Conditional Statements

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Let’s explore conditional statements

- Conditional statements are handled by ‘if’ statements in python
- Conditional statement perform computations or actions depending on the boolean constraint is evaluated as true or false
- If the constraint is ‘True’, the statements in the body are executed, and if it is ‘False’, the statements in body are skipped
- Conditional statements:
    - If statement
    - If-else statement
    - If elif else statement
    - Nested if-else

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If Statement

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If statement

- The syntax of the if-statement is

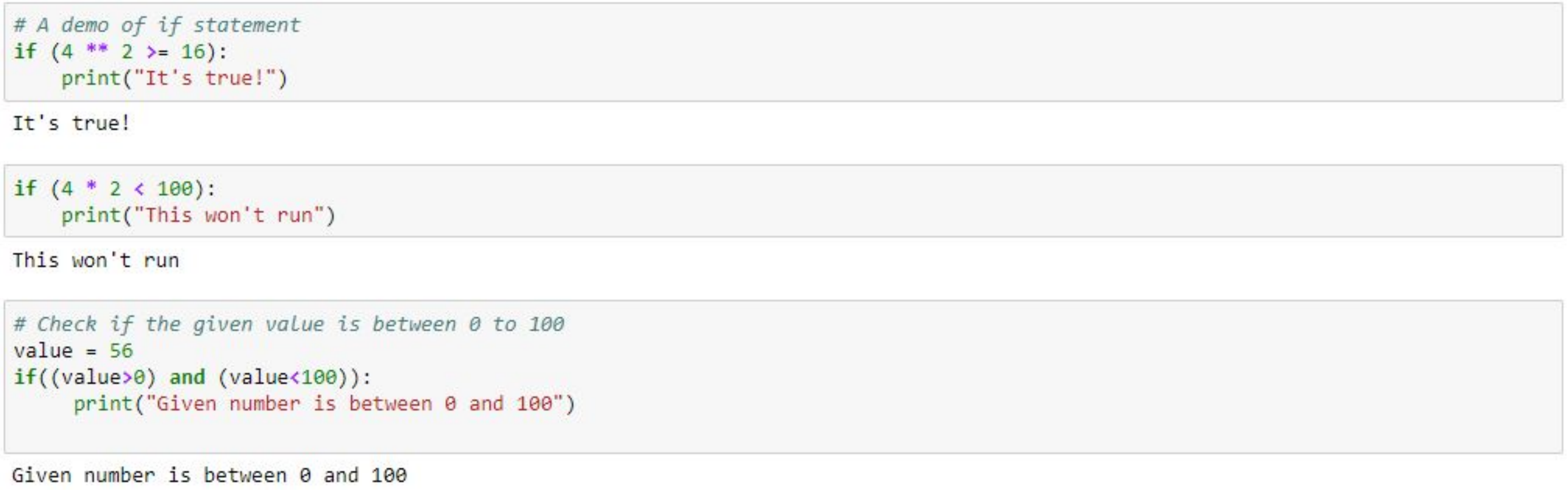
```
if condition:
    statement 1
    statement 2
          ....
```

- Python logical conditions supported are
  - Equivalence `==`
  - Inequivalence `!=`
  - Less than `>`
  - Greater than `<`
  - Less than or equal to `<=`
  - Greater than or equal to `>=`

- If statement can be nested

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If statement



```
# A demo of if statement
if (4 ** 2 >= 16):
    print("It's true!")
```

```
It's true!
```

```
if (4 * 2 < 100):
    print("This won't run")
```

```
This won't run
```

```
# Check if the given value is between 0 to 100
value = 56
if((value>0) and (value<100)):
     print("Given number is between 0 and 100")
```

```
Given number is between 0 and 100
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The if...else Statement

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If-else statement

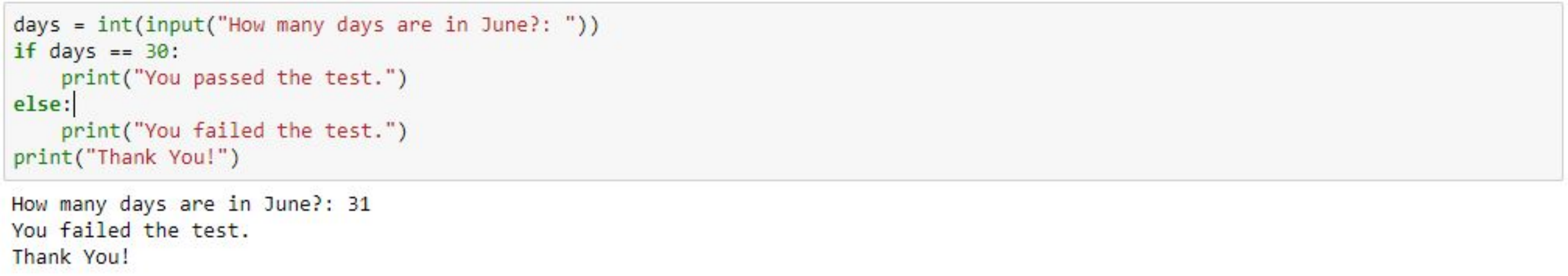
- The syntax of the if-statement is

```
if condition:
    statement(s)
else:
     statement(s)
```

- The statement(s) under ‘else:’ are executed when the condition in ‘if ’ is not satisfies, ie., the boolean output is ‘False’

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If-else statement

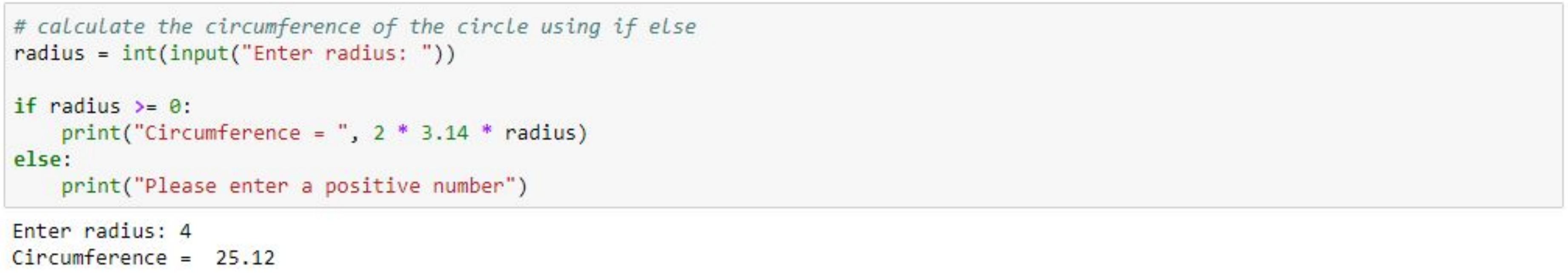


```python
days = int(input("How many days are in June?: "))
if days == 30:
    print("You passed the test.")
else:
    print("You failed the test.")
print("Thank You!")
```

```
How many days are in June?: 31
You failed the test.
Thank You!
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If-else statement



```python
# calculate the circumference of the circle using if else
radius = int(input("Enter radius: "))

if radius >= 0:
    print("Circumference = ", 2 * 3.14 * radius)
else:
    print("Please enter a positive number")
```

```
Enter radius: 4
Circumference =  25.12
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If elif else Statement

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

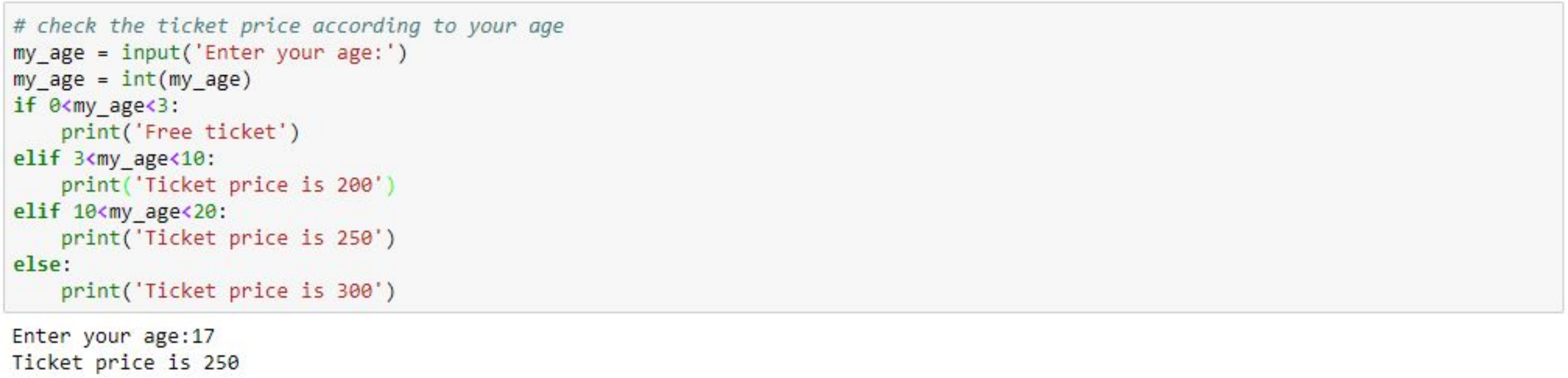
If elif else statement

- Elif statement is used to control multiple conditions only if the given if condition false
- It's similar to an if-else statement and the only exception is that in else we are not checking the condition but in elif, we check the condition
- The syntax of the if elif else is

```
if (condition):
     Stmts
elif (condition):
     Stmts
else:
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If elif else statement

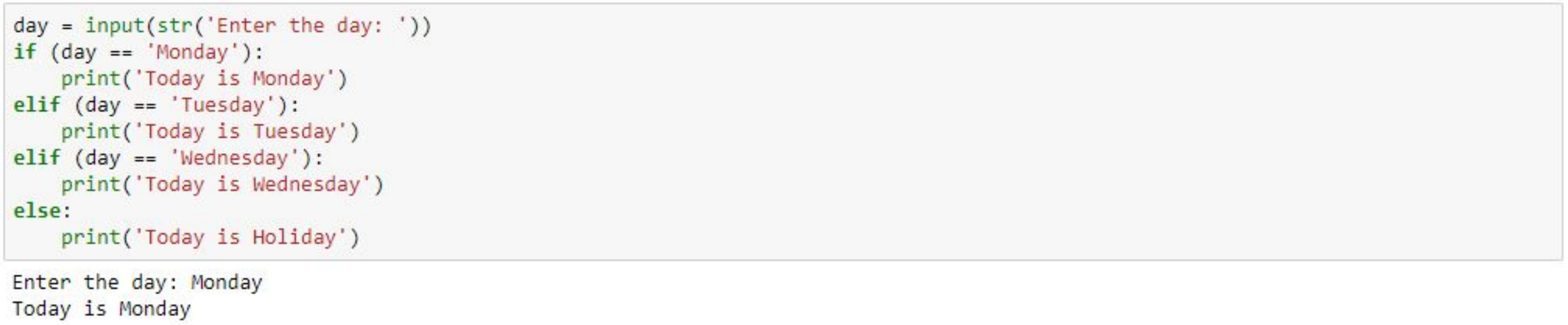


```
# check the ticket price according to your age
my_age = input('Enter your age:')
my_age = int(my_age)
if 0<my_age<3:
    print('Free ticket')
elif 3<my_age<10:
    print('Ticket price is 200')
elif 10<my_age<20:
    print('Ticket price is 250')
else:
    print('Ticket price is 300')
```

```
Enter your age:17
Ticket price is 250
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If elif else statement



```
day = input(str('Enter the day: '))
if (day == 'Monday'):
    print('Today is Monday')
elif (day == 'Tuesday'):
    print('Today is Tuesday')
elif (day == 'Wednesday'):
    print('Today is Wednesday')
else:
    print('Today is Holiday')
```

```
Enter the day: Monday
Today is Monday
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested if and if…else Statement

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested If-else statement

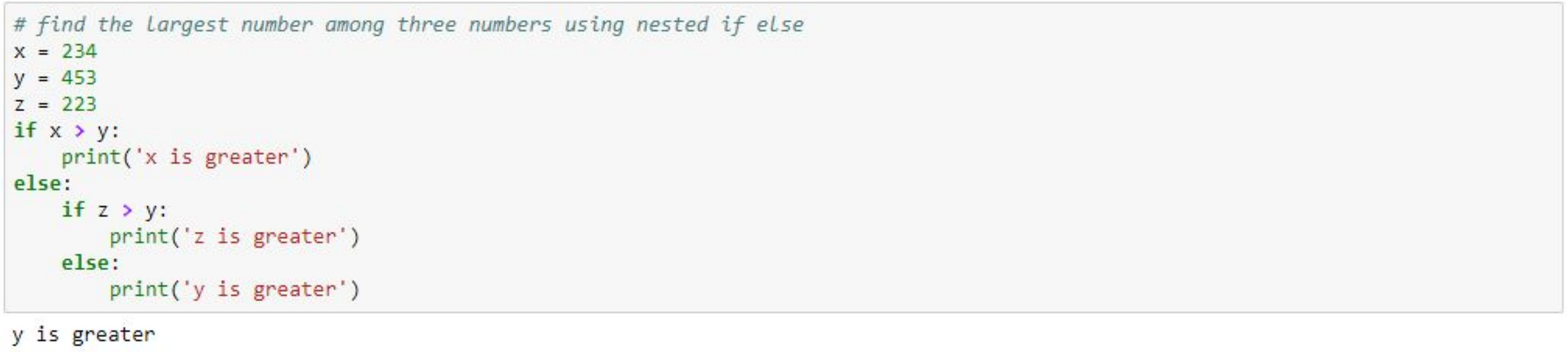
- The syntax of the if-statement is

```
if condition:
  statement(s)
  if condition:
    statement(s)
  else:
    statement(s)
else:
  statement(s)
```

- Note that a ‘if’ statements are within the body of another if statement
- The statements in this manner are called ‘nested if- statements’
- If the first ‘if’ condition is satisfied then the program executes the commands within that ‘if’

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested If-else statement

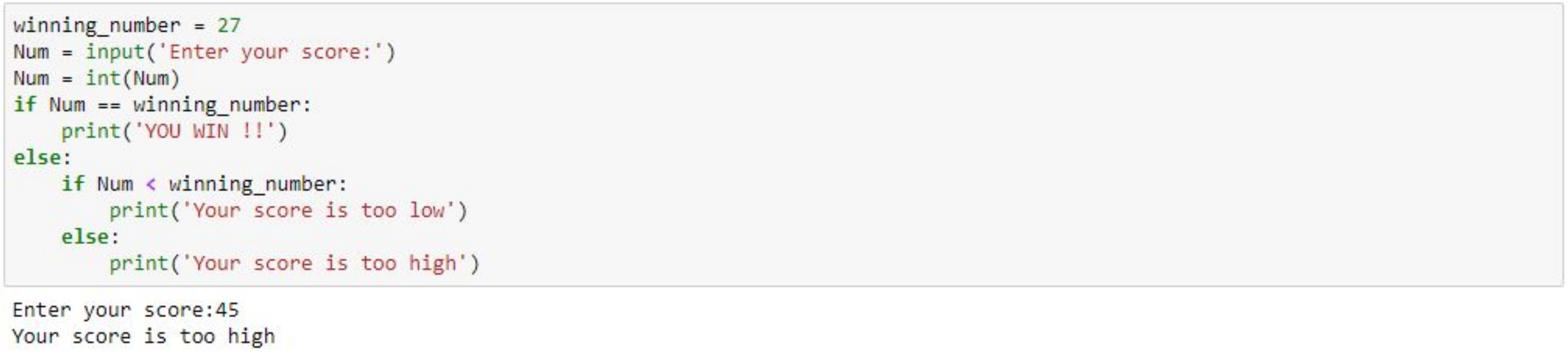


```python
# find the largest number among three numbers using nested if else
x = 234
y = 453
z = 223
if x > y:
    print('x is greater')
else:
    if z > y:
        print('z is greater')
    else:
        print('y is greater')
```

```
y is greater
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested If-else statement

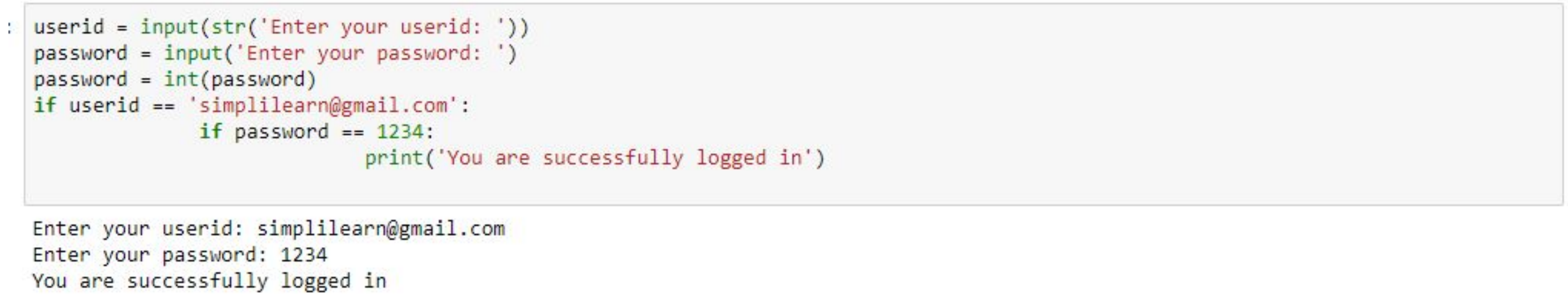


```
winning_number = 27
Num = input('Enter your score:')
Num = int(Num)
if Num == winning_number:
    print('YOU WIN !!')
else:
    if Num < winning_number:
        print('Your score is too low')
    else:
        print('Your score is too high')
```

```
Enter your score:45
Your score is too high
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested If statement



```
userid = input(str('Enter your userid: '))
password = input('Enter your password: ')
password = int(password)
if userid == 'simplilearn@gmail.com':
              if password == 1234:
                            print('You are successfully logged in')
```

```
Enter your userid: simplilearn@gmail.com
Enter your password: 1234
You are successfully logged in
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Loops

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Python Iterables

- Iterable is an object capable of returning its members one by one
- Sequence type objects like lists, strings, tupes, dictionaries and sets are common type of iterables

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Iteration through the Iterables

- Iteration is a general term for taking each item from a sequence one after another

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

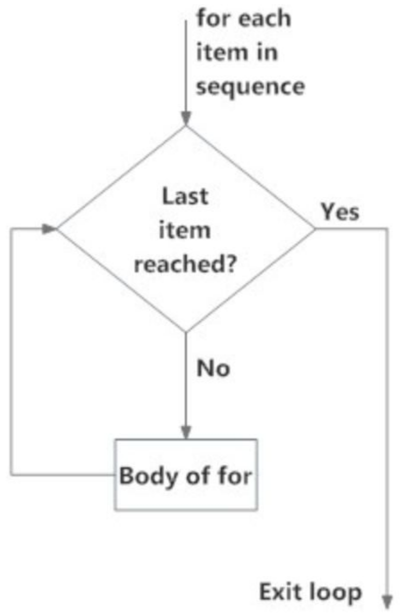
Python’s for loop

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The for loop

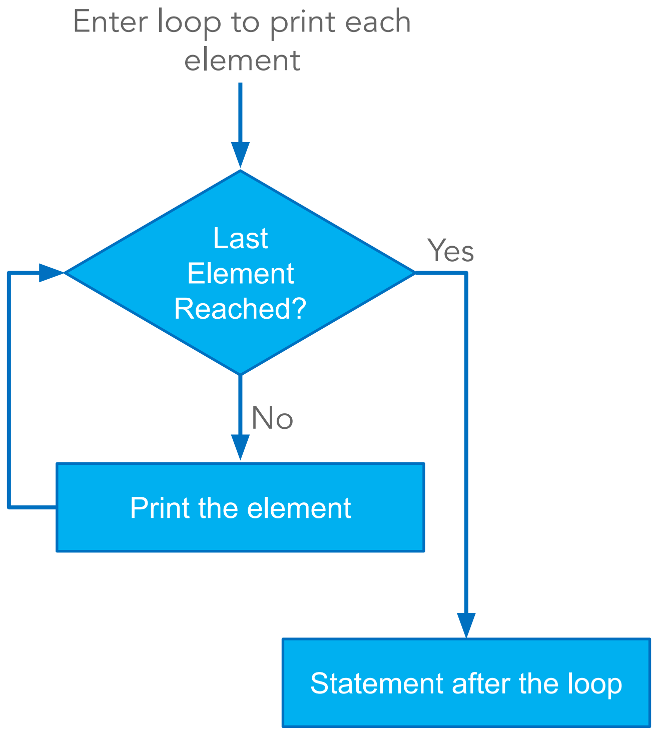
The for loop in Python is used to iterate over a sequence (list, tuple, string) or other iterable objects.

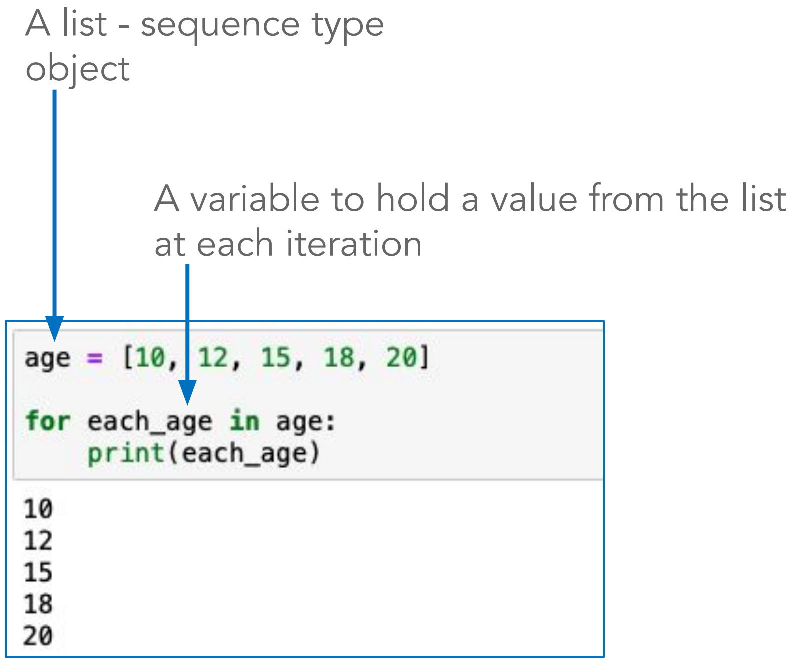
Iterating over a sequence is called traversal.



This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The for loop code example




A list - sequence type object
A variable to hold a value from the list at each iteration

```
age = [10, 12, 15, 18, 20]

for each_age in age:
    print(each_age)
```

```
10
12
15
18
20
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read each element of a string

```
# read a string and print the elements one by one
nm = 'India'

for i in nm:
    print(i)
```

```
I
n
d
i
a
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

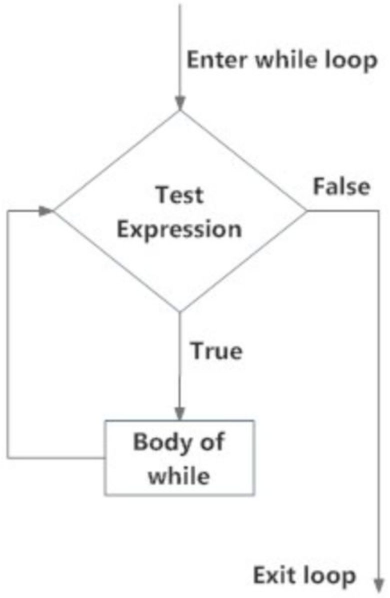
While Loop

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The while loop

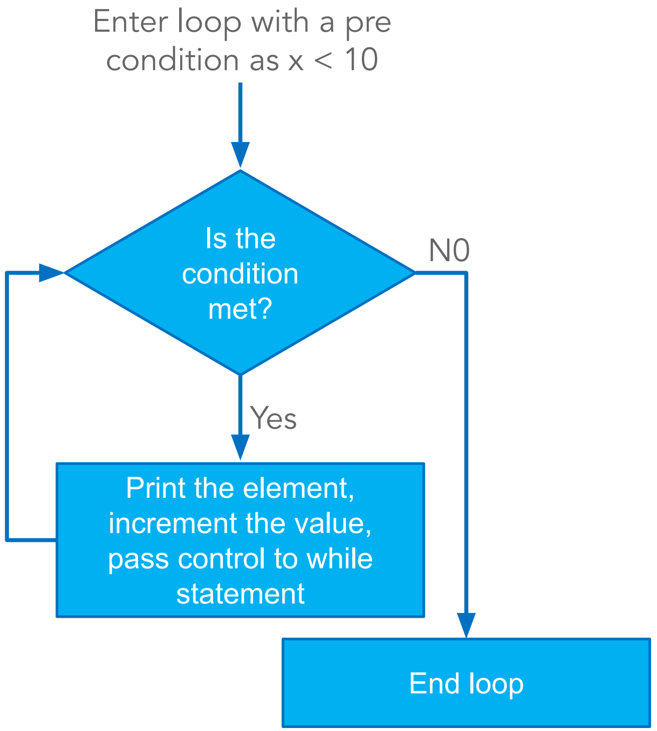
The while loop in Python is used to iterate over a block of code as long as the test expression (condition) is true.

We generally use this loop when we don't know beforehand, the number of times to iterate.

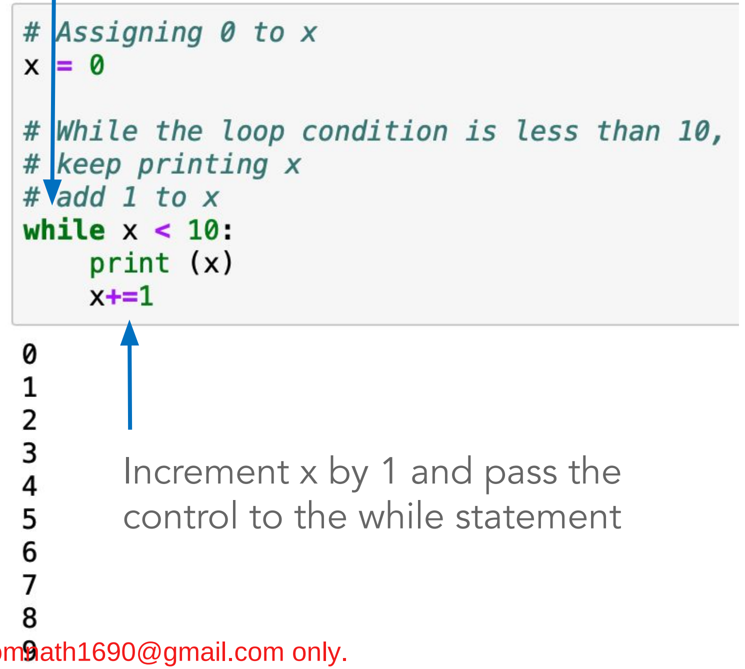


This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The while loop code example


Enter loop with a pre condition as x < 10
Is the condition met?
N0
Yes
Print the element, increment the value, pass control to while statement
End loop

The while statement checks the condition if x < 10



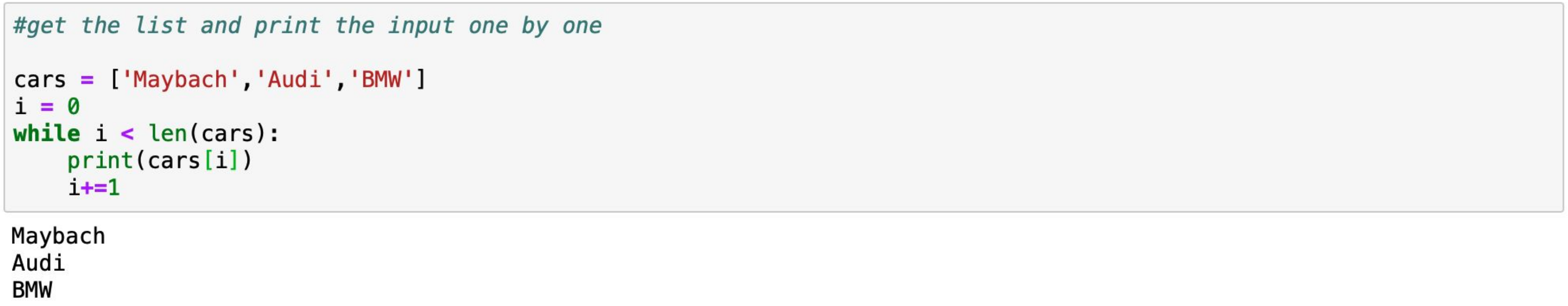
```
# Assigning 0 to x
x = 0

# While the loop condition is less than 10,
# keep printing x
# add 1 to x
while x < 10:
    print (x)
    x+=1
```

```
0
1
2
3
4
5
6
7
8
9
```

Increment x by 1 and pass the control to the while statement

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Print each element of a list using while loop



```
#get the list and print the input one by one

cars = ['Maybach','Audi','BMW']
i = 0
while i < len(cars):
    print(cars[i])
    i+=1
```

```
Maybach
Audi
BMW
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Break, Continue

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Exiting a loop

Loops in Python allows us to automate and repeat tasks.

But at times, due to certain factors, we may want to:

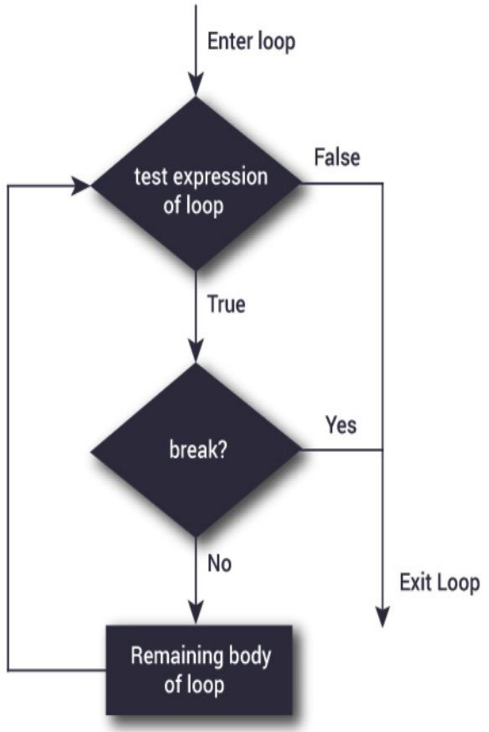
- Exit a loop completely
- Skip part of a loop before continuing

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Exit a loop with break statement

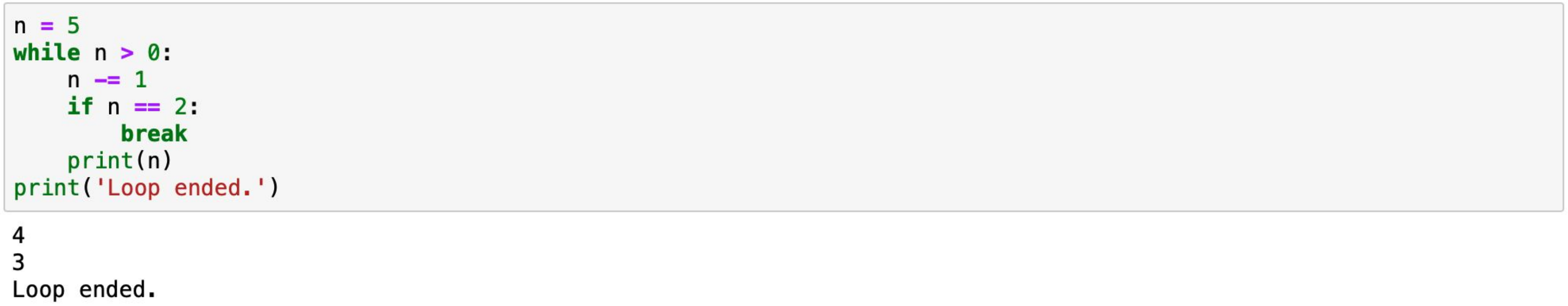
The break statement in Python terminates the current loop and resumes execution at the next statement

The break statement can be used in both while and for loops.



This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Exit a while loop with break statement



```python
n = 5
while n > 0:
    n -= 1
    if n == 2:
        break
    print(n)
print('Loop ended.')
```

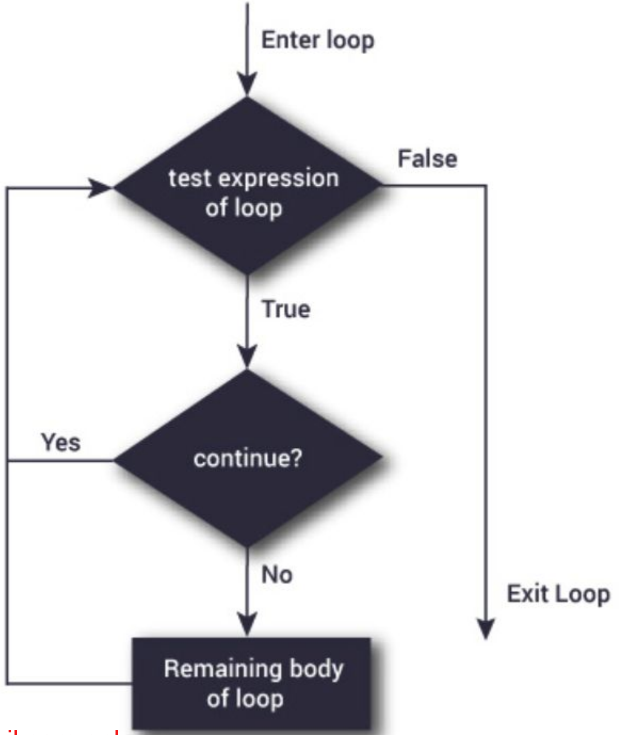
```
4
3
Loop ended.
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Skip part of the loop with continue statement

The continue statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.

The continue statement can be used in both while and for loops.



This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Skip part of the loop with continue statement

The for...loop iterates through the string character by character

```
for i in "greatlearning":
    if i == "i":
        continue
    print(i)

print("The end of code")
```

The if condition checks for the condition, x = "i"

The continue statement, when meets the condition, it ignores that character and moves the control to the beginning of the loop

```
g
r
e
a
t
l
e
a
r
n
n
g
The end of code
```

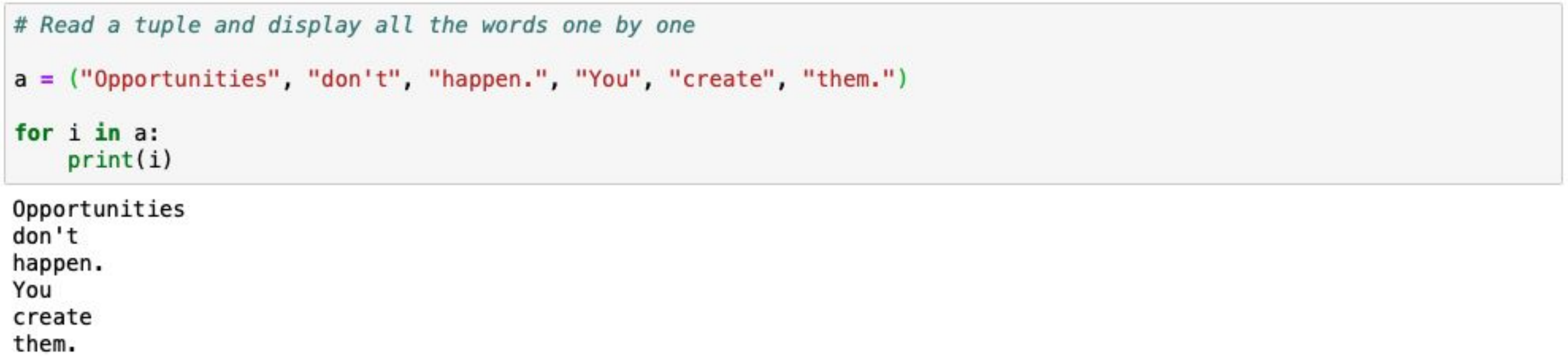
This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

More on Loops

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read each element of a tuple

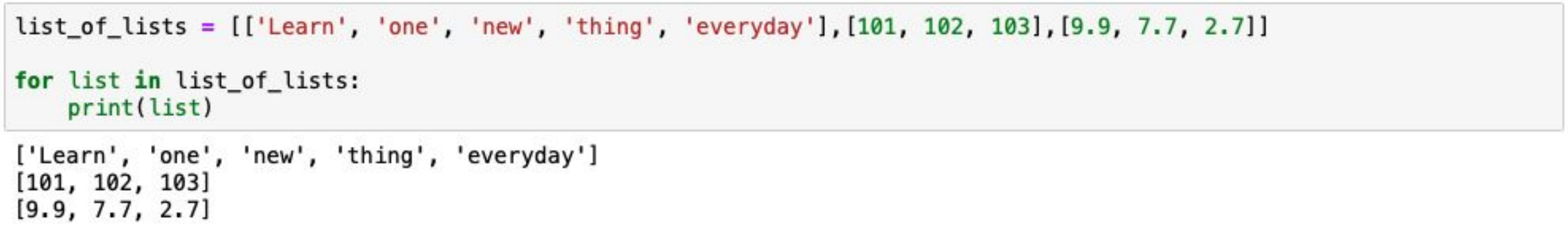


```python
# Read a tuple and display all the words one by one

a = ("Opportunities", "don't", "happen.", "You", "create", "them.")

for i in a:
    print(i)
```

```
Opportunities
don't
happen.
You
create
them.
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read through a list of list

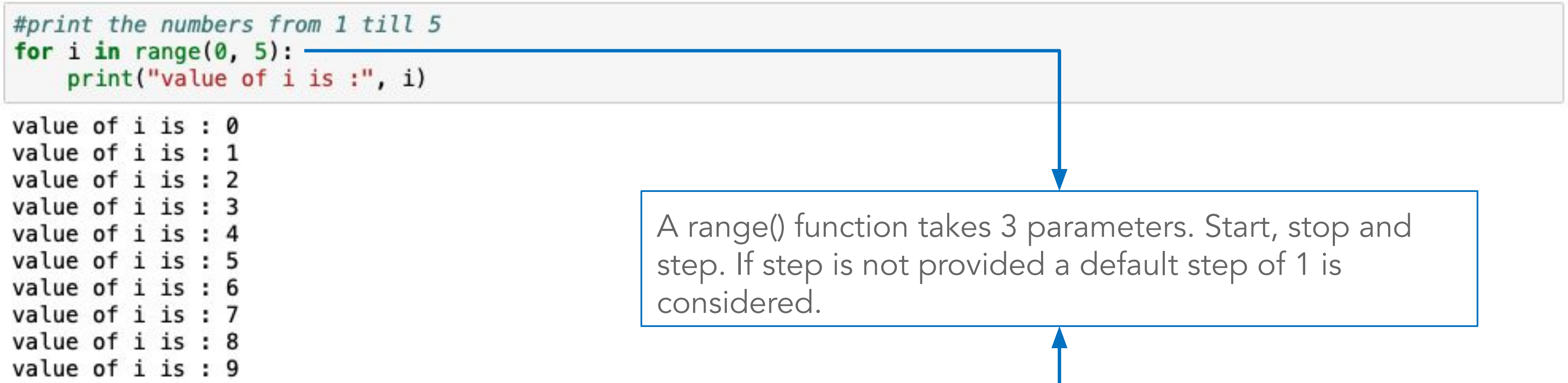


```
list_of_lists = [['Learn', 'one', 'new', 'thing', 'everyday'],[101, 102, 103],[9.9, 7.7, 2.7]]

for list in list_of_lists:
    print(list)
```

```
['Learn', 'one', 'new', 'thing', 'everyday']
[101, 102, 103]
[9.9, 7.7, 2.7]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

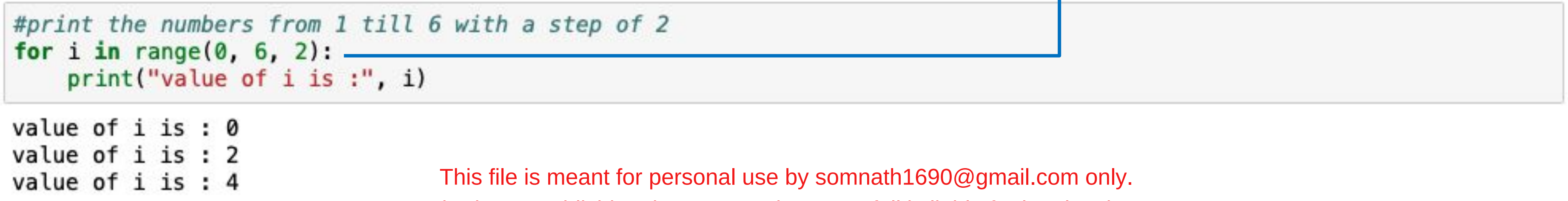
Loop through a range



```
#print the numbers from 1 till 5
for i in range(0, 5):
    print("value of i is :", i)
```

```
value of i is : 0
value of i is : 1
value of i is : 2
value of i is : 3
value of i is : 4
value of i is : 5
value of i is : 6
value of i is : 7
value of i is : 8
value of i is : 9
```

A range() function takes 3 parameters. Start, stop and step. If step is not provided a default step of 1 is considered.

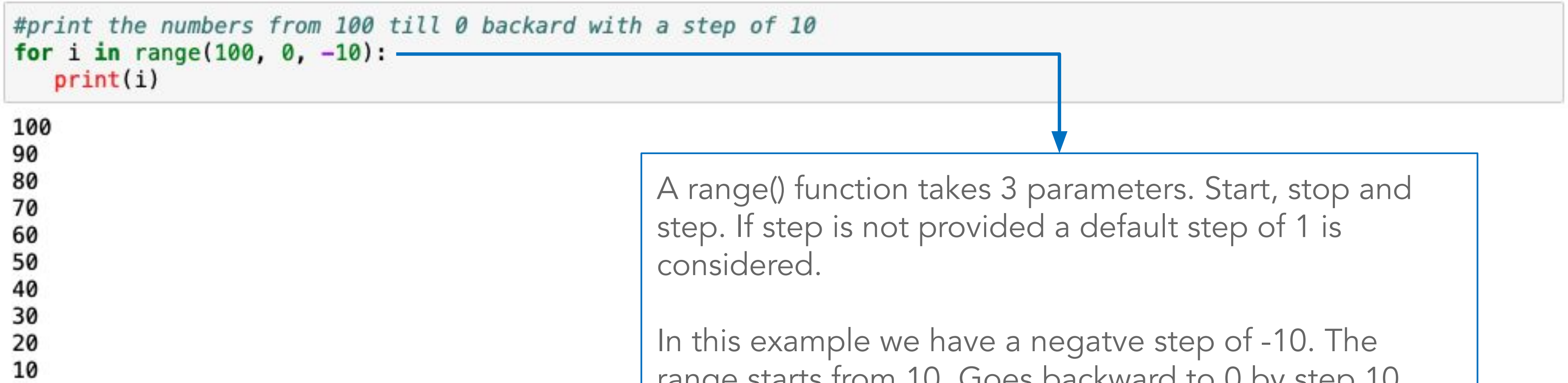


```
#print the numbers from 1 till 6 with a step of 2
for i in range(0, 6, 2):
    print("value of i is :", i)
```

```
value of i is : 0
value of i is : 2
value of i is : 4
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Loop through a range (negative step)



```
#print the numbers from 100 till 0 backard with a step of 10
for i in range(100, 0, -10):
    print(i)
```

```
100
90
80
70
60
50
40
30
20
10
```

A range() function takes 3 parameters. Start, stop and step. If step is not provided a default step of 1 is considered.

In this example we have a negatve step of -10. The range starts from 10. Goes backward to 0 by step 10

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Computation with for…loop

```
# read each element from the range and add 1 to each of the values
for i in range(1, 10, 2):
    print(i, "Squared value is :" , i + 1)
```

```
1 Squared value is : 2
3 Squared value is : 4
5 Squared value is : 6
7 Squared value is : 8
9 Squared value is : 10
```

```
# read each element from the range and square the values
for i in range(1, 10, 2):
    print(i, "Squared value is :" , i*i)
```

```
1 Squared value is : 1
3 Squared value is : 9
5 Squared value is : 25
7 Squared value is : 49
9 Squared value is : 81
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read through a dictionary object

```python
employee_dict = {'Name': 'Rajesh', 'Job': 'Data Scientist', 'Age': 27}
employee_dict
```

```
{'Name': 'Rajesh', 'Job': 'Data Scientist', 'Age': 27}
```

```python
## print all the items
for each_item in employee_dict.items():
    print(each_item)
```

```
('Name', 'Rajesh')
('Job', 'Data Scientist')
('Age', 27)
```

```python
## print all the keys
for item in employee_dict.keys():
    print(item)
```

```
Name
Job
Age
```

```python
## print all the values
for item in employee_dict.values():
    print(item)
```

```
Rajesh
Data Scientist
27
```

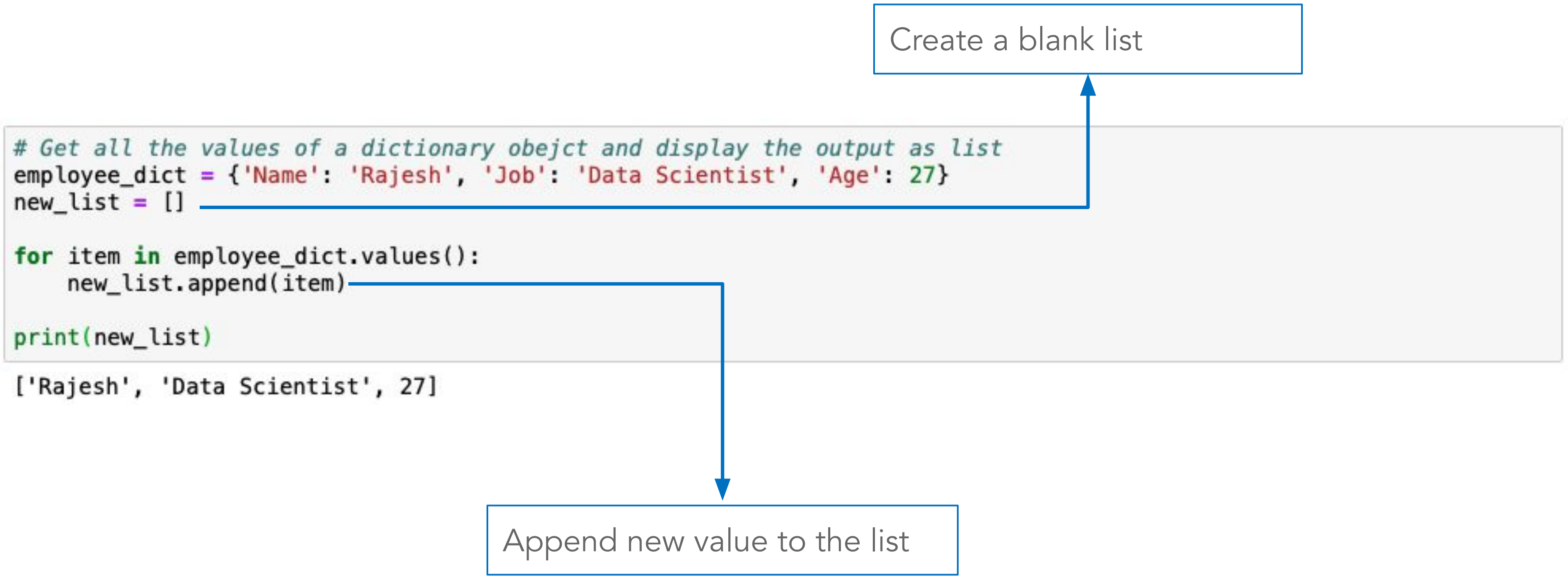
The dict.items() returns all key-value pair

The dict.keys() returns all keys

The dict.values() return all values

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read through the values in a dictionary to create a list



Create a blank list

```python
# Get all the values of a dictionary obejct and display the output as list
employee_dict = {'Name': 'Rajesh', 'Job': 'Data Scientist', 'Age': 27}
new_list = []

for item in employee_dict.values():
    new_list.append(item)

print(new_list)
```

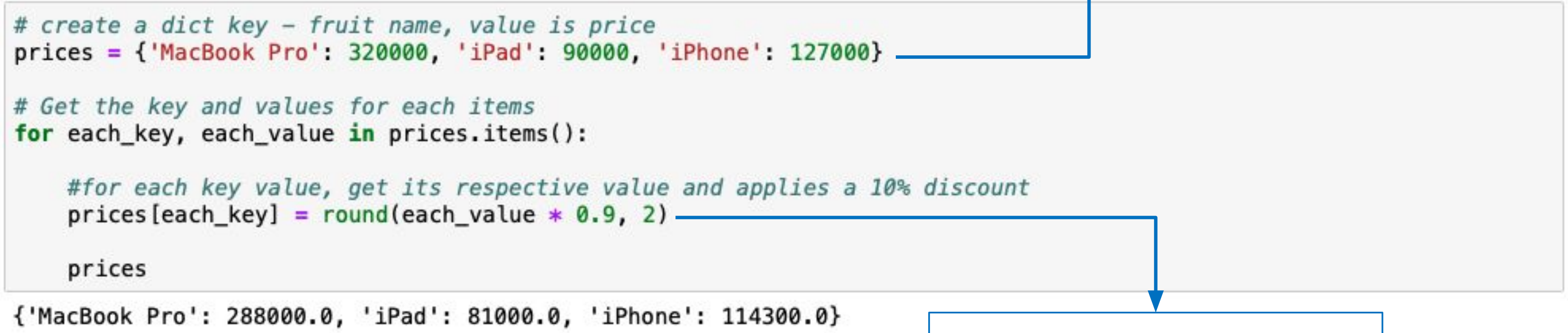
```
['Rajesh', 'Data Scientist', 27]
```

Append new value to the list

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Replace values in a dictionary with computed values

A dictionary object



```python
# create a dict key - fruit name, value is price
prices = {'MacBook Pro': 320000, 'iPad': 90000, 'iPhone': 127000}

# Get the key and values for each items
for each_key, each_value in prices.items():

    #for each key value, get its respective value and applies a 10% discount
    prices[each_key] = round(each_value * 0.9, 2)

    prices
```

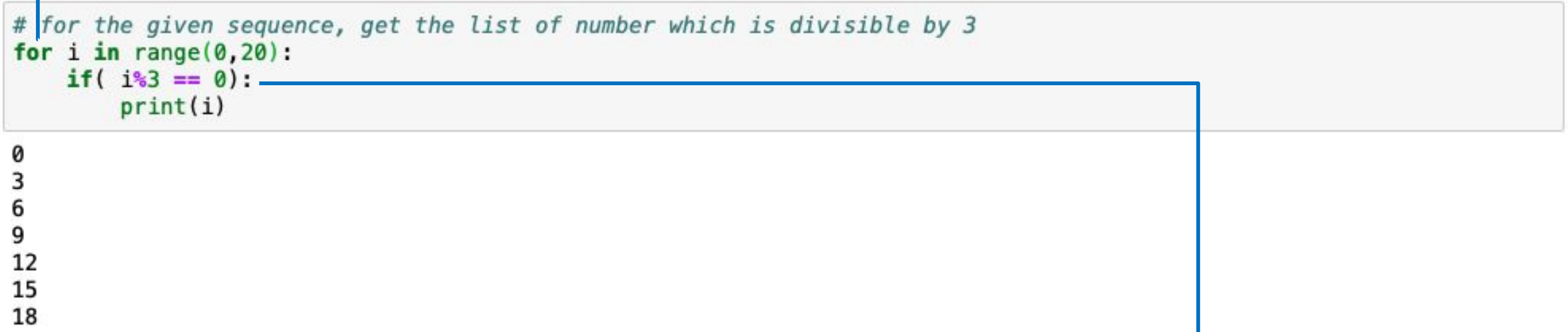
```
{'MacBook Pro': 288000.0, 'iPad': 81000.0, 'iPhone': 114300.0}
```

Appplies a 10* discount on the values

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

The for loop with conditional statements

For loop



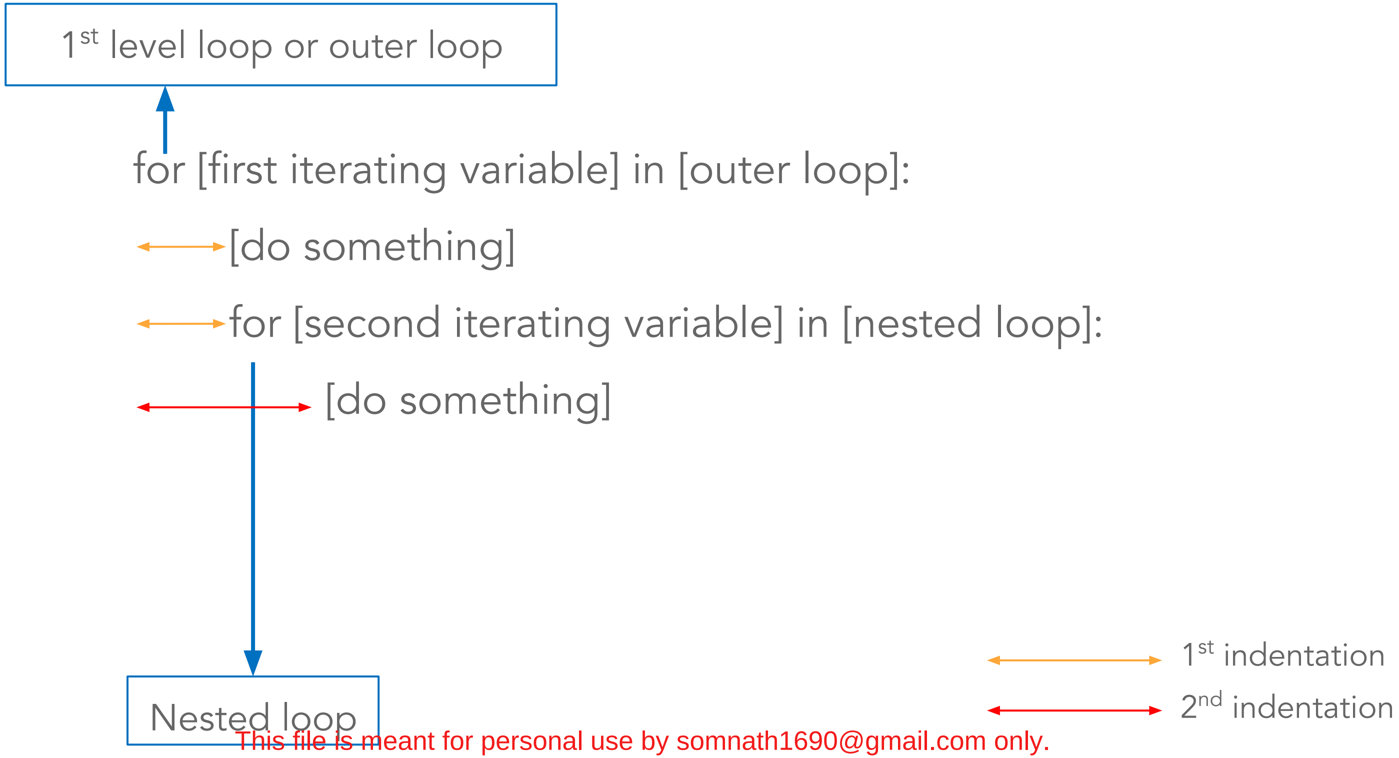
```
# for the given sequence, get the list of number which is divisible by 3
for i in range(0,20):
    if( i%3 == 0):
        print(i)
```

```
0
3
6
9
12
15
18
```

Conditional statement to check if the remainder of a number divider by 3 is zero

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

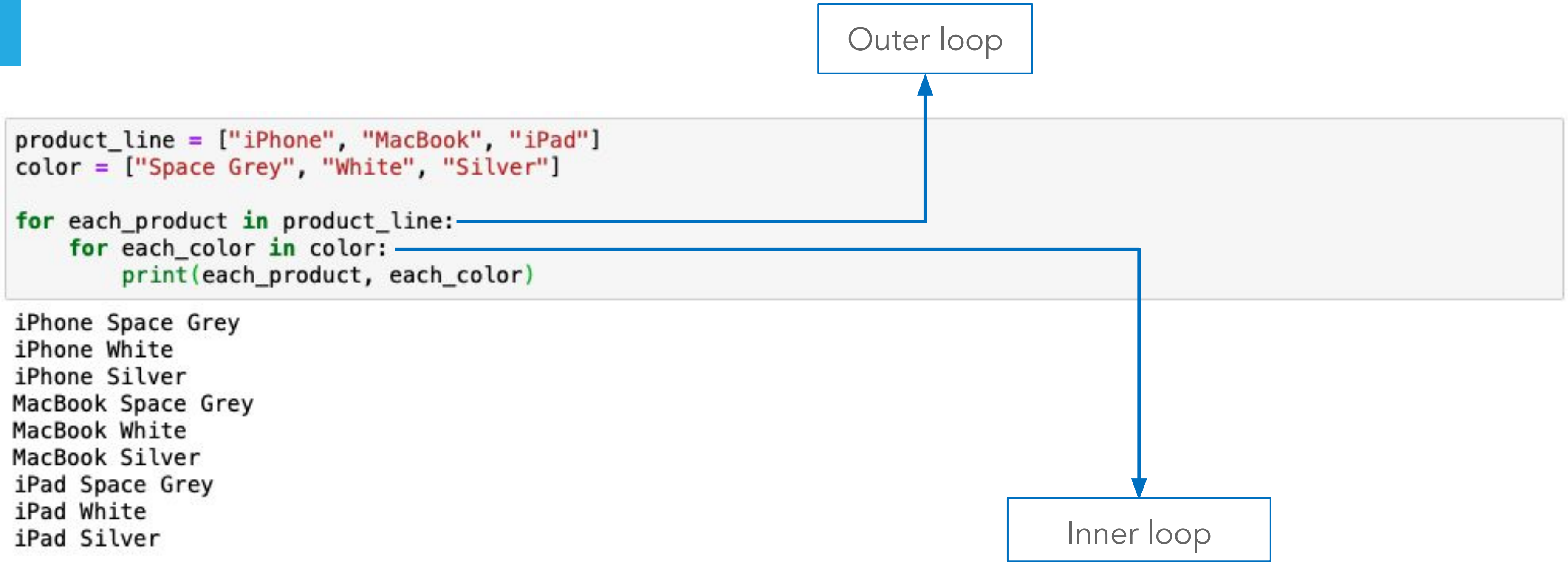
Nested for loop



```
for [first iterating variable] in [outer loop]:
    [do something]
    for [second iterating variable] in [nested loop]:
        [do something]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested for loop

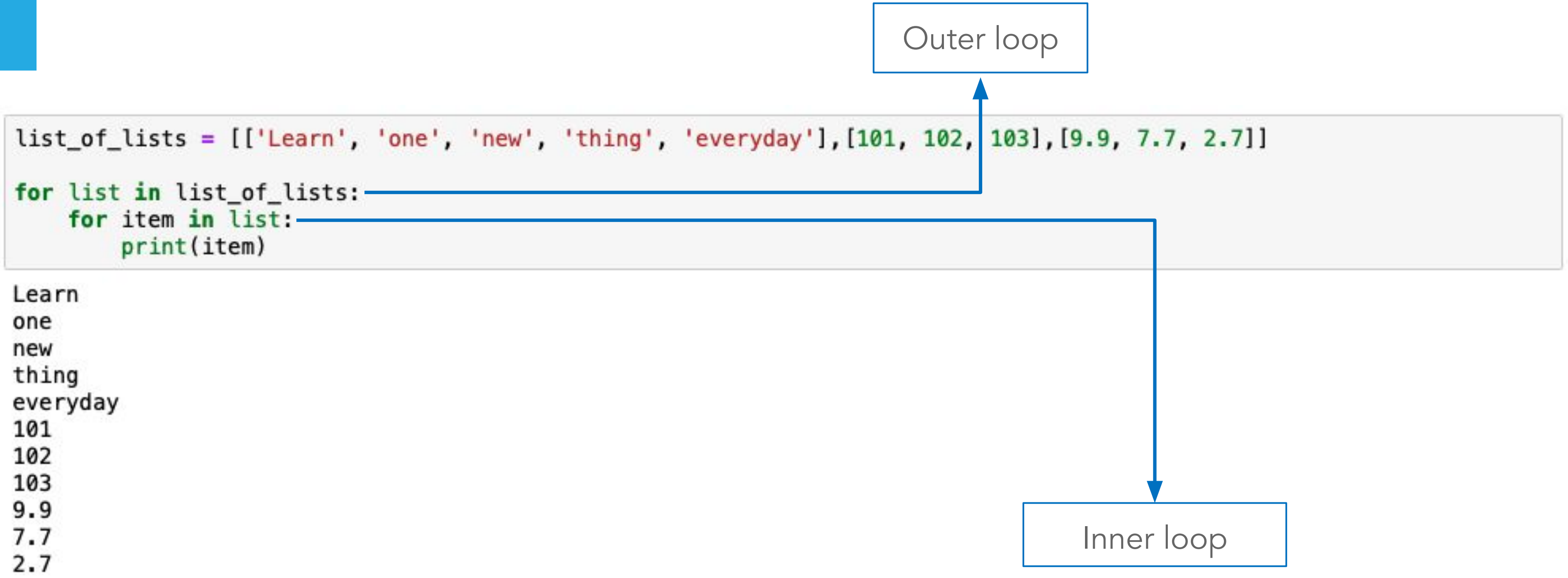


```python
product_line = ["iPhone", "MacBook", "iPad"]
color = ["Space Grey", "White", "Silver"]

for each_product in product_line:
    for each_color in color:
        print(each_product, each_color)
```

```
iPhone Space Grey
iPhone White
iPhone Silver
MacBook Space Grey
MacBook White
MacBook Silver
iPad Space Grey
iPad White
iPad Silver
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested for loop with list of lists



```python
list_of_lists = [['Learn', 'one', 'new', 'thing', 'everyday'],[101, 102, 103],[9.9, 7.7, 2.7]]

for list in list_of_lists:
    for item in list:
        print(item)
```

```
Learn
one
new
thing
everyday
101
102
103
9.9
7.7
2.7
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Example: Calculating sum of numbers

```
n = int(input("enter the count of sequential nummbers you want to add :"))

# initialize sum and counter
sum = 0
i = 1

while i <= n:
    sum = sum + i
    i = i+1    # update counter

    # print the sum
    print("The sum is", sum)
```

```
enter the count of sequential nummbers you want to add :5
The sum is 1
The sum is 3
The sum is 6
The sum is 10
The sum is 15
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

While loop with else statement

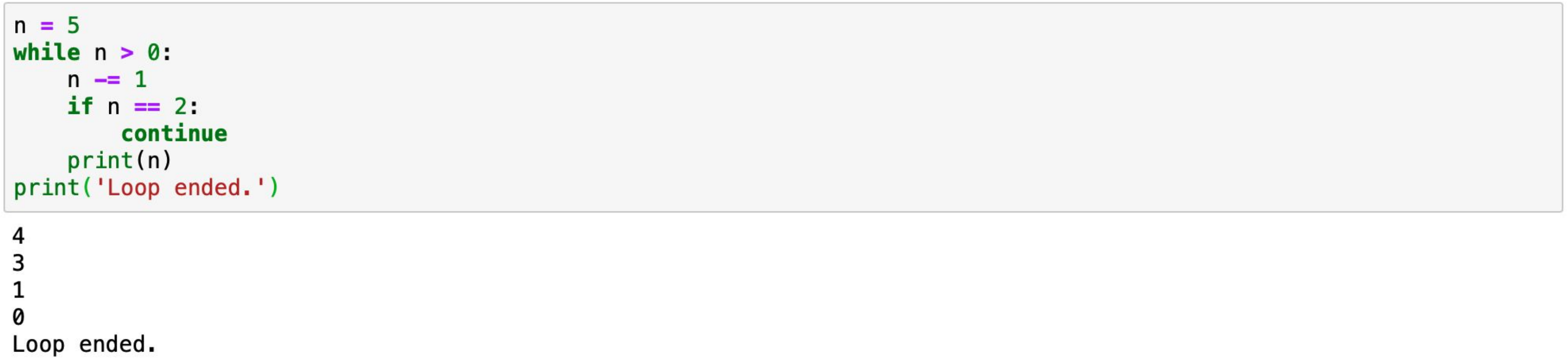
The else part is executed if the condition in the while loop evaluates to False.

```
# initialize  counter
counter = 0

while counter < 3:
    print("Inside loop")
    counter = counter + 1
else:
    print("Inside else") # as the count reached 3, it exits out of the loop
```

```
Inside loop
Inside loop
Inside loop
Inside else
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Exit a while loop with continue statement



```python
n = 5
while n > 0:
    n -= 1
    if n == 2:
        continue
    print(n)
print('Loop ended.')
```

```
4
3
1
0
Loop ended.
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

List Comprehension

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

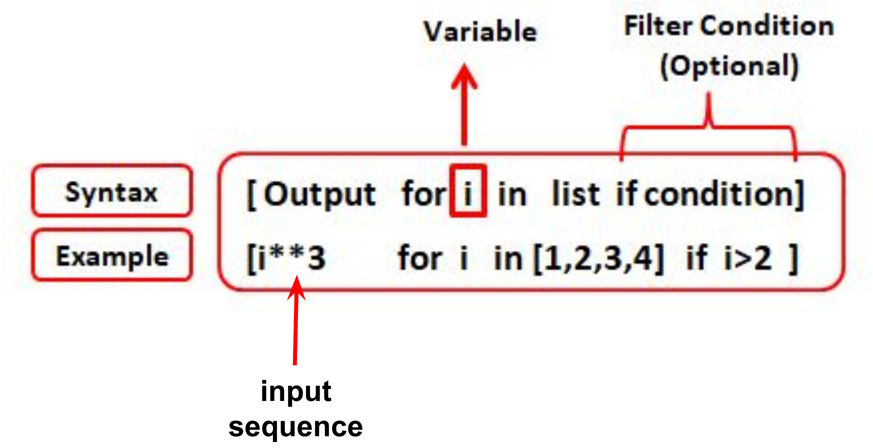
What is a list comprehension?

- List comprehensions provide a concise way to create lists
- It consists of [ ] containing:
  - an input sequence
  - a variable representing numbers of the input sequence
  - a conditional expression (optional)
  - an output expression producing a list
- The result will be a new list resulting from evaluating the expression in the context of the for and if clauses which follow it. The list comprehension always returns a result list

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

What is a list comprehension?

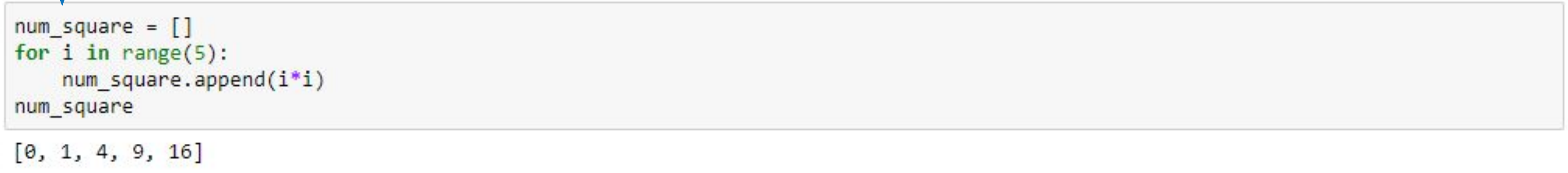
- It consists of [ ] containing:
  - an input sequence
  - a variable representing numbers of the input sequence
  - a conditional expression (optional)
  - an output expression producing a list


Variable
Filter Condition (Optional)
Syntax
[Output for i in list if condition]
Example
[i**3 for i in [1,2,3,4] if i>2]
input sequence

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

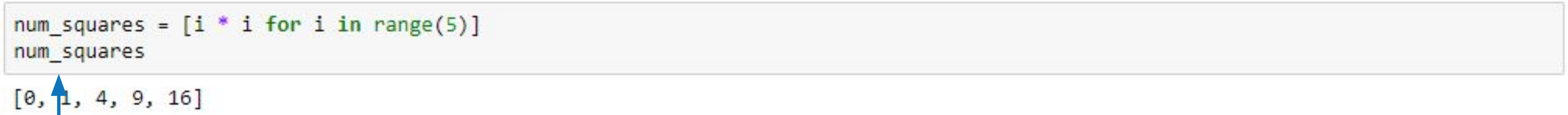
List comprehension vs loops

Create a list using for loop



```
num_square = []
for i in range(5):
    num_square.append(i*i)
num_square
```

```
[0, 1, 4, 9, 16]
```



```
num_squares = [i * i for i in range(5)]
num_squares
```

```
[0, 1, 4, 9, 16]
```

Create a list using list comprehension

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

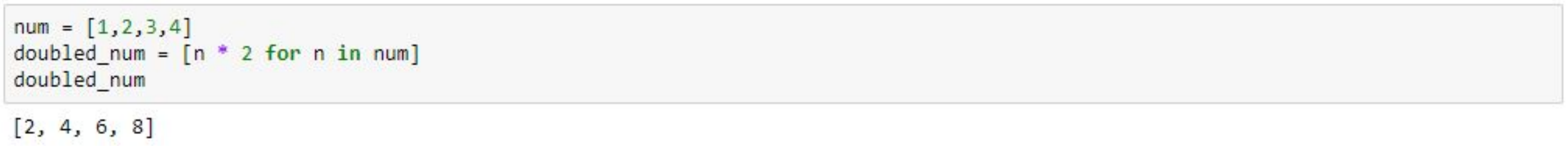
did you know?

List comprehension reduces 3 lines of code into one, which will be instantly recognizable to anyone.

It is faster, as Python will allocate the list's memory first, before appending the elements to it, instead of having to resize on runtime.

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

For loop using list comprehension

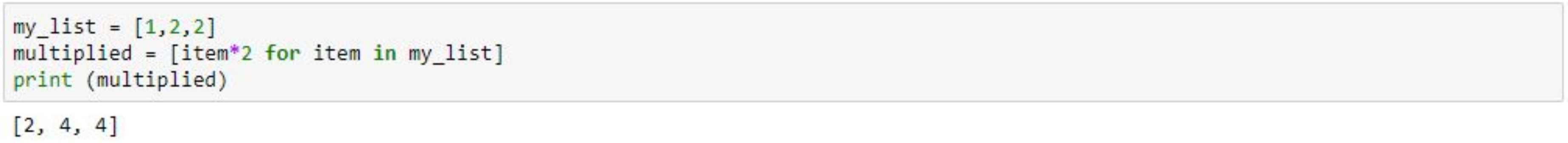


```
num = [1,2,3,4]
doubled_num = [n * 2 for n in num]
doubled_num
```

```
[2, 4, 6, 8]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Multiply each element of the list by 2 using list comprehension

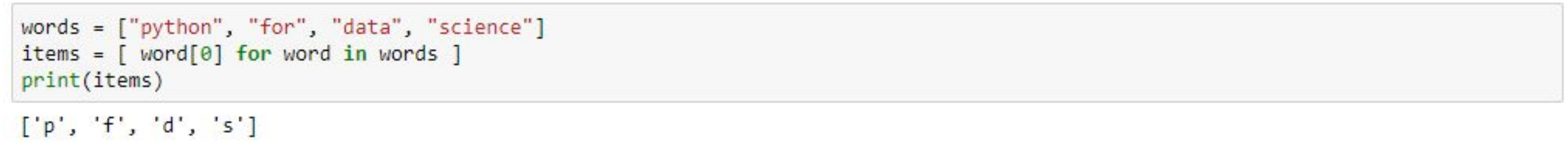


```
my_list = [1,2,2]
multiplied = [item*2 for item in my_list]
print (multiplied)
```

```
[2, 4, 4]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Read the first letter of each word using list comprehension

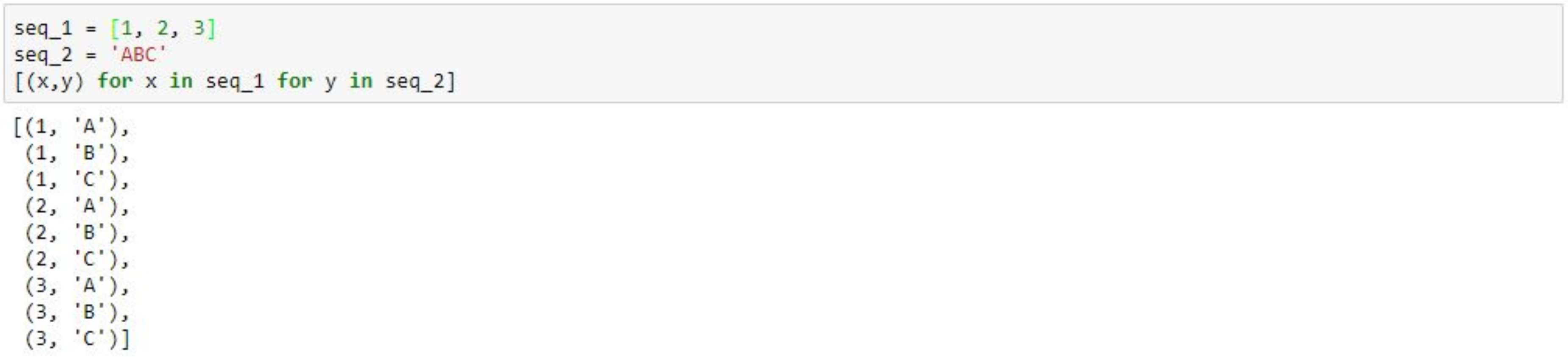


```python
words = ["python", "for", "data", "science"]
items = [ word[0] for word in words ]
print(items)
```

```
['p', 'f', 'd', 's']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Iteration over more than one iterable in a list using list comprehension

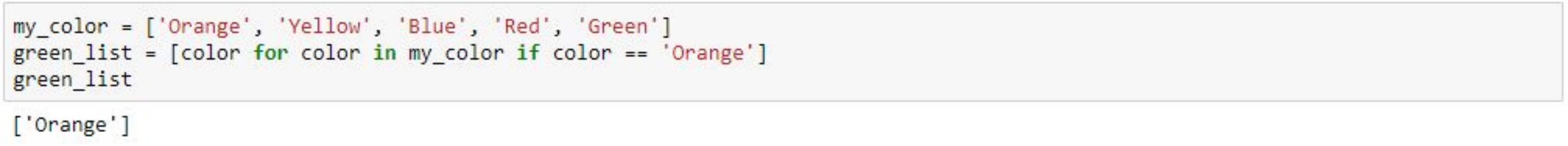


```
seq_1 = [1, 2, 3]
seq_2 = 'ABC'
[(x,y) for x in seq_1 for y in seq_2]
```

```
[(1, 'A'),
 (1, 'B'),
 (1, 'C'),
 (2, 'A'),
 (2, 'B'),
 (2, 'C'),
 (3, 'A'),
 (3, 'B'),
 (3, 'C')]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

If statement within a list comprehension

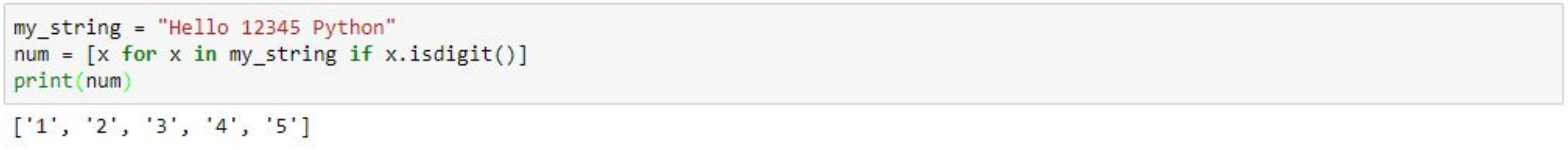


```
my_color = ['Orange', 'Yellow', 'Blue', 'Red', 'Green']
green_list = [color for color in my_color if color == 'Orange']
green_list
```

```
['Orange']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Extract all the numbers from the string using list comprehension

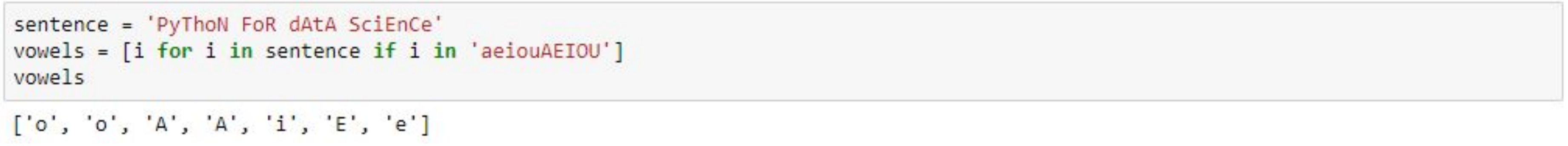


```
my_string = "Hello 12345 Python"
num = [x for x in my_string if x.isdigit()]
print(num)
```

```
['1', '2', '3', '4', '5']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Extract all the vowels from the string using list comprehension

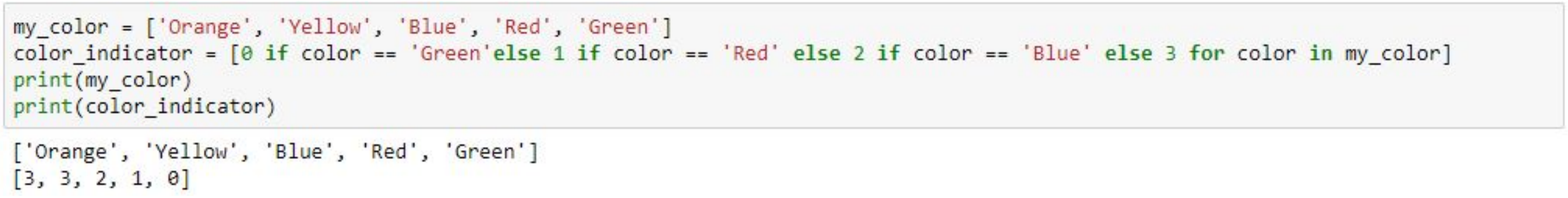


```python
sentence = 'PyThoN FoR dAtA SciEnCe'
vowels = [i for i in sentence if i in 'aeiouAEIOU']
vowels
```

```
['o', 'o', 'A', 'A', 'i', 'E', 'e']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Nested condition within a list comprehension

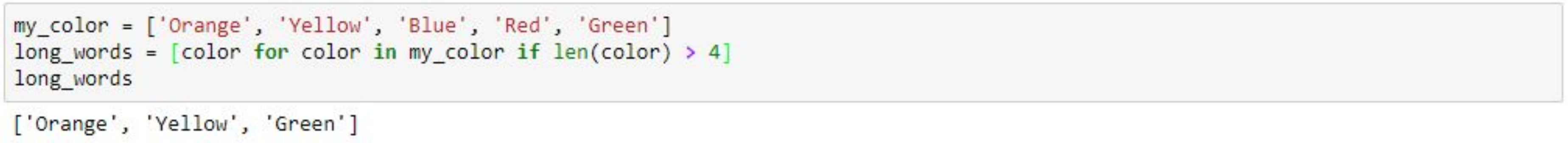


```
my_color = ['Orange', 'Yellow', 'Blue', 'Red', 'Green']
color_indicator = [0 if color == 'Green'else 1 if color == 'Red' else 2 if color == 'Blue' else 3 for color in my_color]
print(my_color)
print(color_indicator)
```

```
['Orange', 'Yellow', 'Blue', 'Red', 'Green']
[3, 3, 2, 1, 0]
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Find colors in list that are longer than 4 character using list comprehension

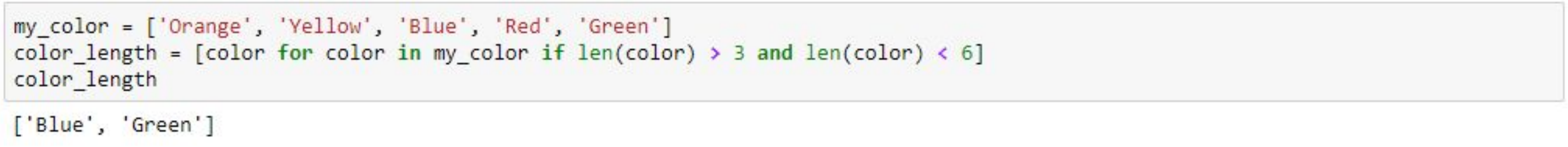


```
my_color = ['Orange', 'Yellow', 'Blue', 'Red', 'Green']
long_words = [color for color in my_color if len(color) > 4]
long_words
```

```
['Orange', 'Yellow', 'Green']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Find colors with a length between 3 and 6 using list comprehension



```
my_color = ['Orange', 'Yellow', 'Blue', 'Red', 'Green']
color_length = [color for color in my_color if len(color) > 3 and len(color) < 6]
color_length
```

```
['Blue', 'Green']
```

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.

Thank You

This file is meant for personal use by somnath1690@gmail.com only.
Sharing or publishing the contents in part or full is liable for legal action.