for loop assignment

October 26, 2023

[1]:

```
#1
for i in range(1,11):
    print(i)
```

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

[3]:

```
#2
numbers=(1,2,3,4,5,6,7,8)
total=0
for i in numbers:
    total +=i
print("sum of all numbers :",total)
```

```
sum of all numbers : 36
```

[5]:

```
#3
word="hello world"
for char in range(len(word)-1,-1,-1):
    print(word[char],end='')
print('\n')
```

```
dlrow olleh
```

[1]:

```
#4
number=5
factorial=1
for i in range(1,number +1):
    factorial *=i
```

```python
print("factorial of number :",factorial)
```

```
factorial of number : 120
```

[2]:

```python
#5
n=int(input("enter a number"))
for i in range(1,11):
    print(n,'x',i,'=',n*i)
```

```
enter a number 2
```

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
```

[4]:

```python
#6
count_even=0
count_odd=0
for i in range(1,10):
    if i%2==0:
        count_even +=1
    elif i%2!=0:
        count_odd +=1
print("total even numbers are:",count_even)
print("total odd numbers are:",count_odd)
```

```
total even numbers are: 4
total odd numbers are: 5
```

[5]:

```python
#7
for i in range(1,6):
    square=i**2
    print("the square of i is:",square)
```

```
the square of i is: 1
the square of i is: 4
the square of i is: 9
the square of i is: 16
the square of i is: 25
```

[6]:

```
#8
word="hello world"
length=0
for char in word:
    length=length+1
print(length)
```

```
11
```

[9]:

```
#9
numbers=[1,2,3,4,5,6,7,8]
sum_of_numbers=0
for numb in numbers:
    sum_of_numbers +=1
average=sum_of_numbers/len(numbers)
print("average:",average)
```

```
average: 1.0
```

[12]:

```
#10
n=10
a=0
b=1
for i in range(n):
    print(a)
    c=a+b
    a=b
    b=c
```

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

[13]:

```
#11
l=[1,2,3,4,5,4,3,2,2,1]
l1=[]
l2=[]
for i in l:
    if i not in l1:
        l1.append(i)
    elif i not in l2:
        l2.append(i)
print(l2)
```

```
[4, 3, 2, 1]
```

[24]:

```
#12
start=9
end=60
for num in range(start,end+1):
    if num >1:
        is_prime=True
        for i in range(2,num):
            if (num %i)==0:
                is_prime=False
                break
        if is_prime:
            print(i)
```

```
10
12
16
18
22
28
30
36
40
42
46
52
58
```

[37]:

```
#13
word="pwskills data science pro"
vowels="aeiouAEIOU"
count=sum(word.count(vowel)for vowel in vowels)
print(count)
```

```
7
```

[38]:

```
#14
def find_max_2d(lst):
    max_element = float('-inf')
    for row in lst:
        for element in row:
            if element > max_element:
                max_element = element
    return max_element

# Example usage:
matrix = [
    [3, 7, 1],
    [9, 4, 8],
    [2, 5, 6]
]

max_value = find_max_2d(matrix)
print("The maximum element is:", max_value)
```

```
The maximum element is: 9
```

[39]:

```
#15
def remove_element(lst, target):
    for item in lst[:]:
        if item == target:
            lst.remove(item)
my_list = [1, 2, 2, 3, 4, 2, 5]
element_to_remove = 2

remove_element(my_list, element_to_remove)
print("List after removing", element_to_remove, ":", my_list)
```

```
List after removing 2 : [1, 3, 4, 5]
```

[40]:

```
#16
for i in range(1, 6):
    for j in range(1, 11):
        print(i, "*", j, "=", i * j)
```

```
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
1 * 4 = 4
1 * 5 = 5
1 * 6 = 6
1 * 7 = 7
1 * 8 = 8
1 * 9 = 9
1 * 10 = 10
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
```

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

[42]:

```python
#17
def fahrenheit_to_celsius(fahrenheit_temperatures):
    celsius_temperatures = []
    for fahrenheit in fahrenheit_temperatures:
        celsius = (fahrenheit - 32) * 5/9
        celsius_temperatures.append(celsius)
    return celsius_temperatures
fahrenheit_temperatures = [32]
celsius_temperatures = fahrenheit_to_celsius(fahrenheit_temperatures)
print("Celsius Temperatures:", celsius_temperatures)
```

```
Celsius Temperatures: [0.0]
```

[43]:

```
#18
def common_elements(list1, list2):
    common = []
    for item1 in list1:
        for item2 in list2:
            if item1 == item2:
                common.append(item1)
    return common
list1 = [1, 2, 3, 4, 5]
list2 = [3, 4, 5, 6, 7]
common_elements_list = common_elements(list1, list2)
print("Common elements:", common_elements_list)
```

```
Common elements: [3, 4, 5]
```

[44]:

```
#19
n = 5
for i in range(1, n + 1):
    for j in range(1, i + 1):
        print("*", end=" ")
    print()
```

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

[45]:

```
#20
def find_gcd(a, b):
    gcd = 1
    for i in range(1, min(a, b) + 1):
        if a % i == 0 and b % i == 0:
            gcd = i
    return gcd
num1 = 48
num2 = 60
gcd = find_gcd(num1, num2)
print("GCD of", num1, "and", num2, "is", gcd)
```

```
GCD of 48 and 60 is 12
```

[46]:

```
#21
def sum_of_digits(num):
    total = 0
    for digit in str(num):
        total += int(digit)
    return total
```

```
numbers = [123, 45, 678]
digit_sums = [sum_of_digits(num) for num in numbers]
print("Sum of digits for each number:", digit_sums)
```

```
Sum of digits for each number: [6, 9, 21]
```

[47]:

```
#22
def prime_factors(n):
    factors = []
    for factor in range(2, n + 1):
        if n % factor == 0:
            prime = True
            for d in range(2, factor):
                if factor % d == 0:
                    prime = False
                    break
            if prime:
                factors.append(factor)
    return factors

number = 36
factors = prime_factors(number)
print("Prime factors of", number, "are:", factors)
```

```
Prime factors of 36 are: [2, 3]
```

[48]:

```
#23
l = [1, 2, 2, 3, 4, 4, 5, 5]
l1= []
for element in l:
    if element not in l1:
        l1.append(element)
print("Unique elements:", l1)
```

```
Unique elements: [1, 2, 3, 4, 5]
```

[49]:

```
#24
limit = 100
palindromic_numbers = []
for num in range(1, limit + 1):
    if str(num) == str(num)[::-1]:
        palindromic_numbers.append(num)
print("Palindromic numbers up to", limit, "are:", palindromic_numbers)
```

```
Palindromic numbers up to 100 are: [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 22, 33, 44,
55, 66, 77, 88, 99]
```

[50]:
```
#25
nested_list = [[1, 2, 3], [4, 5], [6, 7, 8]]
flattened_list = []
for sublist in nested_list:
    for item in sublist:
        flattened_list.append(item)
print("Flattened list:", flattened_list)
```

```
Flattened list: [1, 2, 3, 4, 5, 6, 7, 8]
```

[51]:
```
#26
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]
even_sum = 0
odd_sum = 0
for num in numbers:
    if num % 2 == 0:
        even_sum += num
    else:
        odd_sum += num
print("Sum of even numbers:", even_sum)
print("Sum of odd numbers:", odd_sum)
```

```
Sum of even numbers: 20
Sum of odd numbers: 25
```

[52]:
```
#27
keys = ['a', 'b', 'c']
values = [1, 2, 3]
combined_dict = {keys[i]: values[i] for i in range(len(keys))}
print("Combined dictionary:", combined_dict)
```

```
Combined dictionary: {'a': 1, 'b': 2, 'c': 3}
```

[53]:
```
#28
text = "Hello, world!"
vowels = [char for char in text if char.lower() in 'aeiou']
print("Vowels in the text:", vowels)
```

```
Vowels in the text: ['e', 'o', 'o']
```

[54]:
```
#29
strings = ["abc123", "def456", "ghi789", "jklmno"]
numeric_strings = [''.join(char for char in s if char.isdigit()) for s in␣
 ↪strings]
print("Numeric strings:", numeric_strings)
```

```
Numeric strings: ['123', '456', '789', '']
```

[56]:

```
#31
def sieve_of_eratosthenes(limit):
    sieve = [True] * (limit + 1)
    sieve[0] = sieve[1] = False

    for current in range(2, int(limit**0.5) + 1):
        if sieve[current]:
            for multiple in range(current*current, limit + 1, current):
                sieve[multiple] = False

    return [x for x in range(2, limit + 1) if sieve[x]]
limit = int(input("Enter the limit: "))
prime_list = sieve_of_eratosthenes(limit)
print("Prime numbers up to", limit, "are:", prime_list)
```

```
Enter the limit:  100

Prime numbers up to 100 are: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41,
43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]
```

[57]:

```
#32
def generate_pythagorean_triplets(limit):
    triplets = [(a, b, c) for a in range(1, limit) for b in range(a, limit) for␣
 ↪c in range(b, limit) if a**2 + b**2 == c**2]
    return triplets
limit = int(input("Enter the limit: "))
pythagorean_triplets = generate_pythagorean_triplets(limit)
print("Pythagorean triplets up to", limit, "are:", pythagorean_triplets)
```

```
Enter the limit:  100

Pythagorean triplets up to 100 are: [(3, 4, 5), (5, 12, 13), (6, 8, 10), (7, 24,
25), (8, 15, 17), (9, 12, 15), (9, 40, 41), (10, 24, 26), (11, 60, 61), (12, 16,
20), (12, 35, 37), (13, 84, 85), (14, 48, 50), (15, 20, 25), (15, 36, 39), (16,
30, 34), (16, 63, 65), (18, 24, 30), (18, 80, 82), (20, 21, 29), (20, 48, 52),
(21, 28, 35), (21, 72, 75), (24, 32, 40), (24, 45, 51), (24, 70, 74), (25, 60,
65), (27, 36, 45), (28, 45, 53), (30, 40, 50), (30, 72, 78), (32, 60, 68), (33,
44, 55), (33, 56, 65), (35, 84, 91), (36, 48, 60), (36, 77, 85), (39, 52, 65),
(39, 80, 89), (40, 42, 58), (40, 75, 85), (42, 56, 70), (45, 60, 75), (48, 55,
73), (48, 64, 80), (51, 68, 85), (54, 72, 90), (57, 76, 95), (60, 63, 87), (65,
72, 97)]
```

[58]:

```
#33
list1 = [1, 2, 3]
list2 = ['a', 'b', 'c']

combinations = [(x, y) for x in list1 for y in list2]
print(combinations)
```

```
[(1, 'a'), (1, 'b'), (1, 'c'), (2, 'a'), (2, 'b'), (2, 'c'), (3, 'a'), (3, 'b'),
(3, 'c')]
```

[59]:

```python
#34
import statistics

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

mean = sum(numbers) / len(numbers)
median = statistics.median(numbers)
mode = statistics.mode(numbers)

print("Mean:", mean)
print("Median:", median)
print("Mode:", mode)
```

```
Mean: 5.5
Median: 5.5
Mode: 1
```

[61]:

```python
#35
def generate_pascals_triangle(n):
    triangle = [[1]]
    for i in range(1, n):
        row = [1] + [triangle[i-1][j] + triangle[i-1][j+1] for j in range(i-1)]␣
  ↪+ [1]
        triangle.append(row)
    return triangle

# Example usage:
n = int(input("Enter the number of rows for Pascal's triangle: "))
pascals_triangle = generate_pascals_triangle(n)
for row in pascals_triangle:
    print(row)
```

```
Enter the number of rows for Pascal's triangle:  10

[1]
[1, 1]
[1, 2, 1]
[1, 3, 3, 1]
[1, 4, 6, 4, 1]
[1, 5, 10, 10, 5, 1]
[1, 6, 15, 20, 15, 6, 1]
[1, 7, 21, 35, 35, 21, 7, 1]
[1, 8, 28, 56, 70, 56, 28, 8, 1]
[1, 9, 36, 84, 126, 126, 84, 36, 9, 1]
```

[65]:

```
#37
sentence = "This is a sample sentence with some long words."
words = sentence.split()
longest_word = max(words, key=lambda word: len(word))
print("Longest word:", longest_word)
```

```
Longest word: sentence
```

[66]:

```
#38
strings = ["hello", "world", "programming", "python", "list", "comprehension"]

filtered_strings = [s for s in strings if sum(1 for char in s if char in␣
  ↪"aeiouAEIOU") > 3]
print(filtered_strings)
```

```
['comprehension']
```

[68]:

```
#39
digit_sums = [sum(map(int, str(n))) for n in range(1, 1001)]
print(digit_sums)
```

```
[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]
```

[69]:

```python
#40
def is_prime(n):
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0 or n % 3 == 0:
        return False
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False
        i += 6
    return True

def is_palindrome(n):
    return str(n) == str(n)[::-1]

limit = int(input("Enter the limit:"))
prime_palindromes = [n for n in range(2, limit) if is_prime(n) and␣
 ↪is_palindrome(n)]
print(prime_palindromes)
```

```
Enter the limit: 121
```

```
[2, 3, 5, 7, 11, 101]
```

```
[ ]:
```