Python-For Loop

September 19, 2023

[1]:
```
#Write a Python program to print the numbers from 1 to 10 using a `for` loop.
for i in range(1, 11):
    print(i)
```

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

[2]:
```
lst = []
num = int(input('How many numbers: '))
for n in range(num):
    numbers = int(input('Enter number '))
    lst.append(numbers)
print("Sum of elements in given list is :", sum(lst))
```

```
How many numbers:  3
Enter number  12
Enter number  23
Enter number  45

Sum of elements in given list is : 80
```

[4]:
```
def reverse_string(str):
    str1 = ""   # Declaring empty string to store the reversed string
    for i in str:
        str1 = i + str1
    return str1    # It will return the reverse string to the caller function

str = "Abhishek"    # Given String
print("The original string is: ",str)
print("The reverse string is",reverse_string(str))
```

```
The original string is:  Abhishek
The reverse string is kehsihbA
```

[6]:
```
#4. Develop a program that finds the factorial of a given number using a `for` loop.
num = int(input("enter a number: "))

fac = 1

for i in range(1, num + 1):
 fac = fac * i

print("factorial of ", num, " is ", fac)
```

```
enter a number:  4

factorial of  4  is  24
```

[7]:
```
#Create a program to print the multiplication table of a given number using a `for` loop.
number = int(input ("Enter the number of which the user wants to print the multiplication table: "))
# We are using "for loop" to iterate the multiplication 10 times
print ("The Multiplication Table of: ", number)
for count in range(1, 11):
   print (number, 'x', count, '=', number * count)
```

```
Enter the number of which the user wants to print the multiplication table:  6

The Multiplication Table of:  6
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
```

[8]:
```
#Write a program that counts the number of even and odd numbers in a list using a `for` loop.
# list of numbers
list1 = [10, 21, 4, 45, 66, 93, 1]

even_count, odd_count = 0, 0
```

```
# iterating each number in list
for num in list1:

    # checking condition
    if num % 2 == 0:
        even_count += 1

    else:
        odd_count += 1

print("Even numbers in the list: ", even_count)
print("Odd numbers in the list: ", odd_count)
```

```
Even numbers in the list:  3
Odd numbers in the list:  4
```

[9]:
```
#Develop a program that prints the squares of numbers from 1 to 5 using a `for`␣
 ↪loop.
nums = [1, 2, 3, 4, 5,]
for i in nums:
    print(i)

squared = [ ]

for i in nums:
    sqr = i * i
    squared.append(sqr)
    print("The square of {} is {}".format(i, sqr))
```

```
1
2
3
4
5
The square of 1 is 1
The square of 2 is 4
The square of 3 is 9
The square of 4 is 16
The square of 5 is 25
```

[10]:
```
#Create a program to find the length of a string without using the `len()`␣
 ↪function.
string = 'Hello'

count = 0

for i in string:
```

```
    count+=1
print(count)
```

```
5
```

[11]:
```
#Write a program that calculates the average of a list of numbers using a `for` loop.
n=int(input("Enter the number of elements to be inserted: "))
a=[]
for i in range(0,n):
    elem=int(input("Enter element: "))
    a.append(elem)
avg=sum(a)/n
print("Average of elements in the list",round(avg,2))
```

```
Enter the number of elements to be inserted:  6
Enter element:  1
Enter element:  2
Enter element:  3
Enter element:  4
Enter element:  5
Enter element:  6
Average of elements in the list 3.5
```

[13]:
```
#Develop a program that prints the first `n` Fibonacci numbers using a `for` loop.
n=int(input("Enter the number of terms: "))
a=0
b=1
if n<=0:
    print("The Output of your input is",a)
else:
    print(a,b,end=" ")
    for x in range(2,n):
        c=a+b
        print(c,end=" ")
        a=b
        b=c
```

```
Enter the number of terms:  9
0 1 1 2 3 5 8 13 21
```

[14]:
```
# Write a program to check if a given list contains any duplicates using a `for` loop.
def Repeat(x):
```

```
    _size = len(x)
    repeated = []
    for i in range(_size):
        k = i + 1
        for j in range(k, _size):
            if x[i] == x[j] and x[i] not in repeated:
                repeated.append(x[i])
    return repeated

# Driver Code
list1 = [10, 20, 30, 20, 20, 30, 40,
         50, -20, 60, 60, -20, -20]
print (Repeat(list1))
```

```
[20, 30, -20, 60]
```

[15]:
```
#Create a program that prints the prime numbers in a given range using a `for` loop.
r=int(input("Enter upper limit: "))
for a in range(2,r+1):
    k=0
    for i in range(2,a//2+1):
        if(a%i==0):
            k=k+1
    if(k<=0):
        print(a)
```

```
Enter upper limit:  35
2
3
5
7
11
13
17
19
23
29
31
```

[18]:
```
#Develop a program that counts the number of vowels in a string using a `for` loop.
string = "GeekforGeeks!"
vowels = "aeiouAEIOU"

count = sum(string.count(vowel) for vowel in vowels)
print(count)
```

5

[5]:
```
#Write a program to find the maximum element in a 2D list using a nested `for`␣
 ↪loop.
l = [2,4,6,[5,10,3]]


def flatten(seq):
  for el in seq:
    if isinstance(el, list):
      yield from flatten(el)
    else:
      yield el

print(max(flatten(l)))
```

10

[6]:
```
#Create a program that removes all occurrences of a specific element from a␣
 ↪list using a `for` loop.
# list with integer elements
list = [10, 20, 10, 30, 10, 40, 10, 50]
# number (n) to be removed
n = 10

# print original list
print ("Original list:")
print (list)

# loop to traverse each element in list
# and, remove elements
# which are equals to n
for x in list:
    if x == n:
        list.remove(x)

# print list after removing given element
print ("list after removing elements:")
print (list)
```

```
Original list:
[10, 20, 10, 30, 10, 40, 10, 50]
list after removing elements:
[20, 30, 40, 50]
```

[9]:
```
#Develop a program that generates a multiplication table for numbers from 1 to␣
 ↪5 using a nested `for` loop.
```

```
import sys
for i in range(1,6):
   for j in range(1,11):
      k = i*j
      print (k, end=' ')
   print()
```

```
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
```

[10]:
```
#Write a program that converts a list of Fahrenheit temperatures to Celsius␣
 ↪using a `for` loop.
fah = [89.8, 67.0, 92, 99]

cen = []
for f in fah:
    c = (f - 32) * 5 / 9
    cen.append(c)

print(cen)
```

```
[32.111111111111114, 19.444444444444443, 33.333333333333336, 37.22222222222222]
```

[11]:
```
#18. Create a program to print the common elements from two lists using a `for`␣
 ↪loop.
def common_member(a, b):
    a_set = set(a)
    b_set = set(b)

    if (a_set & b_set):
        print(a_set & b_set)
    else:
        print("No common elements")


a = [1, 2, 3, 4, 5]
b = [5, 6, 7, 8, 9]
common_member(a, b)

a = [1, 2, 3, 4, 5]
b = [6, 7, 8, 9]
common_member(a, b)
```

```
{5}
No common elements
```

[13]:
```
#Develop a program that prints the pattern of right-angled triangles using a␣
 ↪`for` loop. Use '*' to draw the
#pattern
x=int(input("Enter row number=\n"))
for i in range(x):
    for j in range(i+1):
        print("*",end='')
    print("")
```

```
Enter row number=
 3
*
**
***
```

[14]:
```
#Write a program to find the greatest common divisor (GCD) of two numbers using␣
 ↪a `for` loop.
import math

# prints 12
print("The gcd of 60 and 48 is : ", end="")
print(math.gcd(60, 48))
```

```
The gcd of 60 and 48 is : 12
```

[16]:
```
#Create a program that calculates the sum of the digits of numbers in a list␣
 ↪using a list comprehension.
# Initializing list
numlist = [12,34,56,78,90]
res = [sum(map(int, str(x))) for x in numlist]
print(res)
```

```
[3, 7, 11, 15, 9]
```

[20]:
```
#Write a program to find the prime factors of a given number using a `for` loop␣
 ↪and list comprehension.
import math
[x for x in range(2, 21) if  all(x % y != 0 for y in range(2, int(math.sqrt(x +␣
 ↪1)) ) )]
```

[20]:
```
[2, 3, 4, 5, 6, 7, 9, 11, 13, 17, 19]
```

[26]:
```
#Develop a program that extracts unique elements from a list and stores them in␣
 ↪a new list using a list
#comprehension.
# initializing list
test_list = [1, 3, 5, 6, 3, 5, 6, 1]
```

```
print("The original list is : "
      + str(test_list))

# using list comprehension to remove duplicated from list
res = []
[res.append(x) for x in test_list if x not in res]

# printing list after removal
print ("The list after removing duplicates : "
       + str(res))
```

```
The original list is : [1, 3, 5, 6, 3, 5, 6, 1]
The list after removing duplicates : [1, 3, 5, 6]
```

[27]:
```
#Create a program that generates a list of all palindromic numbers up to a␣
 ↪specified limit using a list
#comprehension.
def createPalindrome(inp, b, isOdd):
    n = inp
    palin = inp

    # checks if number of digits is odd or even
    # if odd then neglect the last digit of input in
    # finding reverse as in case of odd number of
    # digits middle element occur once
    if (isOdd):
        n = n // b

    # Creates palindrome by just appending reverse
    # of number to itself
    while (n > 0):
        palin = palin * b + (n % b)
        n = n // b
    return palin

# Function to print decimal palindromic number
def generatePalindromes(n):

    # Run two times for odd and even length palindromes
    for j in range(2):
        # Creates palindrome numbers with first half as i.
        # Value of j decided whether we need an odd length
        # of even length palindrome.
        i = 1
        while (createPalindrome(i, 10, j % 2) < n):
            print (createPalindrome(i, 10, j % 2),end=" ")
            i = i + 1
```

```
# Driver Program to test above function
n = 104
generatePalindromes(n)
```

```
11 22 33 44 55 66 77 88 99 1 2 3 4 5 6 7 8 9 101
```

[28]:
```
#Write a program to flatten a nested list using list comprehension.
# Input list
nested_lists = [[3, 4, 5],[ 7, 8, 9, 10]]

#flatten the lists
flat_list = [y for x in nested_lists for y in x]
print("Flatten List: ",flat_list)
```

```
Flatten List:  [3, 4, 5, 7, 8, 9, 10]
```

[32]:
```
#Develop a program that computes the sum of even and odd numbers in a list␣
 ↪separately using list
#comprehension.
list1=[3, 4, 5, 7, 8, 9, 10]
sum([x for x in list1 if x % 2 != 0])
```

[32]:
```
22
```

[36]:
```
#Develop a program that computes the sum of even and odd numbers in a list␣
 ↪separately using list
#comprehension.
list1=[3, 4, 5, 7, 8, 9, 10]
sum([x for x in list1 if x % 2 == 0])
```

[36]:
```
22
```

[38]:
```
#Create a program that generates a list of squares of odd numbers between 1 and␣
 ↪10 using list
#comprehension.
([x*x for x in range(10) if x % 2 != 0])
```

[38]:
```
[1, 9, 25, 49, 81]
```

[39]:
```
#Write a program that combines two lists into a dictionary using list␣
 ↪comprehension.
index = [1, 2, 3]
languages = ['python', 'c', 'c++']

dictionary = {k: v for k, v in zip(index, languages)}
print(dictionary)
```

```
{1: 'python', 2: 'c', 3: 'c++'}
```

[40]:
```
#Develop a program that extracts the vowels from a string and stores them in a␣
 ↪list using list comprehension.
text = input('Enter text: ')
count = 0
vowel_list = [char for char in text if char.lower() in "aeiou"]
print(len(vowel_list))
print(vowel_list)
```

```
Enter text:  abhi
2
['a', 'i']
```

[41]:
```
#Create a program that removes all non-numeric characters from a list of␣
 ↪strings using list comprehension.
s="1-2$3%4 5a"
s1="".join(c for c in s if  c.isdecimal())
print (s1)
#Output:12345
```

```
12345
```

[42]:
```
#Write a program to generate a list of prime numbers using the Sieve of␣
 ↪Eratosthenes algorithm and list
#comprehension.
def SieveOfEratosthenes(num):
    prime = [True for i in range(num+1)]
# boolean array
    p = 2
    while (p * p <= num):

        # If prime[p] is not
        # changed, then it is a prime
        if (prime[p] == True):

            # Updating all multiples of p
            for i in range(p * p, num+1, p):
                prime[i] = False
        p += 1

    # Print all prime numbers
    for p in range(2, num+1):
        if prime[p]:
            print(p)
```

```
if __name__ == '__main__':
    num = 30
    print("Following are the prime numbers smaller"),
    print("than or equal to", num)
    SieveOfEratosthenes(num)
```

```
Following are the prime numbers smaller
than or equal to 30
2
3
5
7
11
13
17
19
23
29
```

[43]:
```
#32. Create a program that generates a list of all Pythagorean triplets up to a␣
 ↪specified limit using list
#comprehension.
n = int(input('Enter the value of n: '))
print([(a, b, c) for a in range(1, n + 1) for b in range(a, n + 1)
       for c in range(b, n + 1) if a**2 + b**2 == c**2])
```

```
Enter the value of n:  10
[(3, 4, 5), (6, 8, 10)]
```

[44]:
```
#Develop a program that generates a list of all possible combinations of two␣
 ↪lists using list comprehension.

# initializing lists
list1 = [1, 3, 4]
list2 = [6, 7, 9]


# printing lists
print ("The original lists are : " + str(list1) +
                            " " + str(list2))


# using list comprehension
# to compute all possible permutations
res = [[i, j, k] for i in list1
                 for j in list2
```

```
                ]

# printing result
print ("All possible permutations are : " + str(res))
```

```
The original lists are : [1, 3, 4] [6, 7, 9]
All possible permutations are : [[1, 6, 50], [1, 7, 50], [1, 9, 50], [3, 6, 50],
[3, 7, 50], [3, 9, 50], [4, 6, 50], [4, 7, 50], [4, 9, 50]]
```

[46]:
```
#Write a program that calculates the mean, median, and mode of a list of␣
 ↪numbers using list
#comprehension.
def my_mean(sample):
 return sum(sample) / len(sample)
my_mean([4, 8, 6, 5, 3, 2, 8, 9, 2, 5])
```

[46]: 5.2

[47]:
```
def my_median(sample):
     n = len(sample)
     index = n // 2
     # Sample with an odd number of observations
     if n % 2:
        return sorted(sample)[index]
    # Sample with an even number of observations
     return sum(sorted(sample)[index - 1:index + 1]) / 2


my_median([3, 5, 1, 4, 2])
```

[47]: 3

[63]:
```
#mode
list1 = [12, 16, 20, 20, 12, 30, 25, 23, 24, 20]
frequency = {}
for i in list1:
    frequency.setdefault(i, 0)
    frequency[i]+=1

frequent = max(frequency.values())
for i, j in frequency.items():
    if j == frequent:
        mode = i
print(mode)
```

```
20
```

[64]:
```
#Create a program that generates Pascal's triangle up to a specified number of␣
 ↪rows using list
#comprehension.
num = int(input("Enter the number: "))
list1 = [] #an empty list
for i in range(num):
  list1.append([])
  list1[i].append(1)
  for j in range(1, i):
    list1[i].append(list1[i - 1][j - 1] + list1[i - 1][j])
  if(num != 0):
    list1[i].append(1)
for i in range(num):
  print(" " * (num - i), end = " ", sep = " ")
  for j in range(0, i + 1):
    print('{0:6}'.format(list1[i][j]), end = " ", sep = " ")
  print()
```

```
Enter the number:  4
          1
        1      1
      1      2      1
    1      3      3      1
```

[65]:
```
#Write a program that finds the longest word in a sentence using list␣
 ↪comprehension.
# Longest word

# Reading sentence from user

sentence = input("Enter sentence: ")

# Finding longest word
longest = max(sentence.split(), key=len)

# Displaying longest word
print("Longest word is: ", longest)
print("And its length is: ", len(longest))
```

```
Enter sentence:  Tongue tied and twisted just an earth bound misfit I

Longest word is:  twisted
And its length is:  7
```

[66]:
```
#39. Develop a program that calculates the sum of the digits of numbers from 1␣
 ↪to 1000 using list
#comprehension.
```

```
res = [sum(map(int, str(x))) for x in range(1,1001)]
print(res)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 2, 3, 4, 5, 6, 7, 8,
9, 10, 11, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 5,
6, 7, 8, 9, 10, 11, 12, 13, 14, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 7, 8, 9, 10,
11, 12, 13, 14, 15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11, 12, 13,
14, 15, 16, 17, 18, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 2, 3, 4, 5, 6, 7, 8, 9, 10,
11, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 5, 6, 7,
8, 9, 10, 11, 12, 13, 14, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 7, 8, 9, 10, 11,
12, 13, 14, 15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11, 12, 13, 14,
15, 16, 17, 18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 2, 3, 4, 5, 6, 7, 8, 9,
10, 11, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 5, 6,
7, 8, 9, 10, 11, 12, 13, 14, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 7, 8, 9, 10,
11, 12, 13, 14, 15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11, 12, 13,
14, 15, 16, 17, 18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 11, 12, 13, 14, 15,
16, 17, 18, 19, 20, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 4, 5, 6, 7, 8, 9, 10, 11,
12, 13, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9,
10, 11, 12, 13, 14, 15, 16, 17, 18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 11,
12, 13, 14, 15, 16, 17, 18, 19, 20, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 4,
5, 6, 7, 8, 9, 10, 11, 12, 13, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 6, 7, 8, 9,
10, 11, 12, 13, 14, 15, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 10, 11, 12, 13, 14,
15, 16, 17, 18, 19, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 12, 13, 14, 15, 16,
17, 18, 19, 20, 21, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 5, 6, 7, 8, 9, 10,
11, 12, 13, 14, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 7, 8, 9, 10, 11, 12, 13, 14,
15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11, 12, 13, 14, 15, 16, 17,
18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 11, 12, 13, 14, 15, 16, 17, 18, 19,
20, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9,
10, 11, 12, 13, 14, 15, 16, 17, 18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 11,
12, 13, 14, 15, 16, 17, 18, 19, 20, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 13,
14, 15, 16, 17, 18, 19, 20, 21, 22, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 8, 9,
10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 10, 11,
12, 13, 14, 15, 16, 17, 18, 19, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 12, 13,
14, 15, 16, 17, 18, 19, 20, 21, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 14, 15,
16, 17, 18, 19, 20, 21, 22, 23, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 16, 17,
18, 19, 20, 21, 22, 23, 24, 25, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 9, 10, 11,
12, 13, 14, 15, 16, 17, 18, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 11, 12, 13,
14, 15, 16, 17, 18, 19, 20, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 13, 14, 15,
16, 17, 18, 19, 20, 21, 22, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 15, 16, 17,
18, 19, 20, 21, 22, 23, 24, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 17, 18, 19,
20, 21, 22, 23, 24, 25, 26, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 10, 11, 12,
13, 14, 15, 16, 17, 18, 19, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 12, 13, 14,
```

```
15, 16, 17, 18, 19, 20, 21, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 14, 15, 16,
17, 18, 19, 20, 21, 22, 23, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 16, 17, 18,
19, 20, 21, 22, 23, 24, 25, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 1]
```

[72]:
```
#Write a program that generates a list of prime palindromic numbers using list␣
↪comprehension.
[x for x in range(5,20) if not any([x % y == 0 for y in range(2, int(x/2)+1)])]
```

[72]:
```
[5, 7, 11, 13, 17, 19]
```

[ ]:
```

```