****Capstone project** :-**ASSIGNMENT****

**Problem Statement**

Write a Java program to print all natural numbers from 1 to n using loop. C program to print first n natural numbers using loop

**Example**

**Input**

Input upper limit: 10

**Output**

Natural numbers from 1 to 10: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Naturalnum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int n = scanner.nextInt();

// scanner.close();

System.***out***.print("Natural numbers from 1 to " + n + ": ");

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

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

}

}

}

**Problem Statement**

Write a Java program to print all natural numbers in reverse from n to 1 using for loop.

**Example**

**Input**

Input N: 10

**Output**

Natural numbers from 10-1 in reverse:

10, 9, 8, 7, 6, 5, 4, 3, 2, 1

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Naturalrev {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int n = scanner.nextInt();

// scanner.close();

System.***out***.print("Natural numbers from 1 to " + n +":");

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

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

}

}

}

**Problem Statement**

Write a Java program to print all even numbers from 1 to n using for loop.

**Example**

**Input**

Input upper range: 10

**Output**

Even numbers between 1 to 10:

2, 4, 6, 8, 10

**Solution:-**

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int n = scanner.nextInt();

scanner.close();

System.out.print("Even numbers between 1 to " + n + ": ");

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

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

}

}

}

**Problem Statement**

Write a Java program to print all odd numbers from 1 to n using for loop. **Example**

**Input**

Input upper limit: 10

**Output**

Odd numbers between 1 to 10:

1, 3, 5, 7, 9

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Oddnum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int n = scanner.nextInt();

scanner.close();

System.***out***.print("Even numbers between 1 to " + n + ": ");

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

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

}

}

}

**Problem Statement**

Write a java program to find the sum of all natural numbers between 1 to n using for loop.

**Example**

**Input**

Input upper limit: 10

**Output**

Sum of natural numbers 1-10: 55

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Sumnaturalnum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int n = scanner.nextInt();

scanner.close();

int sum = 0;

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

sum +=i;

}

System.***out***.println("Sum of natural numbers is: " + sum);

}

}

**Problem Statement**

Write a Java program to input number from user and find sum of all even numbers between 1 to n.

**Example**

**Input**

Input upper limit of even number: 10

**Output**

Sum of even numbers between 1 to 10: 30

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Sumeven {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int n = scanner.nextInt();

scanner.close();

int sum = 0;

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

sum +=i;

}

System.***out***.println("Sum of even numbers is: " + sum);

}

}

**Problem Statement**

Write a Java program to input a number from user and count number of digits in the given integer using loop.

**Example**

**Input**

Input num: 35419

**Output**

Number of digits: 5

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Digitcount {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input num: ");

int num = scanner.nextInt();

scanner.close();

int Count = 0;

while (num != 0) {

num /= 10;

Count++;

}

System.***out***.println("Number of digits: " + Count);

}

}

**Problem Statement**

Write a Java program to input a number from user and find reverse of the given number.

**Example**

**Input**

Input number: 12345

**Output**

Reverse of 12345 = 54321

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Revnum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input number: ");

int num = scanner.nextInt();

scanner.close();

int reverse = 0;

while (num != 0) {

int digit = num % 10;

reverse = reverse \* 10 + digit;

num /= 10;

}

System.***out***.println("Reverse of " + num + " = " + reverse);

}

}

**Problem Statement**

Write a Java program to find power of a number using for loop. How to find power of a number without using built in library functions

**Logic to find power of any number**

1. Input base and exponents from user. Store it in two variables say *base* and *expo*.
2. Declare and initialize another variable to store power say power = 1.
3. Run a loop from 1 to *expo*, increment loop counter by 1 in each iteration. The loop structure must look similar to for(i=1; i<=expo; i++).
4. For each iteration inside loop multiply *power* with *num* i.e. power = power \* num.
5. Finally after loop you are left with power in *power* variable.

**Example**

**Input**

Input base: 2

Input exponent: 5

**Output**

2 ^ 5 = 32

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Powerloop {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input base: ");

int base = scanner.nextInt();

System.***out***.print("Input exponent: ");

int expo = scanner.nextInt();

scanner.close();

int power = 1;

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

power \*= base;

}

System.***out***.println(base + " ^ " + expo + " = " + power);

}

}

**Problem Statement**

Write a Java program to input a number from user and find all factors of the given number using for loop.

Step by step descriptive logic to find all factors of a number.

1. Input number from user. Store it in some variable say *num*.
2. Run a loop from 1 to *num*, increment 1 in each iteration. The loop structure should look like for(i=1; i<=num; i++).
3. For each iteration inside loop check current counter loop variable *i* is a factor of *num* or not. To check factor we [check divisibility of number](https://codeforwin.org/2015/05/c-program-to-check-whether-number-is-divisible-by-5-and-11.html" \o "https://codeforwin.org/2015/05/c-program-to-check-whether-number-is-divisible-by-5-and-11.html" \t "_blank) by performing [modulo division](https://codeforwin.org/2017/08/c-arithmetic-operators.html" \o "https://codeforwin.org/2017/08/c-arithmetic-operators.html" \t "_blank) i.e. if(num % i == 0) then *i* is a factor of *num*.

If *i* is a factor of *num* then print the value of *i*.

**Example**

**Input**

Input number: 12

**Output**

Factors of 12: 1, 2, 3, 4, 6, 12

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Factorfind {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input number: ");

int num = scanner.nextInt();

scanner.close();

System.***out***.print("Factors of " + num + ": ");

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

if (num % i == 0) {

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

}

}

}

}

**Problem Statement**

Write a Java program to input number from user and check whether number is Strong number or not.

Step by step descriptive logic to check strong number.

* Input a number from user to check for strong number. Store this in a variable say num. Copy it to a temporary variable for calculations purposes, say originalNum = num.
* Initialize another variable to store sum of factorial of digits, say sum = 0.
* Find last digit of the given number num. Store the result in a variable say lastDigit = num % 10.
* Find factorial of lastDigit. Store factorial in a variable say fact.
* Add factorial to sum i.e. sum = sum + fact.
* Remove last digit from num as it is not needed further.
* Repeat steps 3 to 6 till num > 0.
* After loop check condition for strong number. If sum == originalNum, then the given number is Strong number otherwise not.

**Example**

**Input**

Input number: 145

**Output**

145 is STRONG NUMBER

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Strongnumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input number: ");

int num = scanner.nextInt();

scanner.close();

int originalNum = num;

int sum = 0;

while (num > 0) {

int lastDigit = num % 10;

int fact = *factorial*(lastDigit);

sum =sum+fact;

num /= 10;

}

if (sum == originalNum) {

System.***out***.println(originalNum + " is a STRONG NUMBER");

} else {

System.***out***.println(originalNum + " is not a STRONG NUMBER");

}

}

public static int factorial(int n) {

int fact = 1;

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

fact \*= i;

}

return fact;

}

}

**Problem Statement**

Write a Java program to print Fibonacci series up to n terms using loop.

### What is Fibonacci series?

*Fibonacci series* is a series of numbers where the current number is the sum of previous two terms. For Example: 0, 1, 1, 2, 3, 5, 8, 13, 21, … , (n-1th + n-2th)

**Step by step descriptive logic to print n Fibonacci terms.**

* Input number of Fibonacci terms to print from user. Store it in a variable say terms.
* Declare and initialize three variables, I call it as Fibonacci magic initialization. a=0, b=1 and c=0.
* Here c is the current term, b is the n-1th term and a is n-2th term.
* Run a loop from 1 to terms, increment loop counter by 1. The loop structure should look like for(i=1; i<=term; i++). It will iterate through n terms
* Inside the loop copy the value of n-1th term to n-2th term i.e. a = b.
* Next, copy the value of nth to n-1th term b = c.
* Finally compute the new term by adding previous two terms i.e. c = a + b.
* Print the value of current Fibonacci term i.e. c

**Example**

**Input**

Input number of terms: 10

**Output**

Fibonacci series:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34

**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Fibbonasiseries {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input number of terms: ");

int terms = scanner.nextInt();

scanner.close();

int a = 0, b = 1, c = 0;

System.***out***.print("Fibonacci series: ");

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

System.***out***.print(c + " ");

a = b;

b = c;

c = a + b;

}

}

}

**Problem Statement**

Implement a program to calculate the factorial of a given number.

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Fact {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Input a number: ");

int num = scanner.nextInt();

scanner.close();

int factorial = 1;

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

factorial \*= i;

}

System.***out***.println("Factorial of " + num + " = " + factorial);

}

}

**Problem Statement**

Implement a program to display the geometric sequence as given below for a given value n, where n is the number of elements in the sequence.

1, 2, 4, 8, 16, 32, 64, ......, 1024

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Sequence {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Enter the value of n: ");

int n = scanner.nextInt();

scanner.close();

int firstTerm = 1;

int commonRatio = 2;

System.***out***.print("Geometric Sequence: ");

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

int term = firstTerm \* (int) Math.*pow*(commonRatio, i - 1);

System.***out***.print(term + " ");

}

}

}

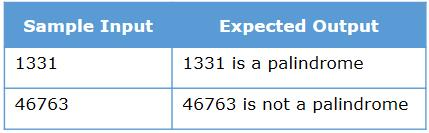
**Problem Statement**

Implement a program to check whether a given number is a palindrome.

Palindrome is a sequence that reads the same backwards as forwards.

E.g.: 121, 1331, 2332, 78900987, 123456654321,  etc.

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Palindrome {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int num = scanner.nextInt();

scanner.close();

int reversedNum = 0;

int originalNum = num;

while (num != 0) {

int digit = num % 10;

reversedNum = reversedNum \* 10 + digit;

num /= 10;

}

if (originalNum == reversedNum) {

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

} else {

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

}

}

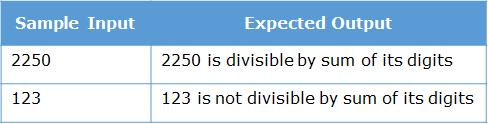
}

**Problem Statement**

Implement a program to find out whether a number is divisible by the sum of its digits.

Display appropriate messages.

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class Divisiblesumdigit

{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int num = scanner.nextInt();

scanner.close();

int sumOfDigits = 0;

int originalNum = num;

while (num != 0) {

int digit = num % 10;

sumOfDigits += digit;

num /= 10;

}

if (originalNum % sumOfDigits == 0) {

System.***out***.println(originalNum + " is divisible by the sum of its digits.");

} else {

System.***out***.println(originalNum + " is not divisible by the sum of its digits.");

}

}

}

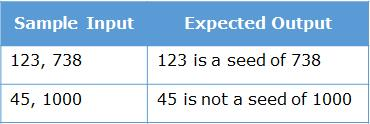
**Problem Statement**

Implement a program to find out whether a number is a seed of another number.

A number X is said to be a seed of number Y if multiplying X by its every digit equates to Y.

E.g.: 123 is a seed of 738 as 123\*1\*2\*3 = 738

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class SeedNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int x = scanner.nextInt();

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

int y = scanner.nextInt();

scanner.close();

int tempX = x;

int product = 1;

while (tempX != 0) {

int digit = tempX % 10;

product \*= digit;

tempX /= 10;

}

if (product == y) {

System.***out***.println(x + " is a seed of " + y);

} else {

System.***out***.println(x + " is not a seed of " + y);

}

}

}

**Problem Statement**

Implement a program to check whether a given number is an Armstrong number.

An Armstrong number is an n-digit number that is equal to the sum of the nth powers of its individual digits.

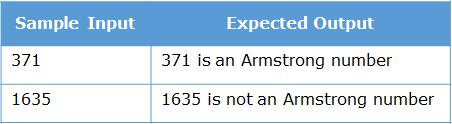
E.g.: 371 is an Armstrong number as 33 + 73 + 13=371

       1634 is an Armstrong number as 14 + 64 + 34+ 44=1634

Hint

Use Math.pow(double a, double b) method to calculate the power of a number

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class ArmstrongNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int num = scanner.nextInt();

scanner.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;

}

if (sum == originalNum) {

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

} else {

System.***out***.println(originalNum + " is not an Armstrong number");

}

}

}

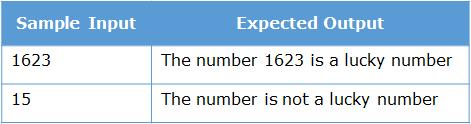
**Problem Statement**

Implement a program to check whether a given number is a lucky number.

A lucky number is a number whose sum of squares of every even-positioned digit (starting from the second position) is a multiple of 9.

E.g. - 1623 = 62+32 = 45 is a multiple of 9 and hence is a lucky number.

**Sample Input and Output**



**Solution:-**

package capstoneproject;

import java.util.Scanner;

public class LuckyNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

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

int num = scanner.nextInt();

scanner.close();

int sum = 0;

int position = 1;

while (num != 0) {

int digit = num % 10;

if (position % 2 == 0) {

sum += digit \* digit;

}

num /= 10;

position++;

}

if (sum % 9 == 0) {

System.***out***.println("The number is a lucky number");

} else {

System.***out***.println("The number is not a lucky number");

}

}

}