LAB CYCLE - 3

Experiment No :1

Date :14/11/2024

Aim :

To find the factorial of a number.

Pseudocode :

```
PROMPT "Enter a number:" AND STORE input IN num
SET factorial TO 1

FOR i FROM 1 TO num DO
        SET factorial TO factorial * I

PRINT "Factorial of", num, "is", factorial
```

Souce Code :

```
num=int(input("Enter a number:"))
f=1
for i in range(1,num+1):
   f=f*i;
print(f"Factorial of {num} is {f}")
```

Output :

```
Enter a Number: 5
Factorial of 5 is 120
```

Result : The program is successfully executed and the output is verified.

Experiment No :2

Date :14/11/2024

Aim :

Generate Fibonacci series of N terms.

Pseudocode :

```
SET a to 0, b to 1
GET num from user

IF num > 0:
    PRINT "Fibonacci series:"
    IF num >= 1:
        PRINT a
    IF num >= 2:
        PRINT b
    FOR i FROM 3 TO num:
        SET c to a + b
        SET a to b
        SET b to c
        PRINT c
```

Souce Code :

```
a,b =0,1
num = int(input("Enter the limit: "))
if num>0:
   print("Fibonacci series:")
   if num >= 1:
      print(a)
   if num >= 2:
      print(b)
   for i in range(3,num+1):
      c = a + b
      a = b
      b = c
      print(c)
```

Output :

```
Enter the limit: 5
Fibonacci series:
0
1
1
2
3
```

Result : The program is successfully executed and the output is verified.

Experiment No :3

Date :14/11/2024

Aim :

To find the sum of all items in a list.

Pseudocode :

```
GET size from user
SET list=[]
SET sum to 0

FOR each i in size:
    GET values from user
    APPEND values TO list
    ADD values to sum

PRINT "Sum is:", sum
```

Souce Code :

```
size=int(input("Enter the size of list:"))
list=[]
sum=0
for i in range(size):
        val=int(input("Enter a value: "))
        list.append(val)
        sum=sum+val
        print("Sum: ",sum)
```

Output :

```
Enter the size of list:4
Enter a value: 12
Sum: 12
Enter a value: 13
Sum: 25
Enter a value: 25
Sum: 50
Enter a value: 50
Sum: 100
```

Result : The program is successfully executed and the output is verified.

Experiment No :4

Date :14/11/2024

Aim :

Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

Pseudocode :

```
DEFINE function EVEN_PERFECT_SQUARES(start, end):
    CREATE empty list results
    SET start_root to square root of start (integer part)
    SET end_root to square root of end (integer part)

    FOR root FROM start_root TO end_root:
        SET num to root * root
        IF num is between 1000 and 9999 AND num is even:
            SET digits to string of num
            IF all digits of num are even:
                ADD num to results
    RETURN results

GET start_range from user
GET end_range from user
SET result to EVEN_PERFECT_SQUARES(start_range, end_range)

IF result is not empty:
    PRINT "Even perfect squares:", result
ELSE:
    PRINT "No perfect squares found in the specified range."
```

Souce Code :

```
def even_perfect_squares(start, end):
    results = []
    start_root = int(start**0.5)
    end_root = int(end**0.5)
    for root in range(start_root, end_root + 1):
        num = root * root
        if 1000 <= num <= 9999 and num % 2 == 0:
            digits = str(num)
            if all(int(digit) % 2 == 0 for digit in digits):
                results.append(num)
    return results
```

```
start_range = int(input("Enter start range: "))
end_range = int(input("Enter end range: "))
result = even_perfect_squares(start_range, end_range)

if result:
    print(f"Even perfect squares: {result}")
else:
    print("No perfect squares found in the specified range.")
```

Output :

```
Enter the start range: 1000
Enter the end range: 9999
Even perfect squares: [4624, 6084, 6400, 8464]
```

Result : The program is successfully executed and the output is verified.

Experiment No :5

Date :14/11/2024

Aim :

Write a program using a for loop to print the multiplication table of n, where n is entered by the user.

Pseudocode :

```
GET n from user

PRINT "Multiplication table of", n

FOR i FROM 1 TO 11:
    PRINT n, "*", i, "=", n * i
```

Souce Code :

```
n=int(input("Enter the number for multiplication table:"))
print(f"multiplication table of {n} is :")
for i in range (1,11):
    print(f"{n} * {i}= {n*i}")
```

Output :

```
Enter the number for multiplication table:2
multiplication table of 2 is:
5 * 1= 5
5 * 2= 10
5 * 3= 15
5 * 4= 20
5 * 5= 25
5 * 6 =30
5 * 7= 35
5 * 8 =40
5 * 9= 45
5 * 10= 50
```

Result : The program is successfully executed and the output is verified.

Experiment No :6

Date :14/11/2024

Aim :

Write a program to display alternate prime numbers till N (obtain N from the user).

Pseudocode :

```
DEFINE function PR(n):
    SET count to 0
    FOR i FROM 1 TO n:
        IF n % i == 0:
            INCREMENT count by 1
    IF count > 2:
        RETURN 0
    ELSE:
        RETURN 1

GET n from user
CREATE empty list ls
CREATE empty list pls

FOR lim FROM 2 TO n:
    ADD lim to ls

FOR lim in ls:
    IF PR(lim) == 1:
        ADD lim to pls

PRINT every second element of pls
```

Souce Code :

```
def pr(n):
    count=0
    for i in range (1,n+1):
        if n%i==0:
            count=count+1
    if count>2:
        return 0
    else:
        return 1
```

```
n=int(input("Enter the limit:"))
ls=[]
pls=[]
for lim in range(2,n+1):
    ls.append(lim)
for lim in ls:
    if pr(lim)==1:
        pls.append(lim)
print(pls[::2])
```

Output :

```
Enter the limit:5
[2, 5]
```

Result : The Program is successfully executed and the output is verified.

Experiment No :7

Date :14/11/2024

Aim :

Write a program to compute and display the sum of all integers that are divisible by 6 but not by 4, and that lie below a user-given upper limit.

Pseudocode :

```
GET n from user
SET sum to 0

FOR i FROM 1 TO n-1:
    IF i is divisible by 6 AND not divisible by 4:
        ADD i to sum

PRINT "sum is:", sum
```

Souce Code :

```
n=int(input("Enter the limit:"))
sum=0
for i in range (1,n):
    if i%6==0 and i%4!=0:
        sum=sum+i
print(f"sum is: {sum}")
```

Output :

```
Enter the limit: 12
Sum is: 6
```

Result : The program is successfully executed and the output is verified.

Experiment No :8

Date :14/11/2024

Aim :

Calculate the sum of the digits of each number within a specified range (from 1 to a user-defined upper limit). Print the sum only if it is prime.

Pseudocode :

```
IMPORT math library

DEFINE function SUM_OF_DIGITS(n):
    RETURN sum of digits of n

DEFINE function IS_PRIME(n):
    IF n <= 1:
        RETURN False
    IF n == 2:
        RETURN True
    IF n is even:
        RETURN False
    FOR i FROM 3 TO square root of n:
        IF n is divisible by i:
            RETURN False
    RETURN True

DEFINE function SUM_DIGITS_IN_RANGE(upper_limit):
    FOR num FROM 1 TO upper_limit:
        SET digit_sum to SUM_OF_DIGITS(num)
        IF digit_sum is prime:
            PRINT "Sum of digits of", num, "is", digit_sum, "which is prime."

GET limit from user
CALL SUM_DIGITS_IN_RANGE(limit)
```

Souce Code :

```
import math
def sum_of_digits(n):
   return sum(int(digit) for digit in str(n))
def is_prime(n):
   if n <= 1:
      return False
```

```
    if n == 2:
        return True
    if n % 2 == 0:
        return False
    for i in range(3, int(math.sqrt(n)) + 1, 2):
        if n % i == 0:
            return False
    return True
def sum_digits_in_range(upper_limit):
    for num in range(1, upper_limit + 1):
        digit_sum = sum_of_digits(num)
        if is_prime(digit_sum):
            print(f"Sum of digits of {num} is {digit_sum}, which is prime.")
limit = int(input("Enter an upper limit: "))
sum_digits_in_range(limit)
```

Output :

```
Enter an upper limit: 5
Sum of digits of 2 is 2, which is prime.
Sum of digits of 3 is 3, which is prime.
Sum of digits of 5 is 5, which is prime.
```

Result : The program is successfully executed and the output is verified.

Experiment No :9

Date :14/11/2024

Aim :

A number is input through the keyboard. Write a program to determine if it's palindromic.

Pseudocode :

```
GET n from user

IF n is equal to n reversed:
    PRINT "Is palindrome"
ELSE:
    PRINT "Is not palindrome"
```

Souce Code :

```
n=input("Enter the number to be checked:")
if n==n[::-1]:
    print("Is palindrome")
else:
    print("Is not palindrome")
```

Output :

```
Enter the number to be checked: 121
121 Is palindrome

Enter the number to be checked: 122
122 Is not palindrome
```

Result : The program is successfully executed and the output is verified.

Experiment No :10

Date :14/11/2024

Aim :

Write a program to generate all factors of a number.

Pseudocode :

```
GET n from user
CREATE empty list fact

FOR i FROM 1 TO n:
    IF n is divisible by i:
        ADD i to fact

PRINT "Factors of", n, "is", fact
```

Souce Code :

```
n=int(input("Enter the number:"))
fact=[]
for i in range(1,n+1):
    if n%i==0:
        fact.append(i)
print(f"factors of {n} is {fact}")
```

Output :

```
Enter the number: 12
factors of 12 is [1, 2, 3, 4, 6, 12]
```

Result : The program is successfully executed and the output is verified.

Experiment No :11

Date :14/11/2024

Aim :

Write a program to find whether the given number is an Armstrong number or not.

Pseudocode :

```
GET number from user
SET original_num to number
SET sum_of_cubes to 0

WHILE number > 0:
    SET digit to number % 10
    ADD digit^3 to sum_of_cubes
    SET number to number // 10

IF sum_of_cubes is equal to original_num:
    PRINT original_num, "is an Armstrong number."
ELSE:
    PRINT original_num, "is not an Armstrong number."
```

Souce Code :

```
number=int(input("Enter a number: "))
original_num=number
sum_of_cubes = 0
while number > 0:
   digit = number % 10
   sum_of_cubes += digit ** 3
   number //= 10
if sum_of_cubes == original_num:
   print(f"{original_num} is an Armstrong number.")
else:
   print(f"{original_num} is not an Armstrong number.")
```

Output :

```
Enter a number: 153
153 is an Armstrong number.
```

```
Enter a number: 370
370 is an Armstrong number.
```

Result : The program is successfully executed and the output is verified.

Experiment No :12

Date :14/11/2024

Aim :

Display the given pyramid with the step number accepted from the user.
1
2 4
3 6 9
4 8 12 16

Pseudocode :

```
GET n from user

FOR i FROM 1 TO n:
      FOR j FROM 1 TO i:
            PRINT i * j, WITHOUT newline
      PRINT a newline
```

Souce Code :

```
n=int(input("Enter the number of steps for the pyramid: "))
for i in range(1,n+1):
   for j in range(1,i+1):
      print(i*j, end=' ')
   print()
```

Output :

```
Enter the number of steps for the pyramid: 4
1
2 4
3 6 9
4 8 12 16
```

Result : The program is successfully executed and the output is verified.

Experiment No :13

Date :14/11/2024

Aim :

Construct following pattern using nested loop

```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
*  *  *  *
*  *  *
*  *
*
```

Pseudocode :

```
FOR i FROM 1 TO 5:
    FOR j FROM 1 TO i:
        PRINT "*", WITHOUT newline
    PRINT a newline

FOR i FROM 4 DOWN TO 1:
    FOR j FROM 1 TO i:
        PRINT "*", WITHOUT newline
    PRINT a newline
```

Souce Code :

```
for i in range(1, 6):
   for j in range(i):
   print("*",end=" ")
   print( )
for i in range(4,0,-1):
   for j in range(i):
   print("*",end=" ")
   print( )
```

Output :

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

Result : The program is successfully executed and the output is verified.