Programming with Python Jan 27, 28 2025

Note: Brief Summary of contents discussed.

One line expression for if-else

<expression 1> if <condition> else <expression 2>

Example:
temp = eval(input("Enter temperature in Celsius : "))
msg = 'Warm Day' if temp > 24 == 0 else 'Not so warm'

For loop

General form or for

```
for variable in sequence:
    Statements
```

range: generates a sequence of integers

```
range(n)
range(start, end)
range(start, end, increment)
```

Exercise:

1. Write a program to find the sum of first n numbers. (Pay attention to the significance of collecting sum in a variable initialized to 0).

2. Write a program to find the factorial of number n. (Pay attention to the significance of accumulating multiplication in a variable initialized to 1).

3. Input a number from the user. Find sum of digits.

4. Input a number from the user. Reverse the digits of the number.

5. Write a program and compute the sum of even digits and sum of odd digits.
6. Write a program to count the number of odd and even digits in a number.

7. Write a program to find the square root of a number num iteratively. Given the initial guess: root0. Iterating 5 times using for loop to

   Root at nth iteration = 0.5 * (root + num / (root at n-1 th iteration))

8. WAP that prints Armstrong numbers in the range 1 to 1000. An Armstrong number is a number whose sum of the cubes of the digits is equal to the number itself.

   For example: 370 = 3**3 + 7**3 + 0**3

   `[Note: In the lab, the number was taken from the user. Here, the check has to be performed for all numbers from 1 till 1000]`

9. Write a program to print the following patterns

   a.

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

   Write the above program

   a. Using string repetition
   b. Using string concatenation.
   c. Use multiple `for` loops

10.

(i).

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

(ii).

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

(iii).

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

(iv).

```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

(v).

```
5 4 3 2 1
5 4 3 2
5 4 3
5 4
5
```

(vi)

```
5 4 3 2 1
4 3 2 1
3 2 1
3 2
1
```

(vii)

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