println(res);

}

}

**o/p:** Enter a number:

153

153 is an AmstrongNumber.

1. Prime Number

import java.util.\*;

class PrimeNumber {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

if(n % 2 == 0)

{

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

}

else{

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

}

}

}

**o/p:** 2 is prime.

1. Factorial

import java.util.\*;

public class Factorial{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

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

int n = sc.nextInt();

sc.close();

int fact = 1;

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

fact \*= i;

}

System.out.println("Factorial of " + n + " is " + fact);

}

}

**o/p:** Enter a number:

5

Factorial of 5 is 120

1. Fibonacci Series

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

import java.util.Scanner;

class Fibonacci {

static int fibonacci(int n){

if(n <= 1)

return n;

return fibonacci(n-1) + fibonacci(n-2);

}

public static void printFibonacci(int n){

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

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

}

System.out.println();

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

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

int n = sc.nextInt();

sc.close();

printFibonacci(n);

}

}

o/p: Enter number:

8

0 1 2 3 4 5 6 7

1. Find GCD

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

import java.util.Scanner;

public class GCD

{

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

int gcd = 1;

for (int i = 1; i <= Math.min(a, b); i++) {

if (a % i == 0 && b % i == 0) {

gcd = i;

}

}

return gcd;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the first number: ");

int num1 = scanner.nextInt();

System.out.print("Enter the second number: ");

int num2 = scanner.nextInt();

int gcd = findGCD(num1, num2);

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

scanner.close();

}

}

o/p: Enter the first number: 6

Enter the second number: 12

The GCD of 6 and 12 is: 6

1. 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

1. Find Repeated Characters in a String

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

public class RepeatedCharacters {

public static void main(String[] args) {

String input1 = "programming";

String input2 = "hello";

System.out.println("Repeated Characters in a string: " + findRepeatedCharacters(input1));

System.out.println("Repeated Characters in a string: " + findRepeatedCharacters(input2));

}

public static String findRepeatedCharacters(String str) {

int[] charCount = new int[256];

String result = "";

for (char c : str.toCharArray()) {

charCount[c]++;

}

for (char c : str.toCharArray()) {

if (charCount[c] > 1 && result.indexOf(c) == -1) {

result += c;

}

}

return result;

}

}

**o/p:**

Repeated Characters in a string: rgm

Repeated Characters in a string: 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

Program:

import java.util.HashMap;

public class FirstNonRepeatedCharacter {

public static Character firstNonRepeated(String str) {

HashMap<Character, Integer> charCountMap = new HashMap<>();

for (char c : str.toCharArray()) {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

for (char c : str.toCharArray()) {

if (charCountMap.get(c) == 1) {

return c;

}

}

return null;

}

public static void main(String[] args) {

String test1 = "stress";

String test2 = "aabbcc";

System.out.println("First non repeated character in a string1 : " + firstNonRepeated(test1));

System.out.println("First non repeated character in a string2 : " +firstNonRepeated(test2));

}

}

**o/p:**

First non repeated character in a string1 : t

First non repeated character in a string2 : null

9. Integer Palindrome

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

Code:

import java.util.Scanner;

class IntegerPalindrome {

public static boolean isPalindrome(int x) {

if (x < 0) return false;

int original = x;

int reversed = 0;

while (x != 0) {

int digit = x % 10;

reversed = reversed \* 10 + digit;

x /= 10;

}

return original == reversed;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter an integer: ");

int num = sc.nextInt();

if (isPalindrome(num)) {

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

} else {

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

}

sc.close();

}

}

Test Cases:

Input: 121

Output: true

Input: -121

Output: false

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

Code:

import java.util.Scanner;

class LeapYear {

public static boolean isLeapYear(int year) {

if (year % 4 == 0) {

if (year % 100 != 0 || year % 400 == 0) {

return true;

}

}

return false;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

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

int year = sc.nextInt();

if (isLeapYear(year)) {

System.out.println(year + " is a leap year.");

} else {

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

}

sc.close();

}

}