Assignment-2

1. Write a program to display first n natural numbers, where n is to be input from the user.
2. display even natural numbers from 2 to n, where n is to be input from the user. For example, if the value of n is 13, the output should be 2 4 6 8 10 12.
3. Write a program to find the sum of first n natural numbers, where n is to be input from the user.
4. To find the sum of first n even natural numbers, where n is to be input from the user.
5. To input an integer and find its factorial. Factorial of an integer n (n>=1), is defined as the product of all natural numbers from 1 to n, and is represented by n!. For example, factorial of 5 is represented by 5! and is equal to 120 (=1x2x3x4x5).

Factorial of 0 is defined as 1 and factorial of negative integers is not defined.

1. script to input 10 numbers and then display the largest of the numbers entered.
2. Write a program to input 10 numbers and then display their sum and average. Also display the largest and the smallest of the numbers entered.
3. input a number and display its first 10 multiples.
4. Write a program to input two numbers ***m*** and ***n***. Then display first ***m*** multiples of ***n***.
5. Write a program to input a number. If the number is negative, then input the number again. Keep on doing so until the user enters a positive number or zero.
6. script to input 10 numbers and then display the smallest number entered. If the smallest number is an integer, then find the sum of its digits, otherwise display the message “smallest number is not an integer”.
7. Write a program to input two numbers and find their LCM and HCF.
8. Write a program to input a number and check whether it is a prime or not.
9. Write a program to display all the prime numbers between ***m*** and ***n***, where ***m*** and ***n*** have to be input from the user.
10. Write a program to input a list of n number and count how many of the entered numbers were prime.
11. input an integer and find the sum of its digits.
12. input an integer and find the product of its odd digits. If the number does not contain any odd digit, then the program should display an appropriate message instead of the product.
13. input 10 integers and find the sum of the digits of each integer.
14. Armstrong number is an integer which is equal to the sum of the cubes of its digits. For example 153 is an Armstrong number because 153 = 13+53+33.

Write script to input an integer and check whether it is an Armstrong number or not.

1. Write a program to input a number and check whether it is palindrome or not.
2. Write a program to input a number and check whether it is a perfect number or not. (A number is a perfect number if it is equal to the sum of its own factors, except itself. For example, 6 is a perfect number as 6=1+2+3, and 8 is not a perfect number as 8 != 1+2+4
3. Write script to display first n terms of Fibonacci series. The Fibonacci series is: 0 1 1 2 3 5 8 13 . . .

First and Second terms of the Fibonacci series are 0 and 1 resepectively, and after that any term is the sum of its previous two terms.

1. Write a program to input a number and check whether it is a Fibonacci number or not. A number is a Fibonacci number if it is a term in the Fibonacci series.
2. Write a menu driven program to calculate the total surface area and volume of a cube, cuboid, or sphere depending upon user’s choice. The program should continue until the user selects the option to exit the program.
3. Write a program to find the sum of first ***n*** terms of the following series without using inbuilt function pow() and math.factorial():

(i) x + x2 + x3 + . . .

(ii) x - x2 + x3 - . . .

*x*2

1. 1+ x +

# 2

*x*2

1. 1+ x +

# 2!

*x*2

*x*3

+ + . . .

# 3

*x*3

+ + . . .

# 3!

*x*3

1. 1- x +

-

# 2! 3!

+ . . .

*x*3

1. 1+ +

# 3!

*x*5 *x*7

+

# 5! 7!

. . .

*x*3

1. 1- +

# 3!

*x*5 *x*7

-

# 5! 7!

. . .

1. Write a program to generate n lines of the following pattern on the computer screen:
   1. 1 12 123

.

.

* 1. 1

12

123

.

.

* 1. 1

121

12321

1234321

.

.

* 1. \*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

### .

**.**

* 1. \*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

.

.

1. Write a program to generate 2n+1 lines of the following pattern on the computer screen:
   1. \*

\*\*\*

\*\*\*\*\*

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

\*\*\*\*\*

\*\*\*

\*

* 1. \* @@@

\*\*\*\*\*

@@@@@@@

\*\*\*\*\* @@@

\*

* 1. \*

\* \*

\* \*

\* \*

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

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

\*\*\*\*\*

\*\*\*

\*