

Loop Statements

Dr. Dongchul Kim

for **Loop**

for loop

A for loop iterates over a given sequence (e.g. list). Here is an example:

```
myList = ["apple", "banana", "cherry", "kiwi", "lime"]  
  
for x in myList:  
    print(x)
```

```
1 myList = ["apple", "banana", "cherry", "kiwi", "lime", "mango"]
2 for x in myList:
3     print(x)
4
```

```
▶ ↑ /home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjects,
■ ↓ apple
☰ banana
🔍 cherry
🔍 kiwi
🗑️ lime
🗑️ mango

Process finished with exit code 0
```

Lab 9-1 ~ 9-9

- Submission
 - Capture the output of your program.
 - Upload both the captured image files and Python files on Blackboard.
 - Create a separate file for each lab; do NOT combine multiple labs into a single file.

Lab 9-1

Define a list of five different car brands (e.g. Ford, Toyota)

Display the five elements of the list using a for loop.

Lab 9-1

List Definition (20 Points):

Successfully defining a list containing five different car brands (20 Points)

Incomplete or incorrect list definition (0 Points)

Using a For Loop (60 Points):

Correctly implementing a for loop to iterate over the list (60 Points)

Incorrect or incomplete for loop implementation (0 Points)

Display of List Elements (20 Points):

Accurately displaying each element of the list during the loop iteration (20 Points)

Incomplete or incorrect display of list elements (0 Points)

Total Points: 100

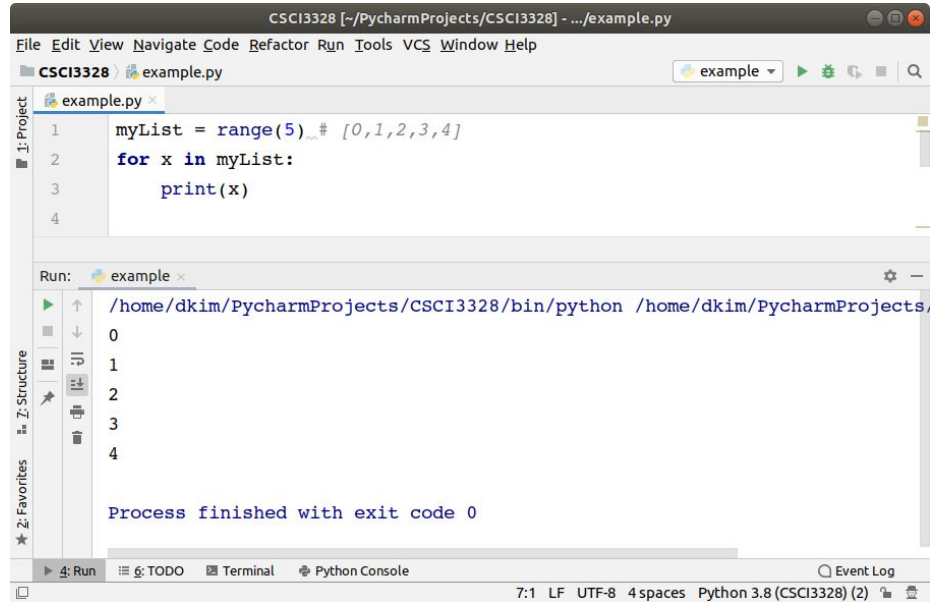
range ()

Instead of using a list of strings, you can utilize a list of integer numbers.

The `range (int)` function generates a sequence of numbers within the specified range.

It's important to note that the `range ()` function is zero-based.

While `range ()` doesn't return a list, you can think of it as a sequence object, which behaves similarly to a list.



The screenshot shows the PyCharm IDE interface. The main editor window displays a Python script named `example.py` with the following code:

```
1 myList = range(5) # [0, 1, 2, 3, 4]
2 for x in myList:
3     print(x)
4
```

Below the editor, the Run console shows the output of the script:

```
Run: example x
/home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjects,
0
1
2
3
4

Process finished with exit code 0
```

The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces, and Python 3.8 (CSCI3328) (2).

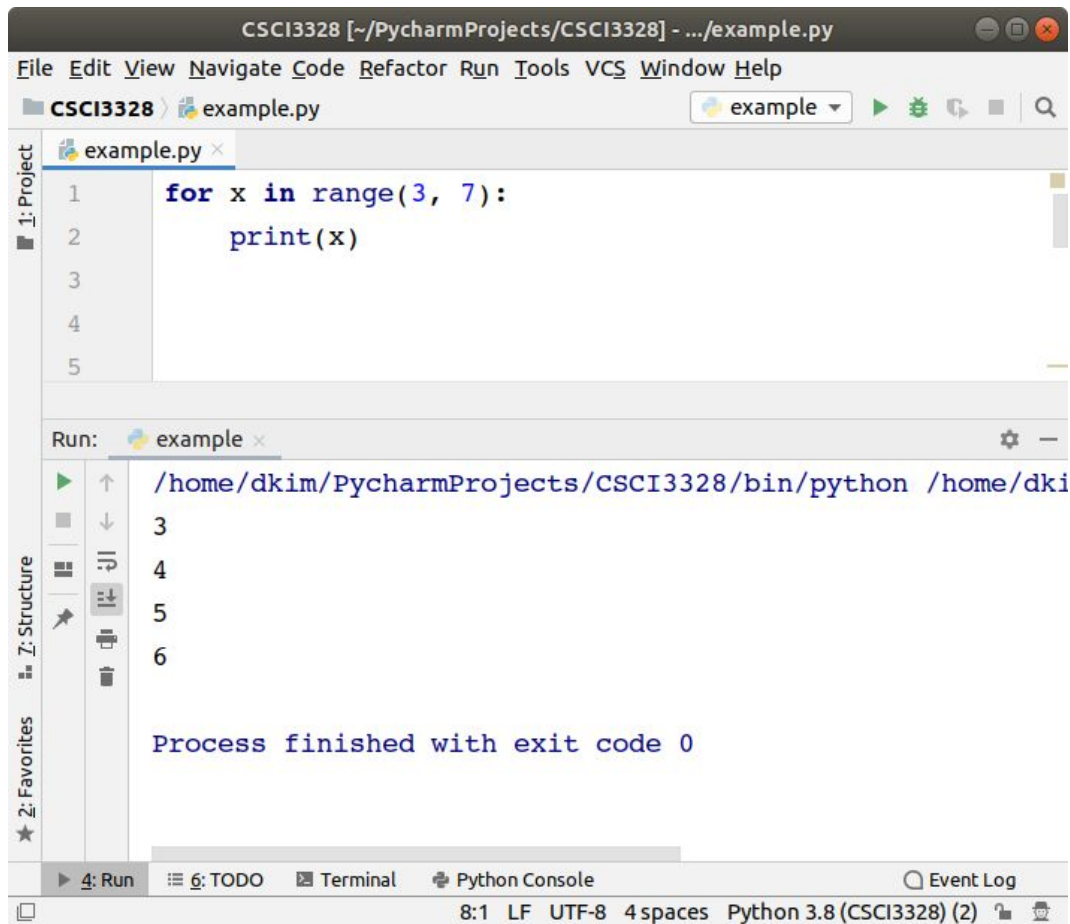

```
range(start, end, step)
```

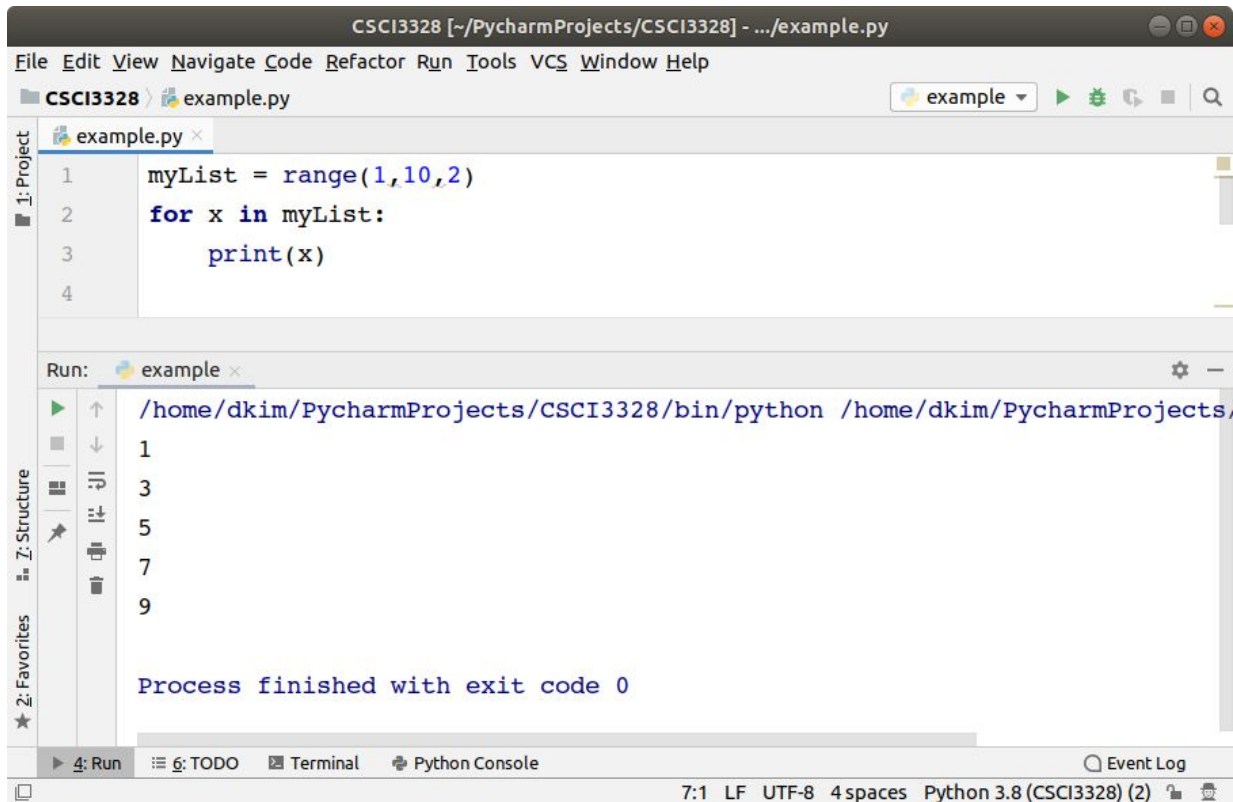
The range() function accepts three arguments, with two of them being optional:

Start Argument (Optional): This argument specifies the starting number of the sequence. If not specified, it defaults to 0.

End Argument: This argument sets an upper limit for generating numbers. The range() function generates numbers up to, but does not include, this specified number in the result.

Step Argument (Optional): The step argument defines the difference between each number in the generated sequence. If not provided, it defaults to 1.





break statement

With the `break` statement we can stop the loop before it has looped through all the items:

Example

Exit the loop when x is "banana":

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
    print(x)
    if x == "banana":
        break
```

continue statement

With the `continue` statement we can stop the current iteration of the loop, and continue with the next:

Example

Do not print banana:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
    if x == "banana":
        continue
    print(x)
```

Lab 9-2

When n is given by user input, make a program to display n asterisks (star character) using a for loop. Print a single star at a time in each iteration of the for loop. Hint: `print("*", end="")`

```
Please input a number? 7(enter)
```

```
*****
```

No hardcoding!

```
print("*****")
```

Lab 9-2

User Input for Number of Asterisks (n) (20 Points):

Correctly prompting for and capturing the user's input for the number of asterisks (n) (20 Points)

Incorrect or missing prompt for number of asterisks (0 Points)

For Loop Implementation (60 Points):

Accurately using a for loop to iterate n times (60 Points)

Incorrect or incomplete for loop implementation (0 Points)

Asterisk Display in Each Iteration (20 Points):

Correctly displaying a single asterisk in each iteration without a newline (using `print("*", end="")`) (20 Points)

Incorrect display of asterisks or incorrect use of print function (0 Points)

Total Points: 100

Nested for loop

Like if statements, loops can be nested.

If a loop is nested, the inner loop will execute all of its iterations for each time the outer loop executes once.

```
for i in range(10):  
    for j in range(10):  
        print(i, j)
```

The `print(i, j)` in this example will execute 100 times.

Lab 9-3

When n is given by a user input, make a program that displays a multiplication table like shown below using a *nested* for loop.

```
Please input a number? 5(enter)
```

```
1 2 3 4 5
2 4 6 8 10
3 6 9 12 15
4 8 12 16 20
5 10 15 20 25
```

Lab 9-3

User Input for Table Size (n) (20 Points):

Correctly prompting for and capturing the user's input for the size of the multiplication table (n) (20 Points)

Incorrect or missing prompt for the table size (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to generate the multiplication table (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Correct Multiplication Table Display (30 Points):

Displaying the multiplication table correctly up to 'n x n' (30 Points)

Incorrect or incomplete display of the multiplication table (0 Points)

Format and Readability (10 Points):

Ensuring the output is well-formatted and easy to read (10 Points)

Poor formatting or readability (0 Points)

Total Points: 100

Lab 9-4

When n is given by a user input, make a program that displays asterisks like shown below using a *nested* for loop and `print("*", end="")`

```
Please input a number? 7(enter)
```

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

No hardcoding!

```
print ("*****")  
print ("*****")  
print ("*****")  
print ("*****")  
print ("*****")  
print ("*****")  
print ("*****")
```

Lab 9-4

User Input for Number of Lines (n) (20 Points):

Correctly prompting for and capturing the user's input for the number of lines (n) (20 Points)

Incorrect or missing prompt for the number of lines (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to iterate through lines and characters (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Asterisk Display for Each Line (30 Points):

Correctly displaying a line of asterisks (n asterisks per line) using `print("*", end="")` for each character and `print()` for each new line (30 Points)

Incorrect display of asterisks or incorrect use of print function (0 Points)

Correct Output Structure (10 Points):

Displaying a square block of asterisks with n lines and n asterisks per line, maintaining the shape as per the input number (10 Points)

Incorrect structure or pattern of the output (0 Points)

Total Points: 100

Lab 9-5

When `n` is given by a user input, make a program that displays asterisks like shown below using a *nested* for loop and `print("*", end="")`

```
Please input a number? 7(enter)
```

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

Lab 9-5

User Input for Pattern Size (n) (20 Points):

Correctly prompting for and capturing the user's input for the size of the asterisk pattern (n) (20 Points)

Incorrect or missing prompt for the pattern size (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to generate the increasing asterisk pattern (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Asterisk Pattern Display (30 Points):

Correctly displaying an increasing number of asterisks in each line, starting with 1 asterisk up to n asterisks, using `print("*", end="")` for each asterisk and `print()` for a new line (30 Points)

Incorrect display of the asterisk pattern (0 Points)

Output Format and Alignment (10 Points):

Ensuring the output is well-formatted, with each line's asterisks aligned to the left and matching the input number in the final line (10 Points)

Poor formatting or alignment of asterisks (0 Points)

Total Points: 100

Lab 9-6

When `n` is given by a user input, make a program that displays asterisks like shown below using a *nested* for loop and `print ("*", end="")`

```
Please input a number? 7(enter)

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

Lab 9-6

User Input for Triangle Height (n) (20 Points):

Correctly prompting for and capturing the user's input for the height of the asterisk triangle (n) (20 Points)

Incorrect or missing prompt for the triangle height (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to generate the right-aligned asterisk triangle (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Asterisk Triangle Display (30 Points):

Correctly displaying the triangle with increasing asterisks in each line, right-aligned, using `print("*", end="")` for asterisks and `print()` for a new line (30 Points)

Incorrect display of the asterisk triangle or alignment issues (0 Points)

Correct Output Structure and Spacing (10 Points):

Ensuring the output is well-structured with appropriate spacing to the left of the asterisks, maintaining the triangle shape as per the input number (10 Points)

Poor structure, spacing, or misalignment of the triangle (0 Points)

Total Points: 100

Lab 9-7

User Input for Triangle Height (n) (20 Points):

Correctly prompting for and capturing the user's input for the height of the inverted asterisk triangle (n) (20 Points)

Incorrect or missing prompt for the triangle height (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to generate the right-aligned inverted asterisk triangle (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Asterisk Triangle Display (30 Points):

Correctly displaying the triangle with decreasing asterisks in each line, right-aligned, using `print(" ", end=" ")` for asterisks and `print()` for a new line (30 Points)

Incorrect display of the asterisk triangle or alignment issues (0 Points)

Correct Output Structure and Spacing (10 Points):

Ensuring the output is well-structured with appropriate spacing to the left of the asterisks, maintaining the inverted triangle shape as per the input number (10 Points)

Poor structure, spacing, or misalignment of the triangle (0 Points)

Total Points: 100

Lab 9-8

When `n` is given by a user input, make a program that displays asterisks like shown below using a *nested* for loop and `print ("*", end="")`

```
Please input a number? 5(enter)
```

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

Lab 9-8

User Input for Pyramid Height (n) (20 Points):

Correctly prompting for and capturing the user's input for the height of the asterisk pyramid (n) (20 Points)

Incorrect or missing prompt for the pyramid height (0 Points)

Nested For Loop Implementation (40 Points):

Accurately using nested for loops to generate the center-aligned asterisk pyramid (40 Points)

Incorrect or incomplete nested for loop implementation (0 Points)

Asterisk Pyramid Display (30 Points):

Correctly displaying the pyramid with an increasing number of asterisks in each line, centered, using `print("*", end="")` for asterisks and `print()` for a new line (30 Points)

Incorrect display of the asterisk pyramid or center alignment issues (0 Points)

Correct Output Structure and Symmetry (10 Points):

Ensuring the output is symmetric and well-structured, maintaining the pyramid shape as per the input number (10 Points)

Poor structure, asymmetry, or misalignment of the pyramid (0 Points)

Total Points: 100

Lab 9-9

John is visiting a bakery to purchase cookies and needs to determine the shortfall in case his current money isn't sufficient. Each cookie costs K dollars, he intends to buy N cookies, and he currently has M dollars. If he calculates the shortfall (S), his mother will provide the necessary funds to cover it. For example, if the cost of a single cookie is \$3, he wants to purchase 4 cookies, and he has \$10, the amount needed is \$2. In this case, the shortfall (S) is \$2, which his mother will provide.

Here are some additional examples:

- If one cookie costs \$2.50, he plans to buy 2 cookies, and he has \$1.50, his mother needs to pay \$3.50 to cover the shortfall.
- If the price of one cookie is \$0.20, he intends to buy 6 cookies, and he has \$1.50, he doesn't need any additional money from his mother as he has enough to cover the cost.

Your program should calculate the shortfall (S) and determine if John's mother needs to provide extra money to cover the cost of the cookies.

Lab 9-9

Input & Output Format

T = the number of test case

K = the price of one cookie

N = the number of cookie

M = the amount of money John has

S = the amount of money he will ask his mother to pay

(K , N , and M are non-negative)

Input

T

K N M

Output

S

Lab 9-9

Example

Input

3

30.00 4 100.00

2.50 2 1.40

2.00 6 12.00

Output

20.00

3.60

0.00

Lab 9-9

Input Capture for Test Cases (T) (10 Points):

Correctly prompting for and capturing the number of test cases (T) (10 Points)

Incorrect or missing prompt for the number of test cases (0 Points)

Input Capture for Each Test Case (20 Points):

Accurately capturing the price of one cookie (K), the number of cookies (N), and the amount of money John has (M) for each test case (20 Points)

Incorrect or incomplete input capture for any test case (0 Points per case)

Shortfall Calculation Logic (40 Points):

Correctly calculating the shortfall (S) for each test case, considering the total cost and John's current money (40 Points)

Incorrect shortfall calculation (0 Points per case)

Correct Output for Each Test Case (20 Points):

Displaying the calculated shortfall (S) for each test case in the correct format (20 Points)

Incorrect or missing output for any test case (0 Points per case)

Determination of Additional Funds Requirement (10 Points):

Accurately determining whether John needs additional money from his mother for each test case (10 Points)

Incorrect determination of additional funds requirement (0 Points per case)

Total Points: 100

List Comprehension

What is List Comprehension?

Definition: A concise way to create and manipulate lists in Python.

An alternative to loops and map/filter functions for creating lists.

Syntax: `[expression for item in iterable]`

Example: Create a list of squares `[x*x for x in range(10)]`

Advantages

Enhanced Readability and Clarity: Easier to understand and write than loops for simple list operations.

Reduced Code Length: Condenses multiple lines of loop code into a single, readable line.

Improved Performance: Often faster than equivalent code written using loops or map/filter functions.