Python Basic To Advanced

WRITTEN By Meharin Hasna Puspo

Python-Basic to Advanced

Table of Contents

Introduction to Python

Basic Of Python

Operators in Python

if -else in python

Loop in python

More Basic of Python

Function in Python

Advanced Topic in Python

Python-Mini Project

Written By Meharin Hasna Puspo

Python-Basic to Advanced

Introduction to Python

What is Python?

Python is high-level programming language. Python language Created by Guido van Rossum. Python First Released in 1991.

Why Python So Popular Language ?

- Simple Syntax
- Totally Free & Open Source
- Support For Beginners & Experts
- Used By Popular Organization

What is Python Used For?

- Machine Learning & Artificial Intelligence
- Data Analysis
- Web Development
- Automation
- Game Development

Python Required Softwares

- Python
- PyCharm (Ofline Code IDE)
- Google Colab(Online Code IDE)

Python Version History

- Python 1.X & 2.X
- Python 3.X

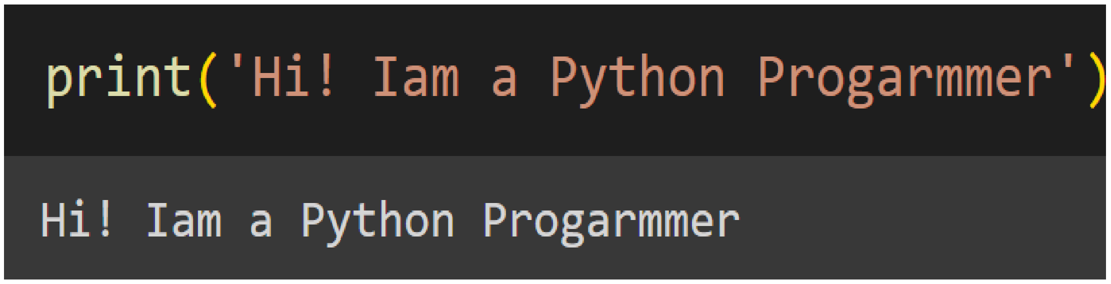
Python-Basic to Advanced

Basic Of Python

Python Print(“ “)

String in python Use **(” ”)**

Example

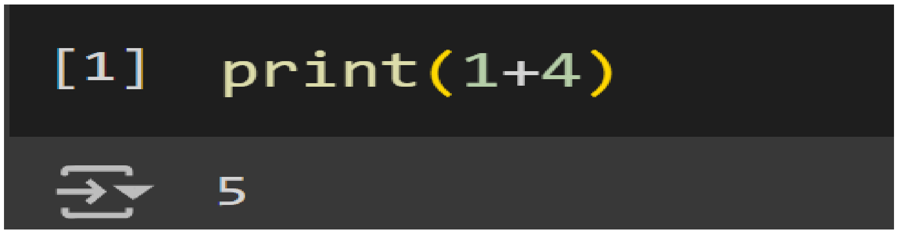


```
print('Hi! Iam a Python Progarmmer')
```

```
Hi! Iam a Python Progarmmer
```

Print(1+4)

Example



```
[1] print(1+4)
```

```
5
```

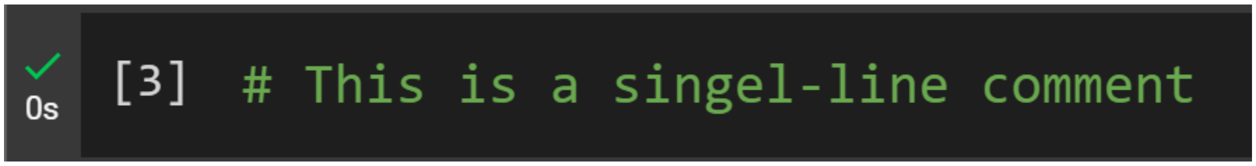
Python Comment

Comments are used for code explanation

- Singel-Line Comment
- Multi-Line Comment

Singel-Line Comment Use **#** Symbol

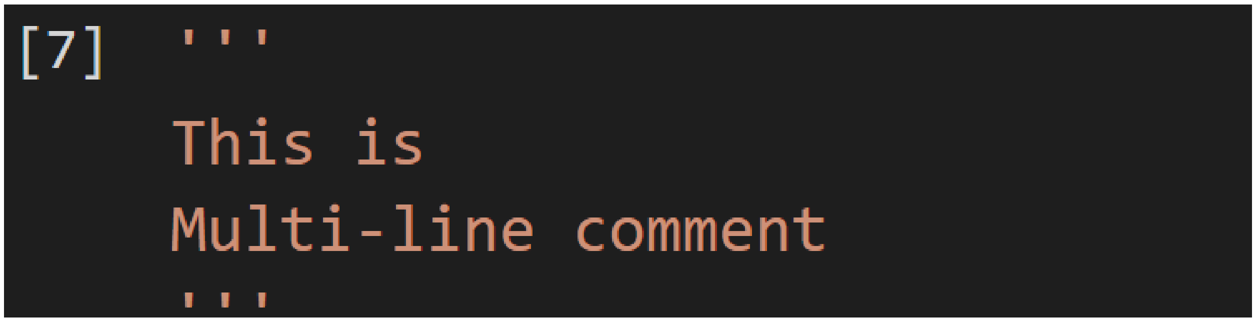
Example



```
[3] # This is a singel-line comment
```

Multi-Line Comment Use “”” “”” Symbol

Example



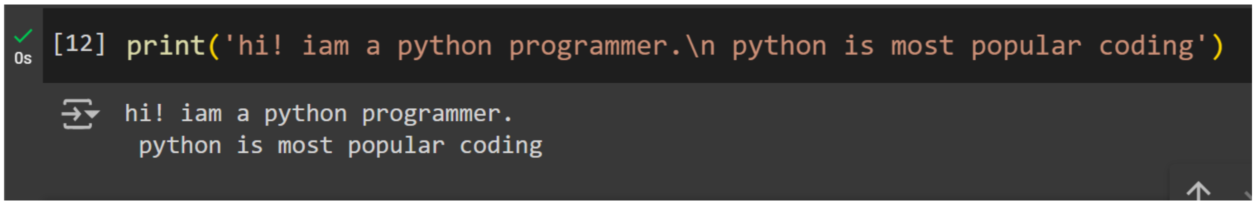
```
[7] '''
    This is
    Multi-line comment
    '''
```

Python Escape Sequences

Python Escape Sequences Is a combination characters that represent a special characters when used inside a string .

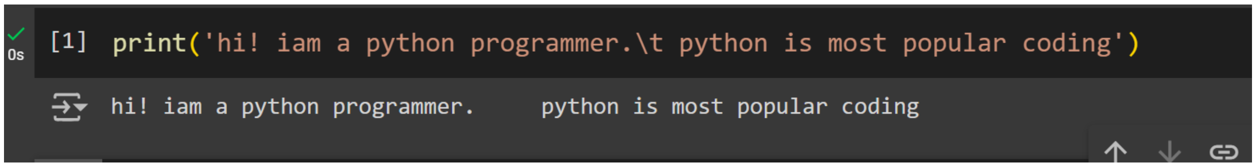
Common Python Escape Sequences:

Newline (line break) = \n



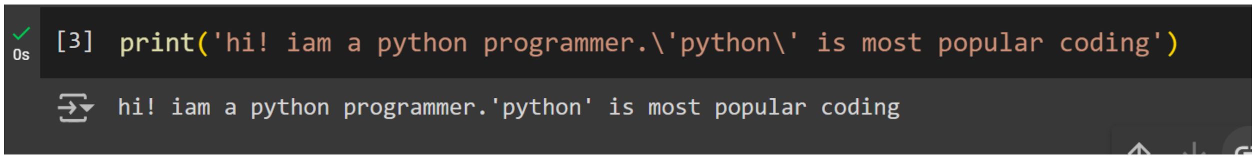
```
[12] print('hi! iam a python programmer.\n python is most popular coding')

hi! iam a python programmer.
 python is most popular coding
```

Tab=\t



```
[1] print('hi! iam a python programmer.\t python is most popular coding')

hi! iam a python programmer.      python is most popular coding
```

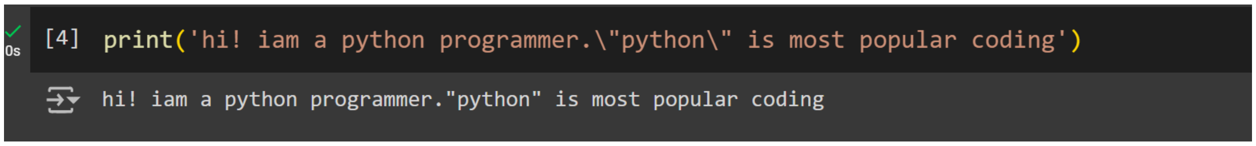
Singel Quote = \’



```
[3] print('hi! iam a python programmer.\'python\' is most popular coding')
```

```
hi! iam a python programmer.'python' is most popular coding
```

Double Quote = \”



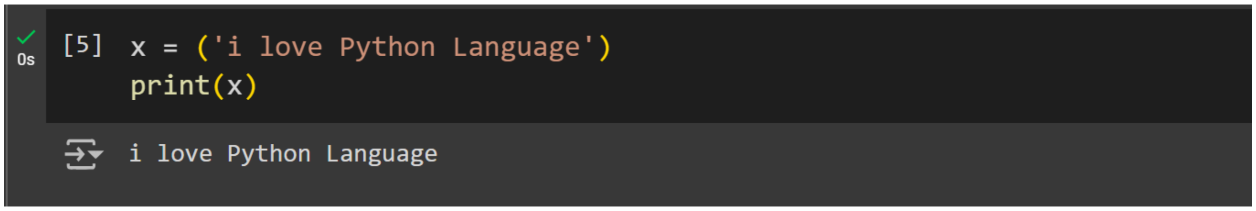
```
[4] print('hi! iam a python programmer.\"python\" is most popular coding')
```

```
hi! iam a python programmer."python" is most popular coding
```

Python Variables

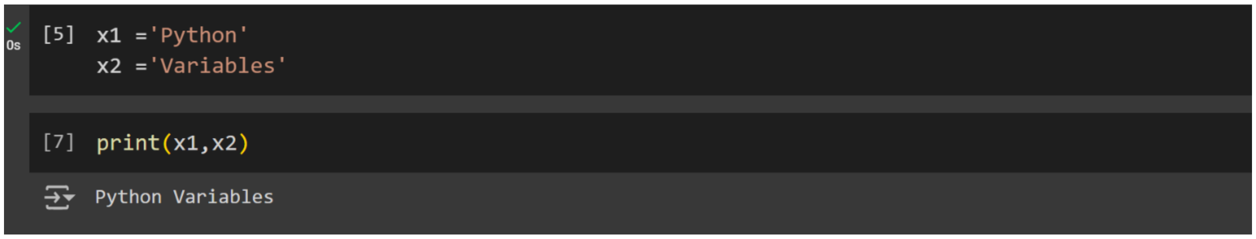
Python Variables Used to Store **Data Values .** Python variables are initialized with strings.

Example



```
[5] x = ('i love Python Language')
    print(x)
```

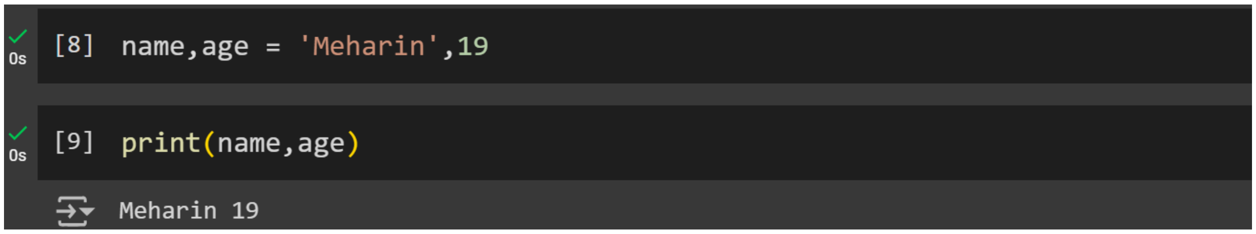
```
i love Python Language
```



```
[5] x1 ='Python'
    x2 ='Variables'
```

```
[7] print(x1,x2)
```

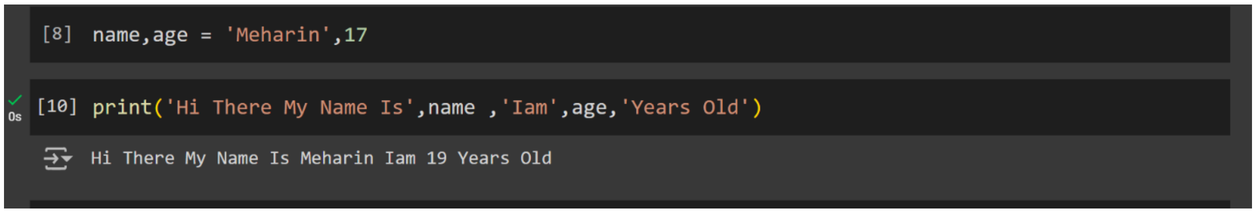
```
Python Variables
```



```
[8] name,age = 'Meharin',19
```

```
[9] print(name,age)
```

```
Meharin 19
```



```
[8] name,age = 'Meharin',17
```

```
[10] print('Hi There My Name Is',name ,'Iam',age,'Years Old')
```

```
Hi There My Name Is Meharin Iam 19 Years Old
```

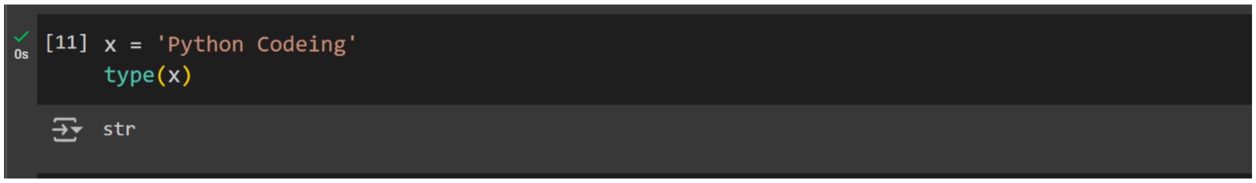
Python Data Types

Python data type specifies the type of data a variable can hold.

Common Python Data Types:

String Data 'str'

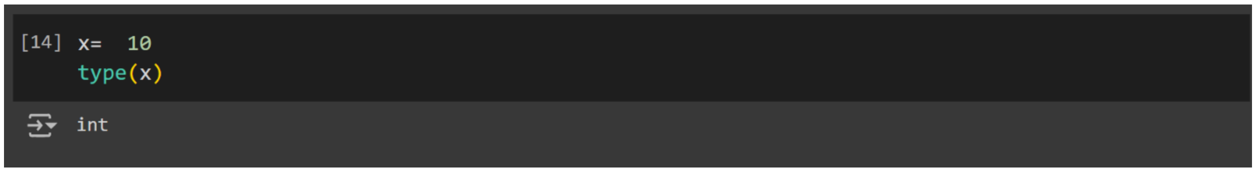
Example



```
[11] x = 'Python Codeing'
     type(x)

str
```

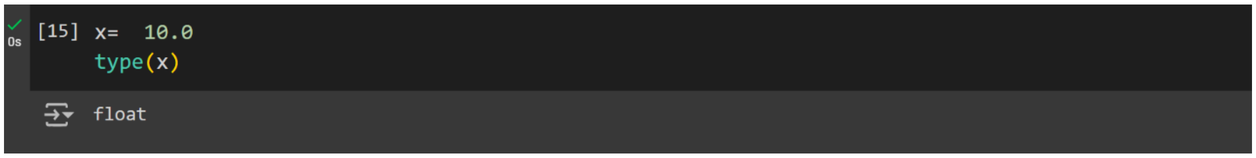
Numeric Data Types **(int , float,complex)**

Example **(int data type)**



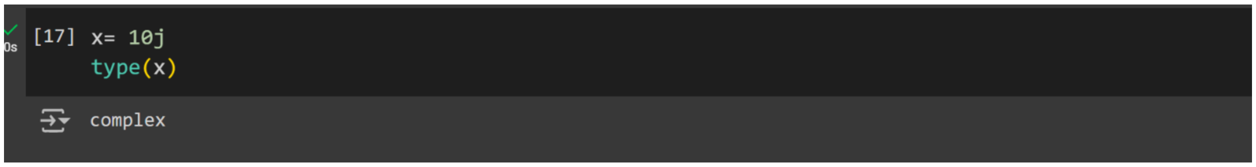
```
[14] x=  10
     type(x)

int
```

Example **(float data type)**



```
[15] x=  10.0
     type(x)

float
```

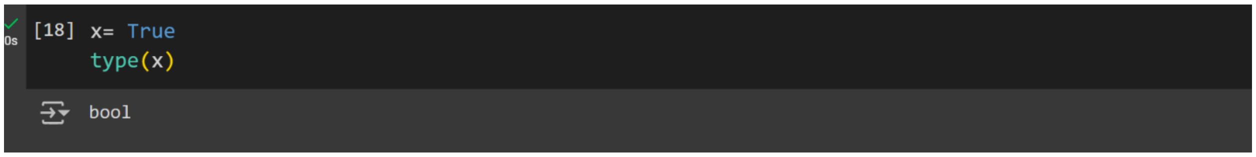
Example **(Complex data type)**



```
[17] x= 10j
     type(x)

complex
```

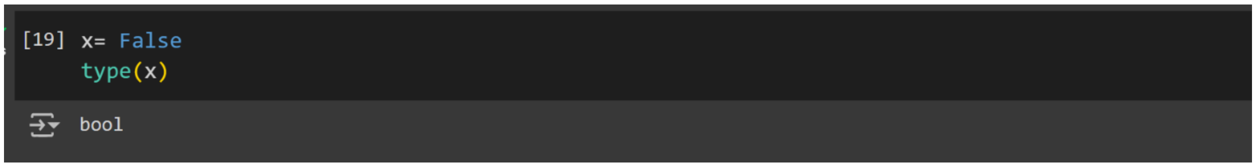
Boolean Data Type **(True , False)**

Example (True)



```
[18] x= True
     type(x)

bool
```

Example (False)

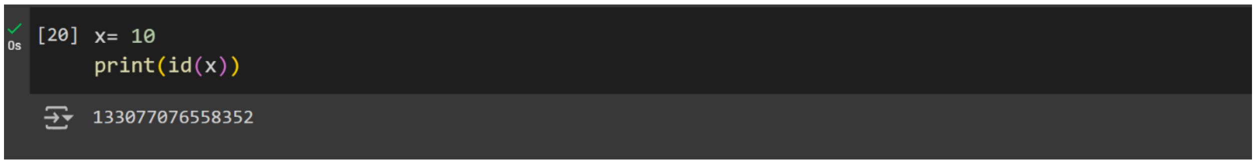


```
[19] x= False
     type(x)

bool
```

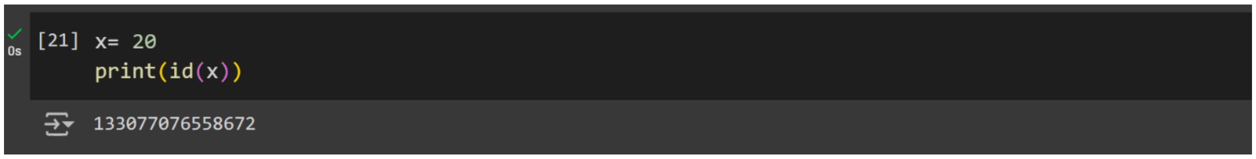
Python Memory Management

Memory management in Python refers to how the language handles the allocation memory during the execution of programs.

Example(id())



```
[20] x= 10
     print(id(x))

133077076558352
```



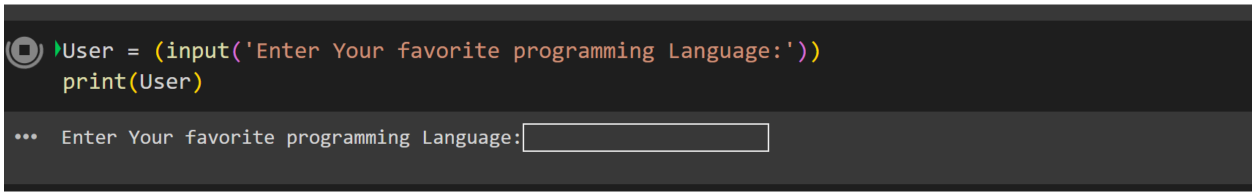
```
[21] x= 20
     print(id(x))

133077076558672
```

Python-Basic to Advanced

Python User Input

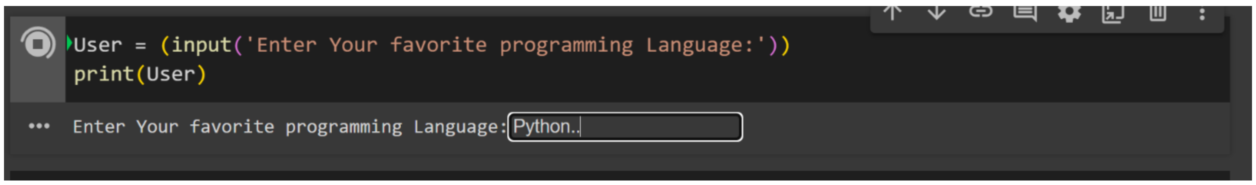
Python User Input the input() function is used to capture user input.

Example



```
User = (input('Enter Your favorite programming Language:'))
print(User)
```

```
Enter Your favorite programming Language:
```



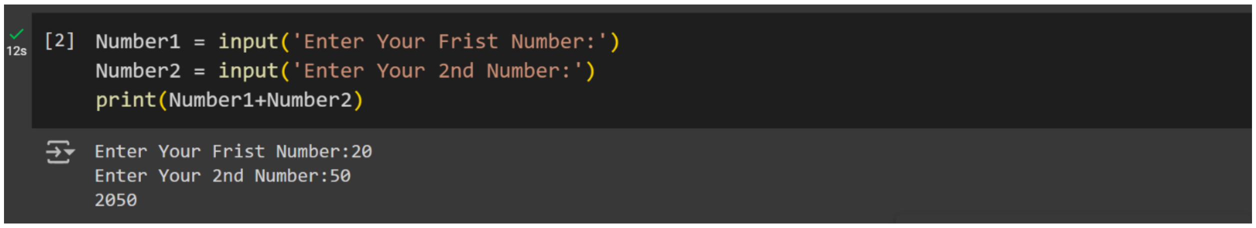
```
User = (input('Enter Your favorite programming Language:'))
print(User)
```

```
Enter Your favorite programming Language: Python..
```

Python Type Casting

Python always accepts everything as string data when taking user input. Many times when user input is required to take other data then Python type casting is required.

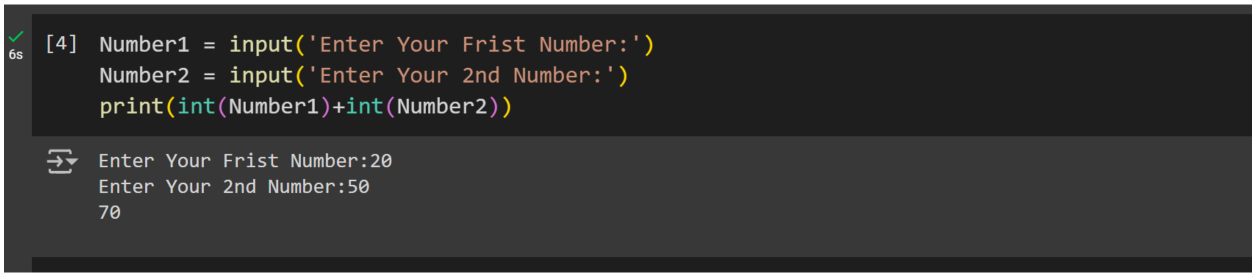
Example **(Before type casting)**



```
[2] Number1 = input('Enter Your Frist Number:')
    Number2 = input('Enter Your 2nd Number:')
    print(Number1+Number2)
```

```
Enter Your Frist Number:20
Enter Your 2nd Number:50
2050
```

Example **(Before type casting)**



```
[4] Number1 = input('Enter Your Frist Number:')
    Number2 = input('Enter Your 2nd Number:')
    print(int(Number1)+int(Number2))
```

```
Enter Your Frist Number:20
Enter Your 2nd Number:50
70
```

Written By Meharin Hasna Puspo

Python-Basic to Advanced

Operators in Python

Python Operators

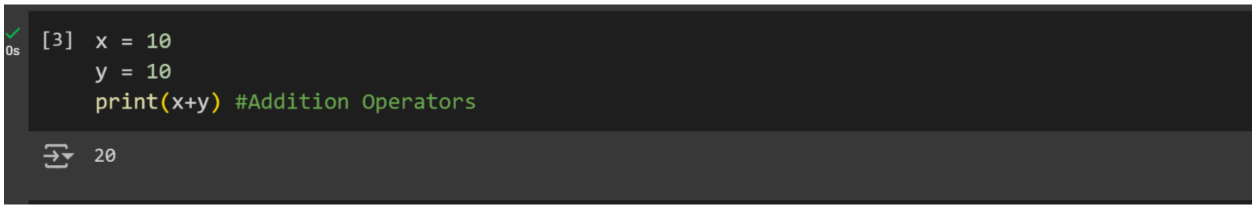
Operators in Python are special symbols that operate on variables and values.

Types of Operators in Python:

Arithmetic Operators

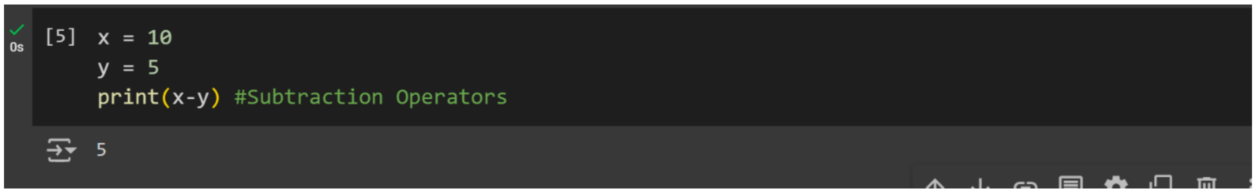
Operator	Name	Example
+	Addition	5+5=10
-	Subtraction	5-5=0
*	Multiplication	5*5=25
/	Division	5/5=1
%	Modulus	5%5=0
**	Exponentiation	5**5=3125
//	Floor Division	5//5=1

Example (**Addition Operators)**



```
[3] x = 10
    y = 10
    print(x+y) #Addition Operators

20
```

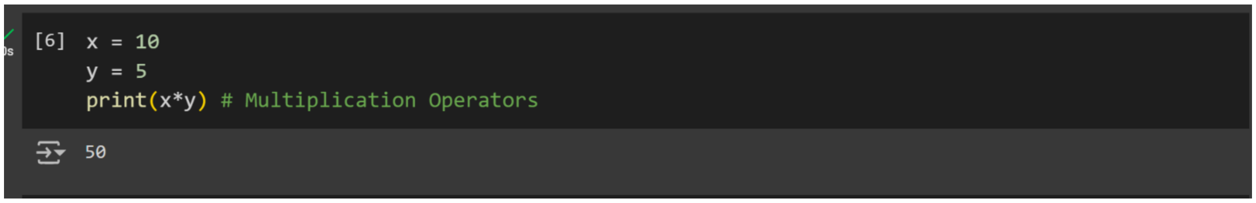
Example (**Subtraction Operators)**



```
[5] x = 10
    y = 5
    print(x-y) #Subtraction Operators

5
```

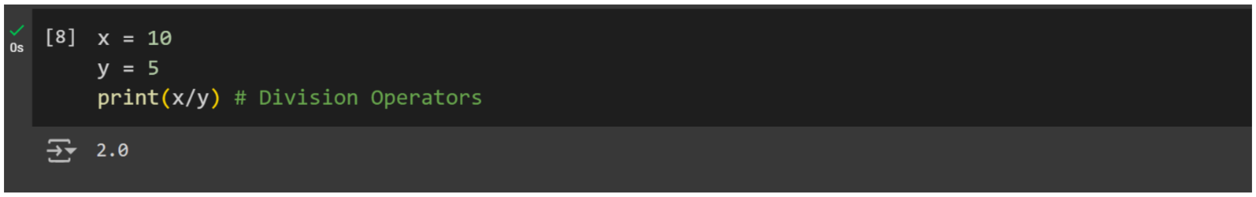
Written By Meharin Hasna Puspo

Example (Multiplication Operators)



```
[6] x = 10
    y = 5
    print(x*y) # Multiplication Operators

50
```

Example (Division Operators)

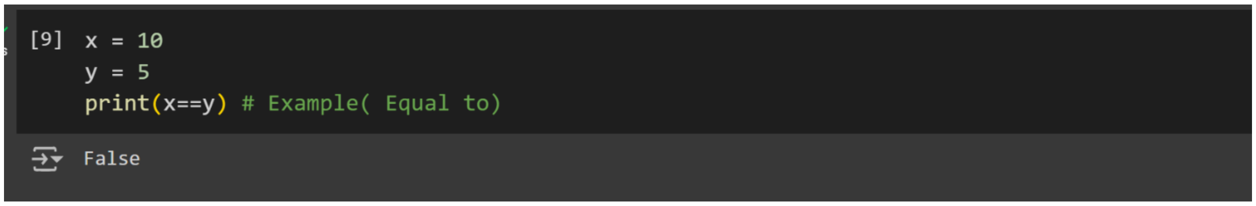


```
[8] x = 10
    y = 5
    print(x/y) # Division Operators

2.0
```

Comparison Operators

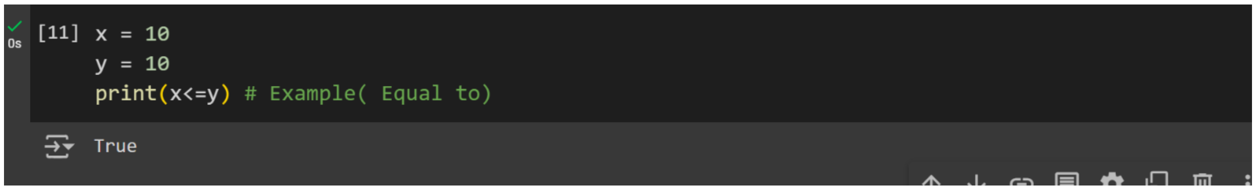
Operator	Name	Example
==	Equal to	X==y
!=	Not equal to	X!=y
<	Greater than	X<y
>	Less Than	x>y
<=	Greater than or equal to	X<=y
>=	Less than or equal to	x>=y

Example (Equal to)



```
[9] x = 10
    y = 5
    print(x==y) # Example( Equal to)

False
```

Example (Greater than or equal to)



```
[11] x = 10
     y = 10
     print(x<=y) # Example( Equal to)

True
```

Logical Operators

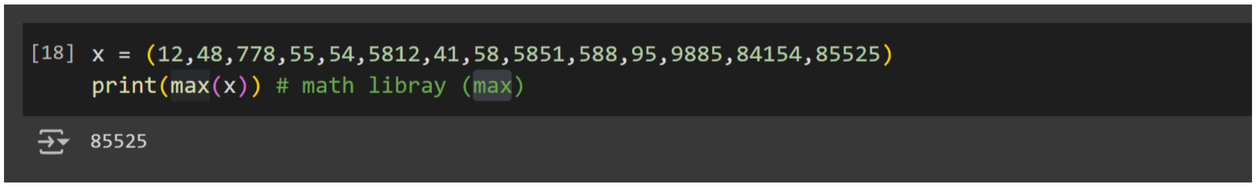
Operators	Condition
and	True if both conditions are true
or	True if at least one condition is true
not	makes true conditions false and vice versa

Python Math Library Function

Math Lib	Condition
max	Returns the largest value
min	Returns the smallest value
floor	Rounds a number down to the nearest integer
ceil	Rounds a number up to the nearest integer
sqrt	Returns the square root of a number
round	Rounds a number to the nearest integer or decimal place
abs	Returns the absolute (non-negative) value of a number

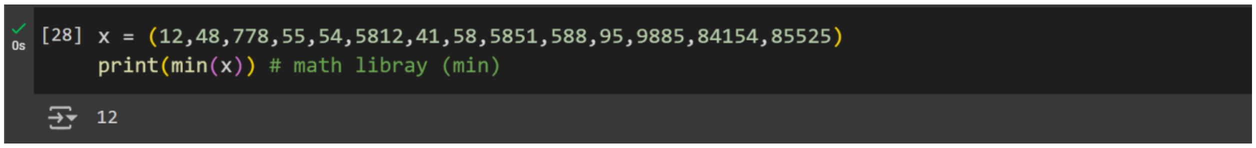
Example

max



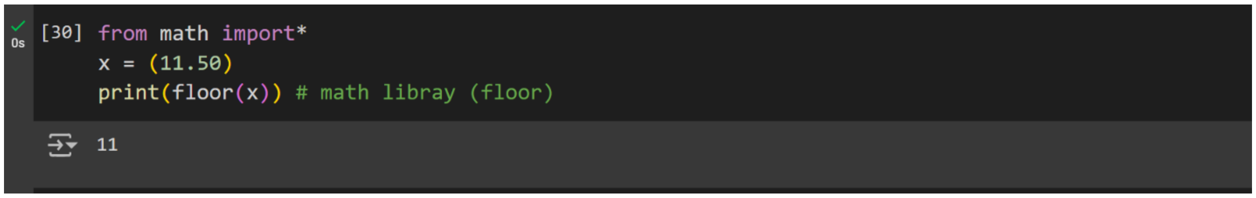
```
[18] x = (12,48,778,55,54,5812,41,58,5851,588,95,9885,84154,85525)
     print(max(x)) # math libray (max)

85525
```

min



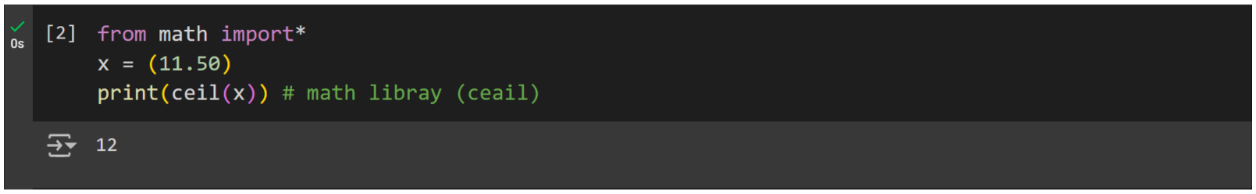
```
[28] x = (12,48,778,55,54,5812,41,58,5851,588,95,9885,84154,85525)
     print(min(x)) # math libray (min)

12
```

Floor



```
[30] from math import*
     x = (11.50)
     print(floor(x)) # math libray (floor)

11
```

Ceil



```
[2] from math import*
    x = (11.50)
    print(ceil(x)) # math libray (ceail)

12
```

if-else in python

if-else

if-else statement in Python is used for decision-making.

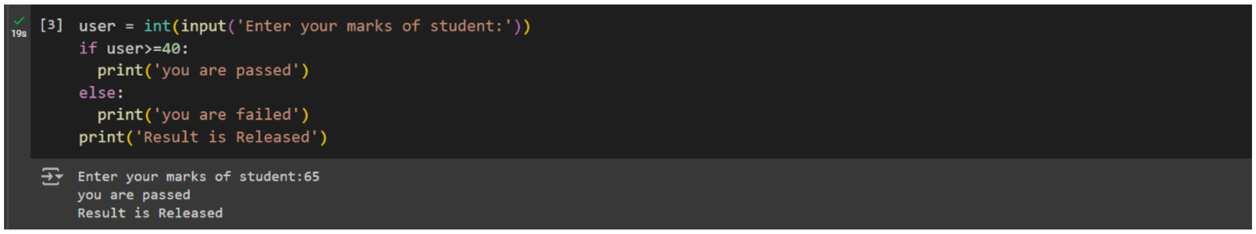
if expression:

  statement ()

else:

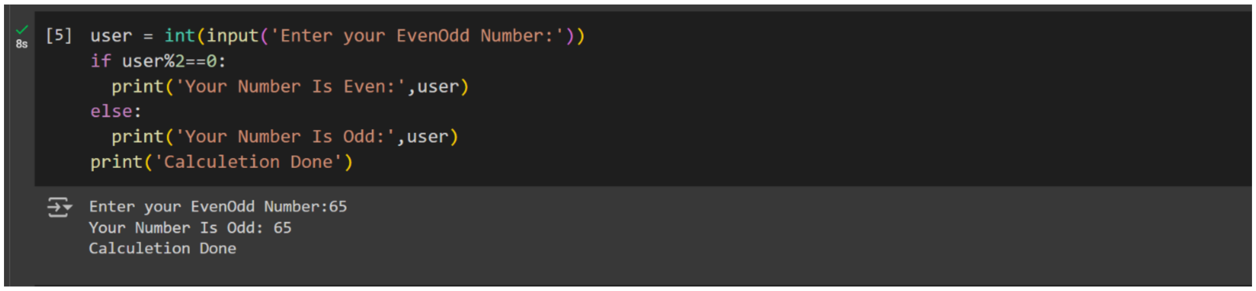
 statement ()

Example 1



```
[3] user = int(input('Enter your marks of student:'))
    if user>=40:
      print('you are passed')
    else:
      print('you are failed')
    print('Result is Released')

Enter your marks of student:65
you are passed
Result is Released
```

Example 2 (Calculate EvenOdd Number)



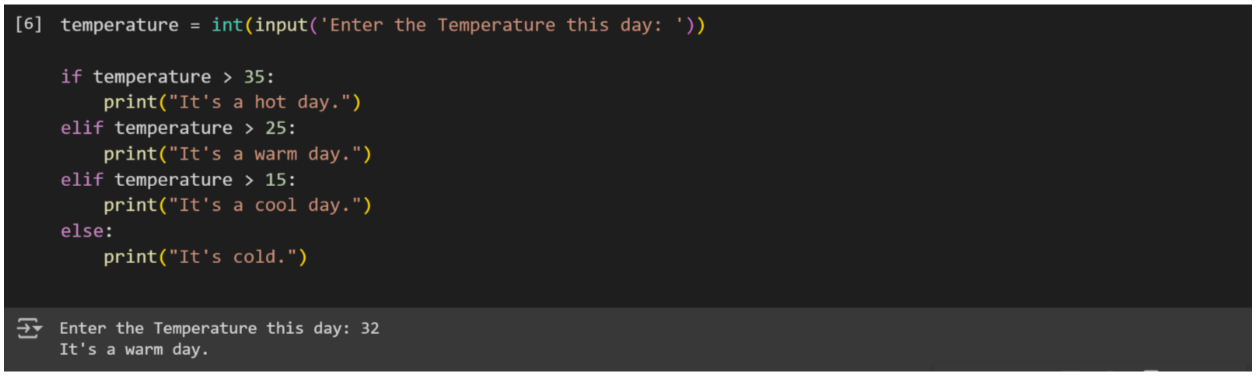
```
[5] user = int(input('Enter your EvenOdd Number:'))
    if user%2==0:
      print('Your Number Is Even:',user)
    else:
      print('Your Number Is Odd:',user)
    print('Calculetion Done')
```

```
Enter your EvenOdd Number:65
Your Number Is Odd: 65
Calculetion Done
```

elif in Python

elif statement in Python is short for "else if" and is used in decision making to test multiple conditions.

Example

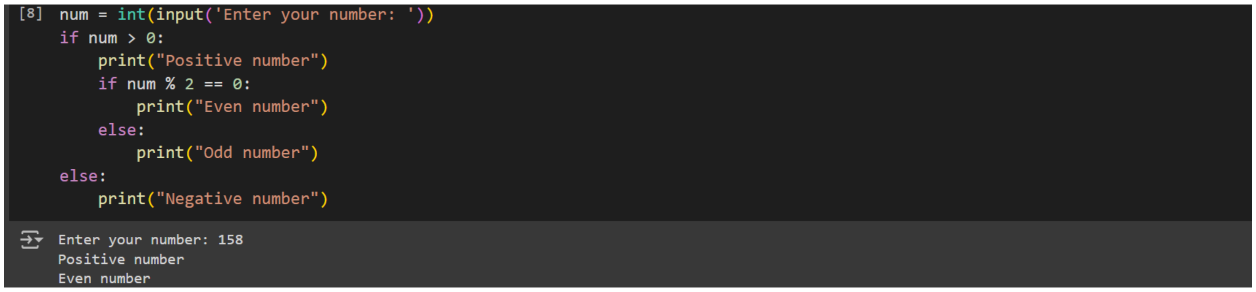


```
[6] temperature = int(input('Enter the Temperature this day: '))

    if temperature > 35:
        print("It's a hot day.")
    elif temperature > 25:
        print("It's a warm day.")
    elif temperature > 15:
        print("It's a cool day.")
    else:
        print("It's cold.")
```

```
Enter the Temperature this day: 32
It's a warm day.
```

Nested-if in Python

Nested if statement in Python is when you place one if statement inside another if (elif or else) statement.



```
[8] num = int(input('Enter your number: '))
    if num > 0:
        print("Positive number")
        if num % 2 == 0:
            print("Even number")
        else:
            print("Odd number")
    else:
        print("Negative number")
```

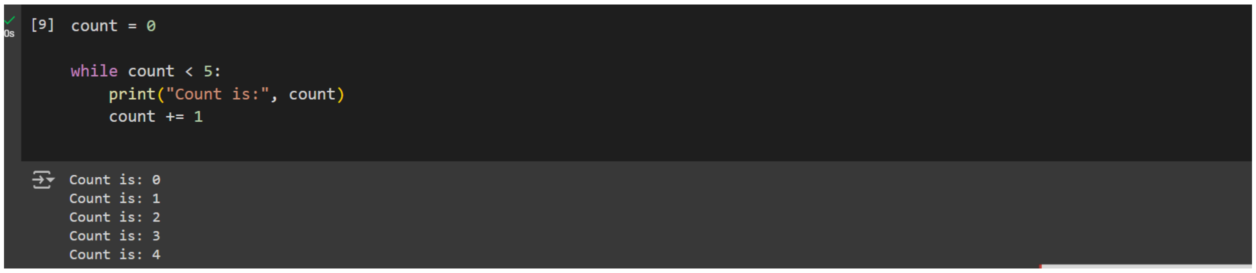
```
Enter your number: 158
Positive number
Even number
```

Loop in python

While loop in Python

Python while loop is a control flow statement that repeatedly executes a block of code as long as a given condition is true.

Example

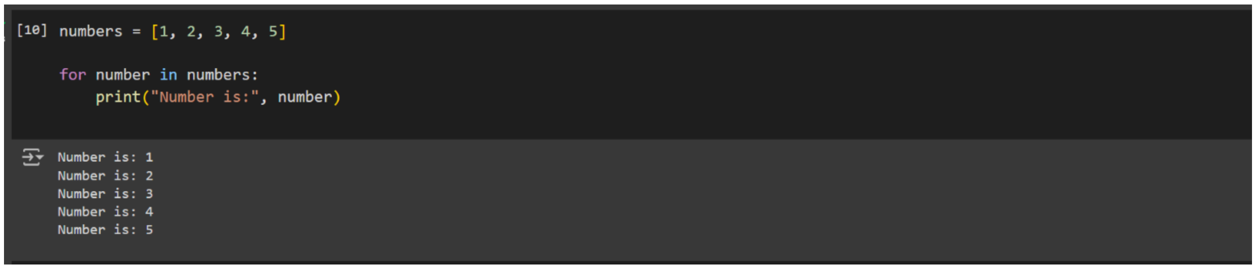


```
[9] count = 0

    while count < 5:
        print("Count is:", count)
        count += 1

Count is: 0
Count is: 1
Count is: 2
Count is: 3
Count is: 4
```

For loop in Python

Python for loop is another control flow statement that iterates over a sequence and executes a block of code for each element in the sequence.

Example

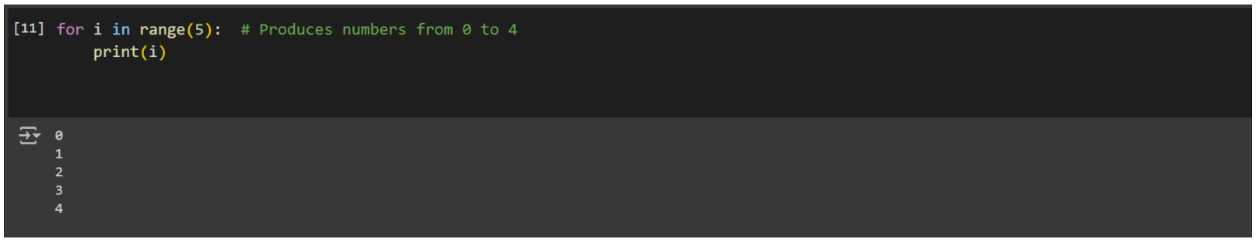


```
[10] numbers = [1, 2, 3, 4, 5]

     for number in numbers:
         print("Number is:", number)

Number is: 1
Number is: 2
Number is: 3
Number is: 4
Number is: 5
```

Range in Python

The range () function in Python generates a sequence of numbers.

Example



```
[11] for i in range(5):  # Produces numbers from 0 to 4
         print(i)

0
1
2
3
4
```

Python break and continue

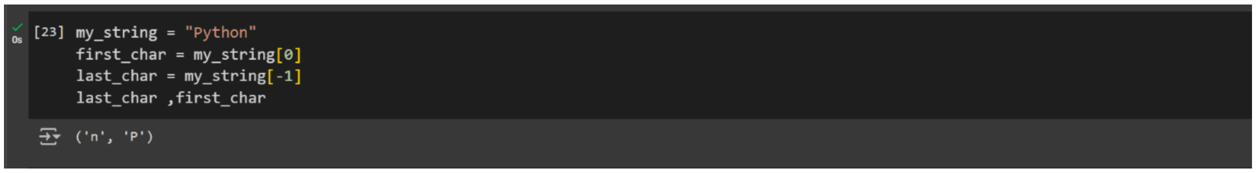
Python break and continue control flow statements that alter the execution of loops.

More Basic of Python

String In Python

Python String is a sequence of characters.

Example

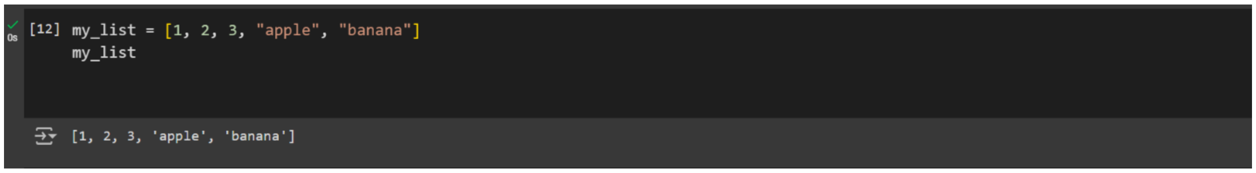


```
[23] my_string = "Python"
     first_char = my_string[0]
     last_char = my_string[-1]
     last_char ,first_char

('n', 'P')
```

List In Python

Python list is a mutable, ordered collection of items.

Example

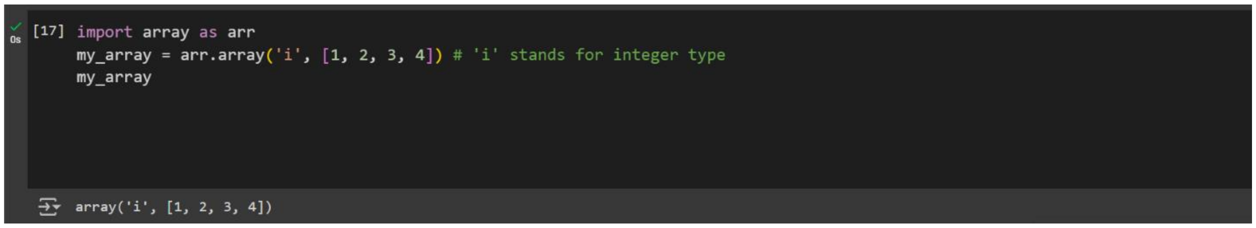


```
[12] my_list = [1, 2, 3, "apple", "banana"]
     my_list

[1, 2, 3, 'apple', 'banana']
```

Array In Python

Python array is similar to a list, but typically used with more specific numeric types.

Example

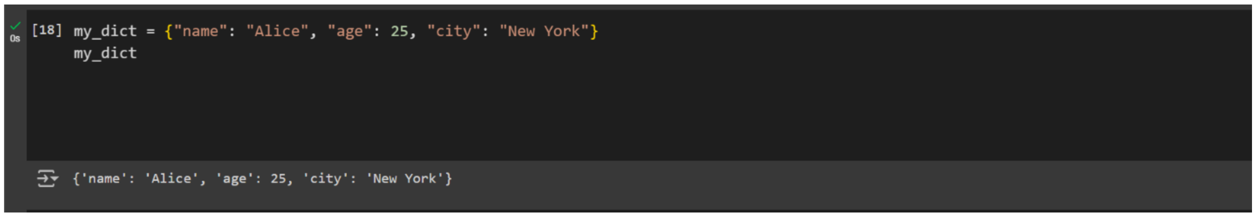


```
[17] import array as arr
     my_array = arr.array('i', [1, 2, 3, 4]) # 'i' stands for integer type
     my_array

array('i', [1, 2, 3, 4])
```

Dictionary In Python

Python dictionary is an unordered collection of key-value pairs.

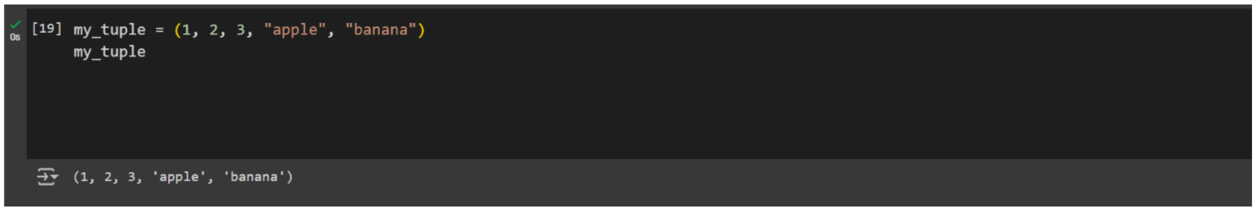
Example



```
[18] my_dict = {"name": "Alice", "age": 25, "city": "New York"}
     my_dict

{'name': 'Alice', 'age': 25, 'city': 'New York'}
```

Tuple In Python

Python tuple is an immutable, ordered collection of items.



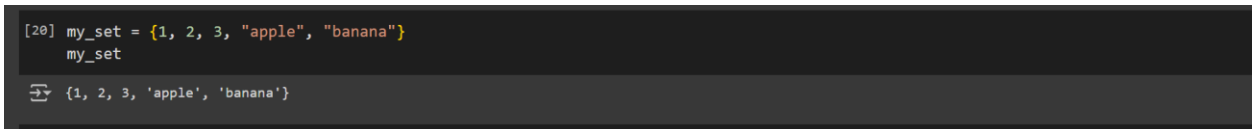
```
[19] my_tuple = (1, 2, 3, "apple", "banana")
     my_tuple

(1, 2, 3, 'apple', 'banana')
```

Example

Set In Python

Python set is an unordered collection of unique items.

Example



```
[20] my_set = {1, 2, 3, "apple", "banana"}
     my_set

{1, 2, 3, 'apple', 'banana'}
```

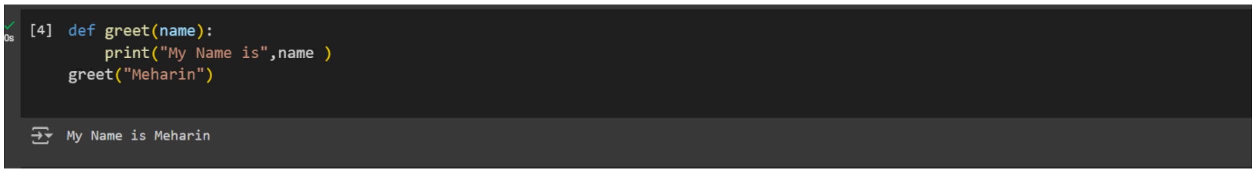
Python-Basic to Advanced

Function in Python

Def function In Python

In Python def is used to define a function. Def Functions help modularize code, improving readability and reusability.

Example

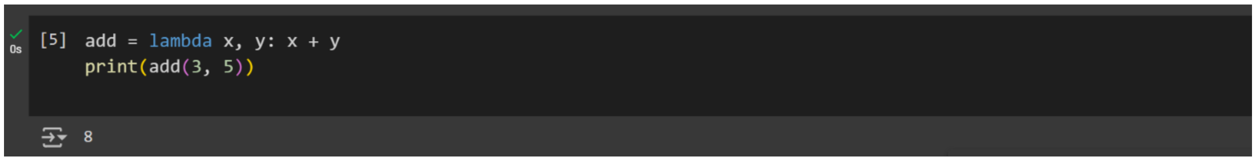


```
[4] def greet(name):
        print("My Name is",name )
    greet("Meharin")

My Name is Meharin
```

Lambda function In Python

Lambda function to define the anonymous function.

Example

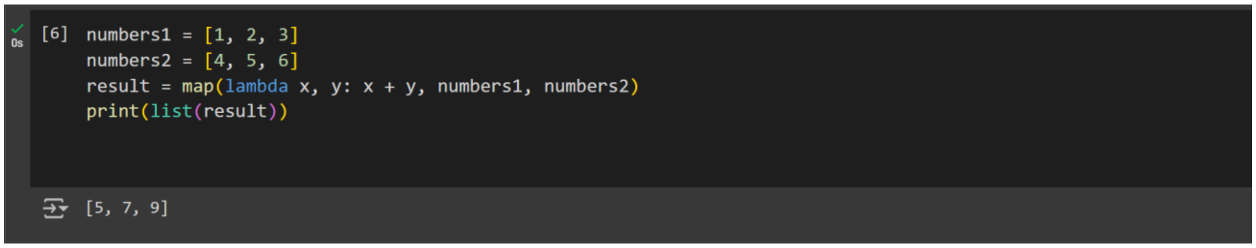


```
[5] add = lambda x, y: x + y
    print(add(3, 5))

8
```

Map function In Python

Map function is used to apply a given function to every item in an Repeatable and return a map object.

Example



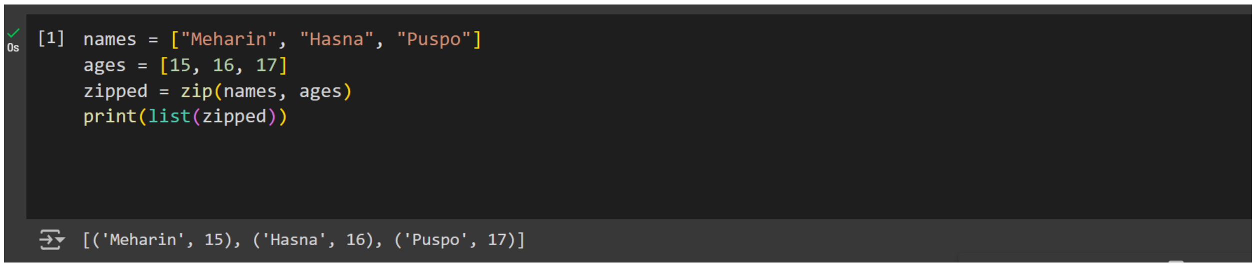
```
[6] numbers1 = [1, 2, 3]
    numbers2 = [4, 5, 6]
    result = map(lambda x, y: x + y, numbers1, numbers2)
    print(list(result))

[5, 7, 9]
```

Written By Meharin Hasna Puspo

Zip function In Python

zip function in Python takes multiple Repeatables and aggregates them into tuples, pairing elements from each Repeatable at the same index.

Example



```
[1] names = ["Meharin", "Hasna", "Puspo"]
    ages = [15, 16, 17]
    zipped = zip(names, ages)
    print(list(zipped))
```

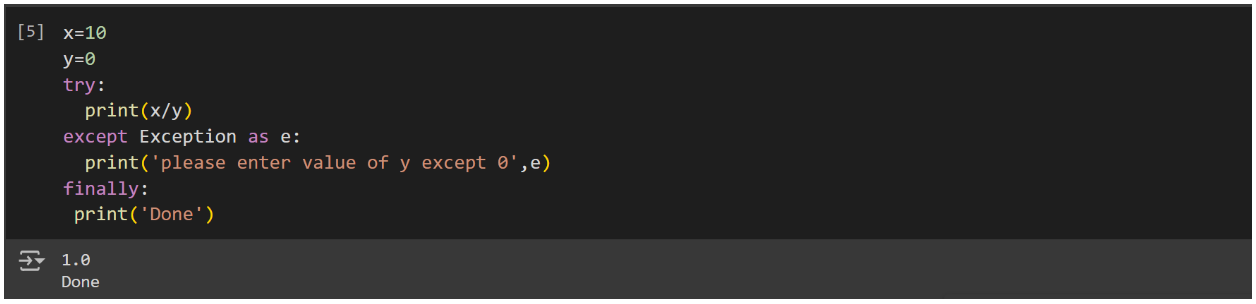
```
[('Meharin', 15), ('Hasna', 16), ('Puspo', 17)]
```

Advanced Topic in Python

Exception Handling In Python

Exception handling in Python allows you to handle errors gracefully and avoid program crashes.

Example



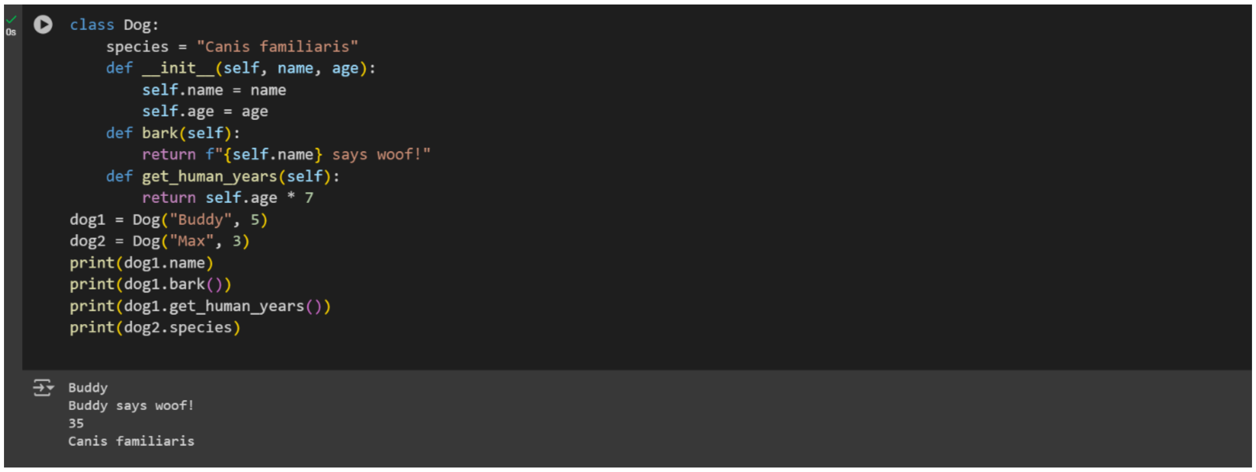
```
[5] x=10
    y=0
    try:
      print(x/y)
    except Exception as e:
      print('please enter value of y except 0',e)
    finally:
     print('Done')
```

```
1.0
Done
```

Class & Object In Python

Python classes are blueprints for creating objects (instances), which encapsulate both data (properties) and behavior (methods).

Example



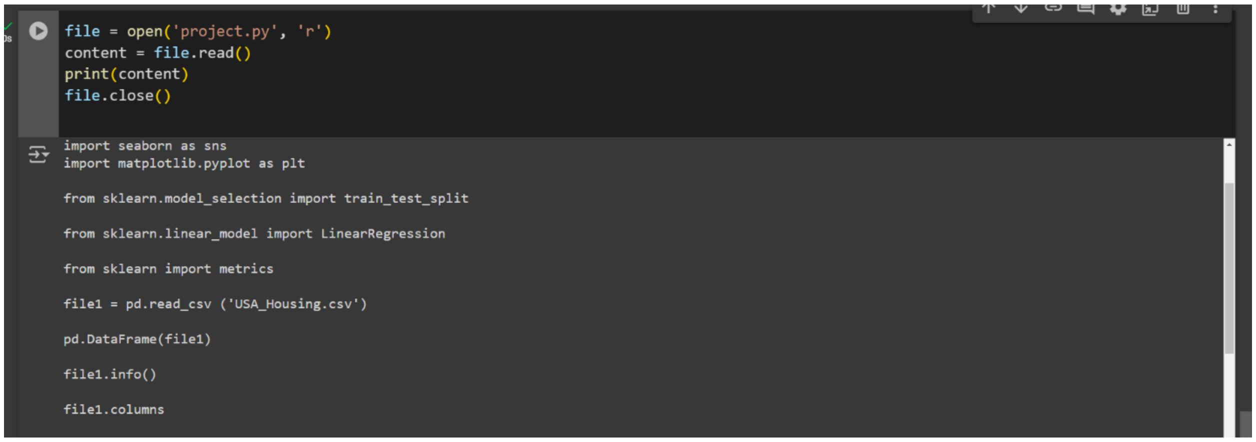
```
class Dog:
    species = "Canis familiaris"
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def bark(self):
        return f"{self.name} says woof!"
    def get_human_years(self):
        return self.age * 7
dog1 = Dog("Buddy", 5)
dog2 = Dog("Max", 3)
print(dog1.name)
print(dog1.bark())
print(dog1.get_human_years())
print(dog2.species)
```

```
Buddy
Buddy says woof!
35
Canis familiaris
```

File In Python

Python file handling allows you to read from and write to files on your filesystem.

Example



```
file = open('project.py', 'r')
content = file.read()
print(content)
file.close()
```

```
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn import metrics

file1 = pd.read_csv ('USA_Housing.csv')

pd.DataFrame(file1)

file1.info()

file1.columns
```

Python has some advanced libraries. For example:

Pandas Library

Numpy Library

Matplotlib Library

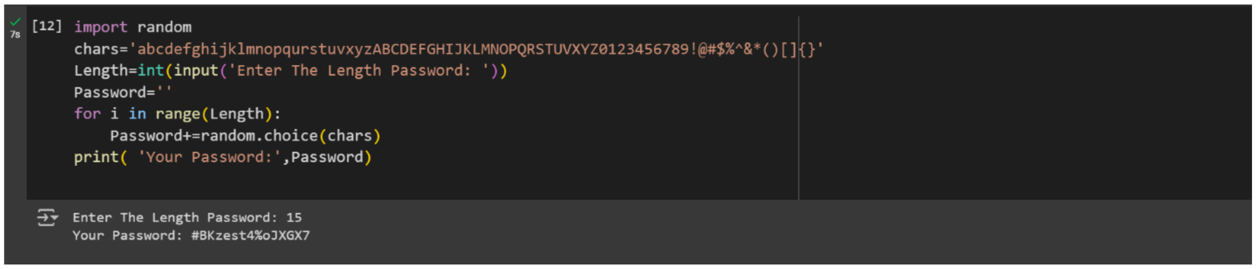
Seaborn Library ,etc

These libraries are especially used in data science, data analysis, machine learning.

Python-Mini Project

Random Password Generate

```
import random

chars='abcdefghijklmnopqurstuvxyzABCDEFGHIJKLMNOPQRSTUVXYZ0123456789!@#$%^&*()[]{}'

Length=int(input('Enter The Length Password: '))

Password=''

for i in range(Length):

    Password+=random.choice(chars)

print( 'Your Password:',Password)
```



```
[12] import random
     chars='abcdefghijklmnopqurstuvxyzABCDEFGHIJKLMNOPQRSTUVXYZ0123456789!@#$%^&*()[]{}'
     Length=int(input('Enter The Length Password: '))
     Password=''
     for i in range(Length):
         Password+=random.choice(chars)
     print( 'Your Password:',Password)
```

```
Enter The Length Password: 15
Your Password: #BKzest4%oJXGX7
```

Author identity

Nmae: Meharin Hasna Puspo

Studying Diploma in Computer Science Engineering

Python |Data Science | Machine learning