Python

UTOMO BUDIYANTO

Data Types

In computer programming, **data type is a classification of data** which tells the compiler or interpreter how the programmer intends to use data

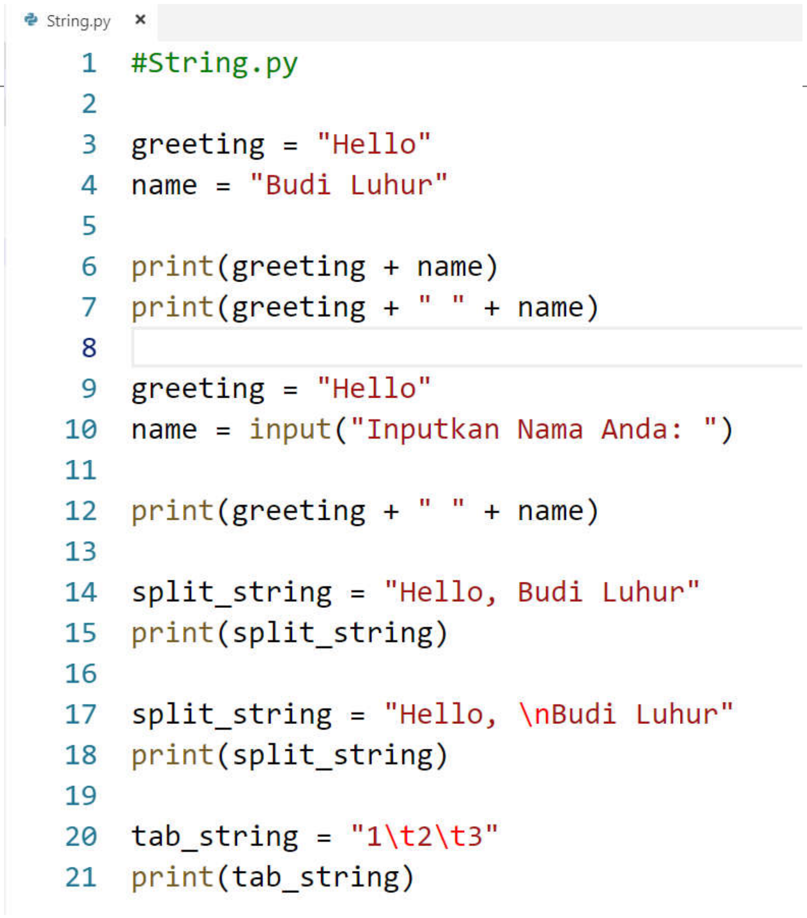
Data type is an attribute that tells what kind of data that a value can have

Every value in Python has a data type. **Everything is an object in Python programming**, data types are actually classes and variables are objects of those classes

Data Types in Python

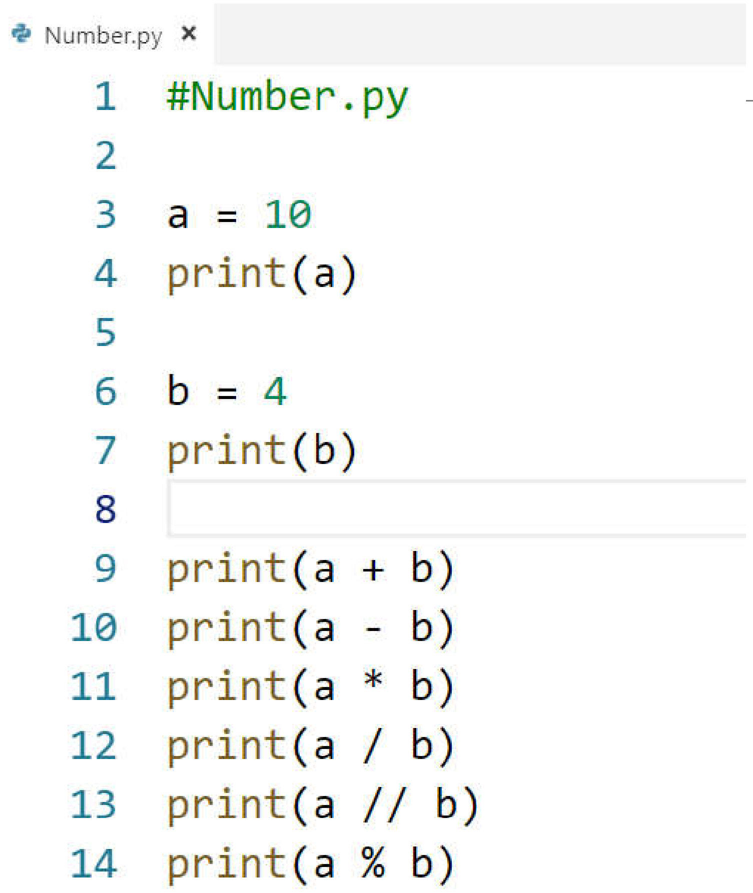
Type	Description
Numbers	Whole numbers ex. 19, 100, 200 Decimal point numbers ex 19.79, 1.23, 20.21
Strings	Ordered sequence of characters Ex. “Budi”, “1979”, “Jakarta”
Lists	Ordered sequence of objects Ex. [“Luhur”, 2021, 19.79]
Dictionaries	Unordered key value pairs Ex. {“nim”:”1911500123”, “nama”:”Budi”}
Tuples	Ordered immutable sequence of objects Ex. (“Luhur”, 2021, 19.79)
Sets	Unordered collection of unique objects Ex. (“a”,”b”,”c”)
Booleans	Logical values (True or False)

Strings



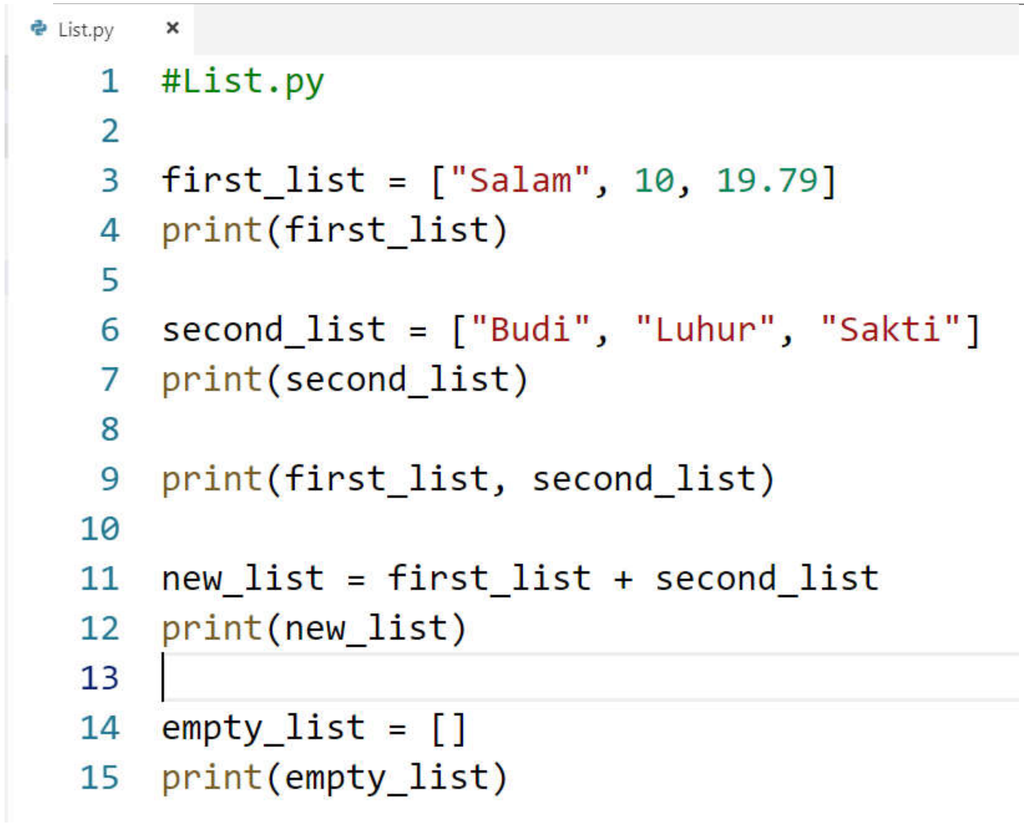
```python
#String.py

greeting = "Hello"
name = "Budi Luhur"

print(greeting + name)
print(greeting + " " + name)

greeting = "Hello"
name = input("Inputkan Nama Anda: ")

print(greeting + " " + name)

split_string = "Hello, Budi Luhur"
print(split_string)

split_string = "Hello, \nBudi Luhur"
print(split_string)

tab_string = "1\t2\t3"
print(tab_string)
```

Numbers


Number.py ×

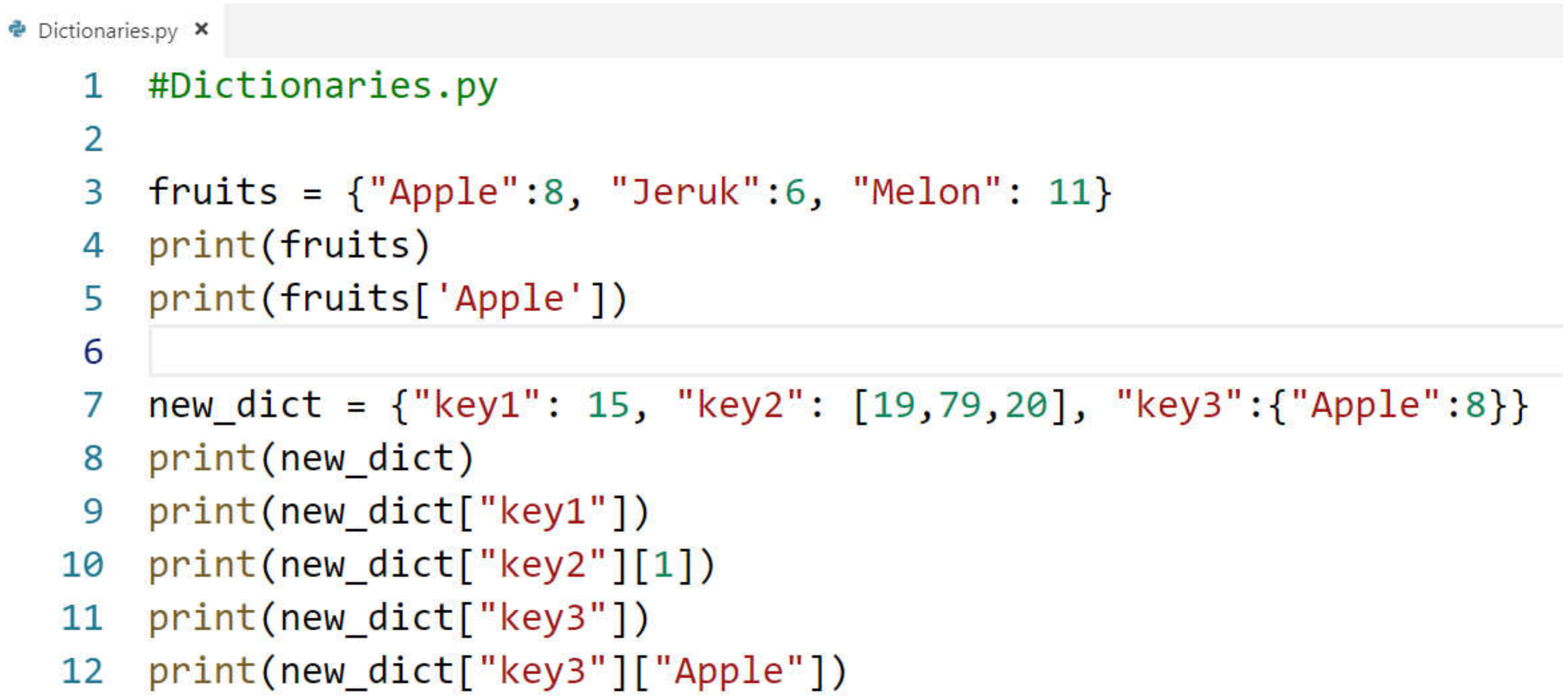
```
#Number.py

a = 10
print(a)

b = 4
print(b)

print(a + b)
print(a - b)
print(a * b)
print(a / b)
print(a // b)
print(a % b)
```

List



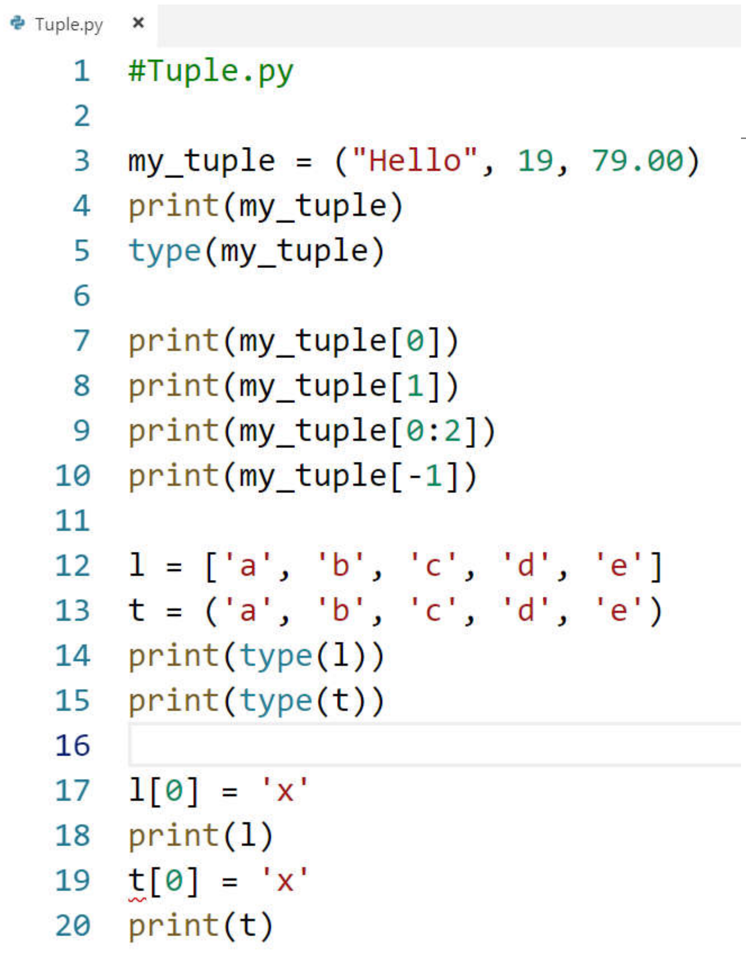
```python
#List.py

first_list = ["Salam", 10, 19.79]
print(first_list)

second_list = ["Budi", "Luhur", "Sakti"]
print(second_list)

print(first_list, second_list)

new_list = first_list + second_list
print(new_list)

empty_list = []
print(empty_list)
```

Dictionaries



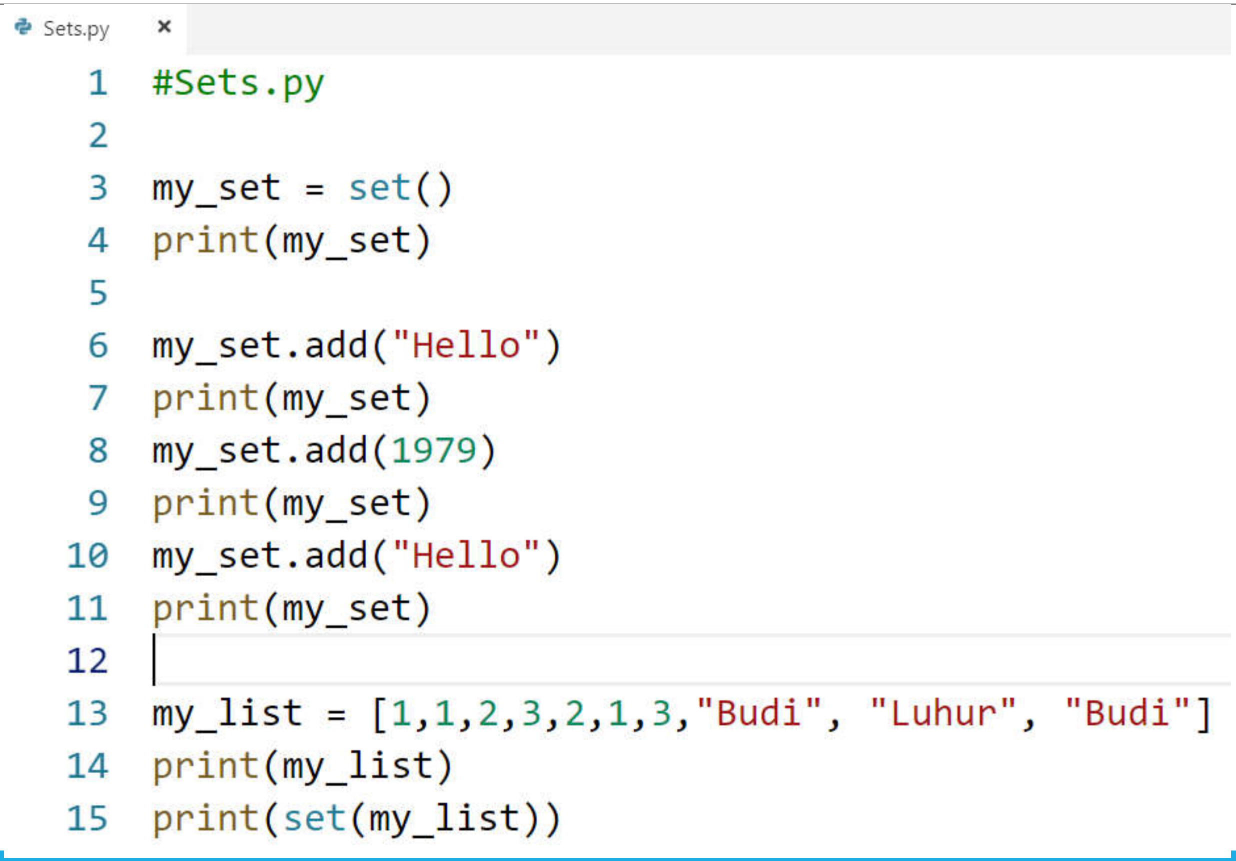
```python
#Dictionaries.py

fruits = {"Apple":8, "Jeruk":6, "Melon": 11}
print(fruits)
print(fruits['Apple'])

new_dict = {"key1": 15, "key2": [19,79,20], "key3":{"Apple":8}}
print(new_dict)
print(new_dict["key1"])
print(new_dict["key2"][1])
print(new_dict["key3"])
print(new_dict["key3"]["Apple"])
```

Tuple



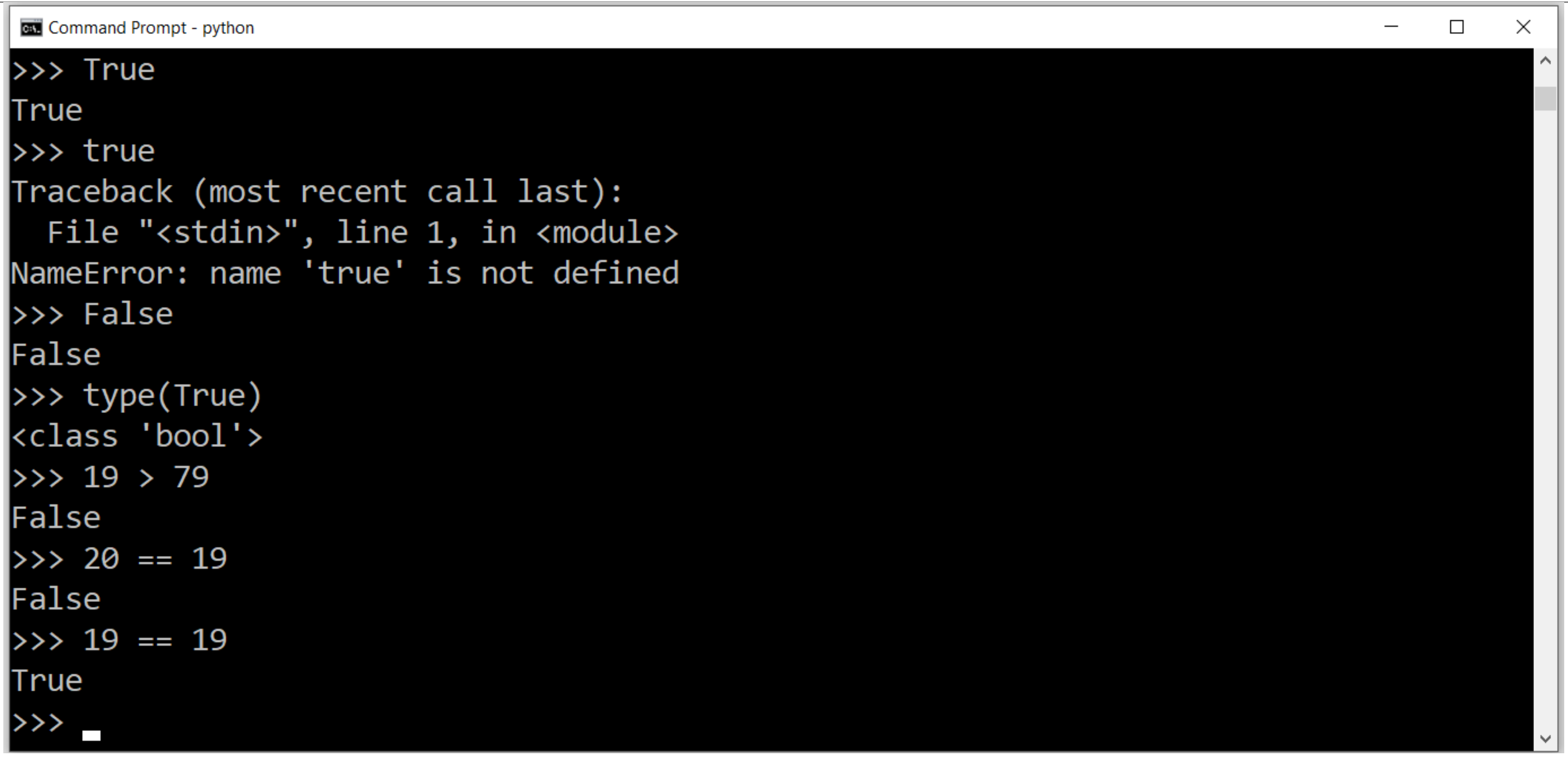
```python
#Tuple.py

my_tuple = ("Hello", 19, 79.00)
print(my_tuple)
type(my_tuple)

print(my_tuple[0])
print(my_tuple[1])
print(my_tuple[0:2])
print(my_tuple[-1])

l = ['a', 'b', 'c', 'd', 'e']
t = ('a', 'b', 'c', 'd', 'e')
print(type(l))
print(type(t))

l[0] = 'x'
print(l)
t[0] = 'x'
print(t)
```

Sets



```
#Sets.py

my_set = set()
print(my_set)

my_set.add("Hello")
print(my_set)
my_set.add(1979)
print(my_set)
my_set.add("Hello")
print(my_set)

my_list = [1,1,2,3,2,1,3,"Budi", "Luhur", "Budi"]
print(my_list)
print(set(my_list))
```

Boolean

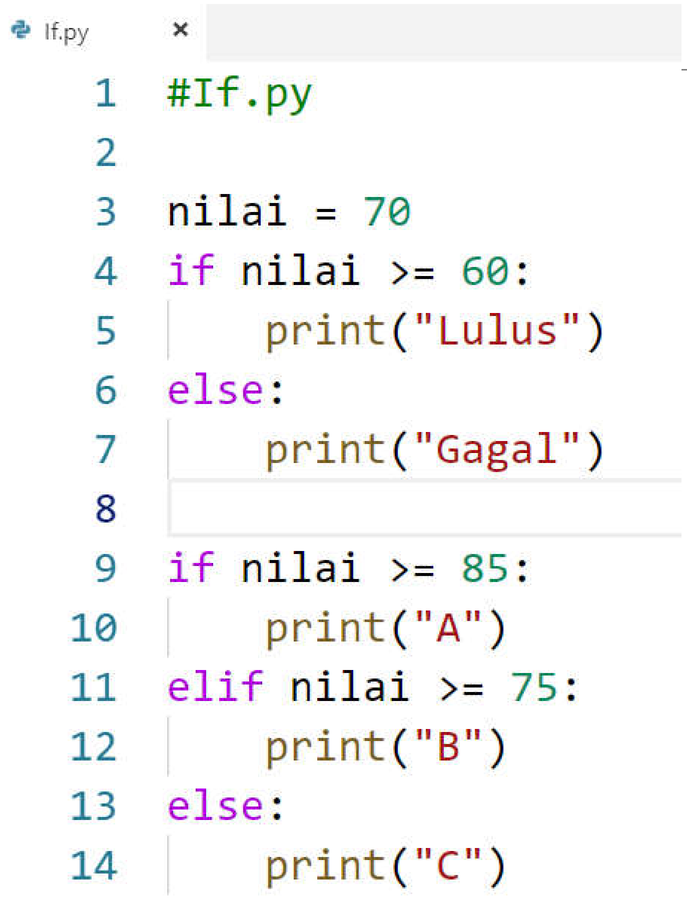


```
>>> True
True
>>> true
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'true' is not defined
>>> False
False
>>> type(True)
<class 'bool'>
>>> 19 > 79
False
>>> 20 == 19
False
>>> 19 == 19
True
>>>
```

If, Else, Elif

Syntax

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



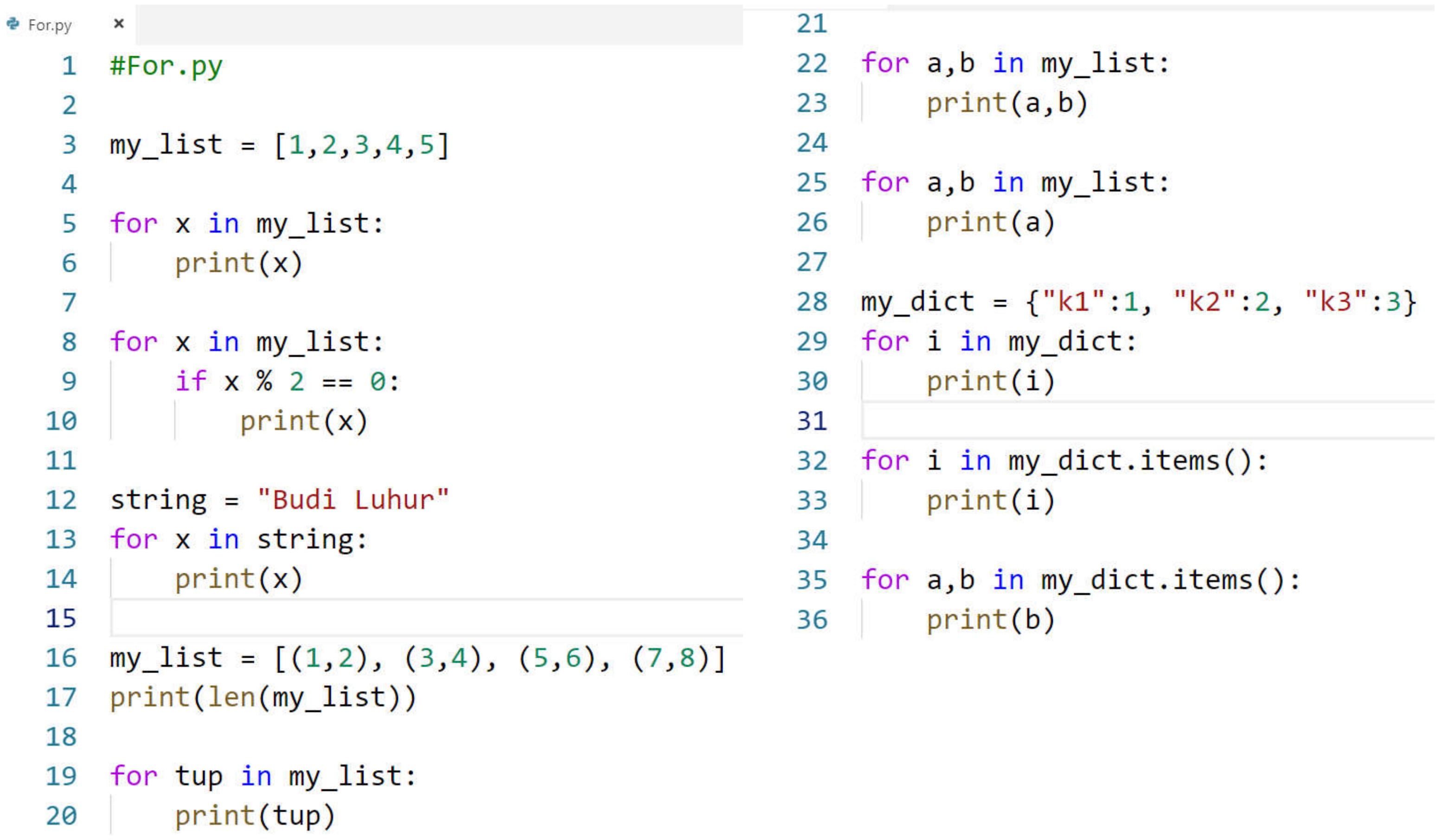
```python
#If.py

nilai = 70
if nilai >= 60:
    print("Lulus")
else:
    print("Gagal")

if nilai >= 85:
    print("A")
elif nilai >= 75:
    print("B")
else:
    print("C")
```

For Loop

Syntax:

for var in sequence:

statement(s)



For.py ×

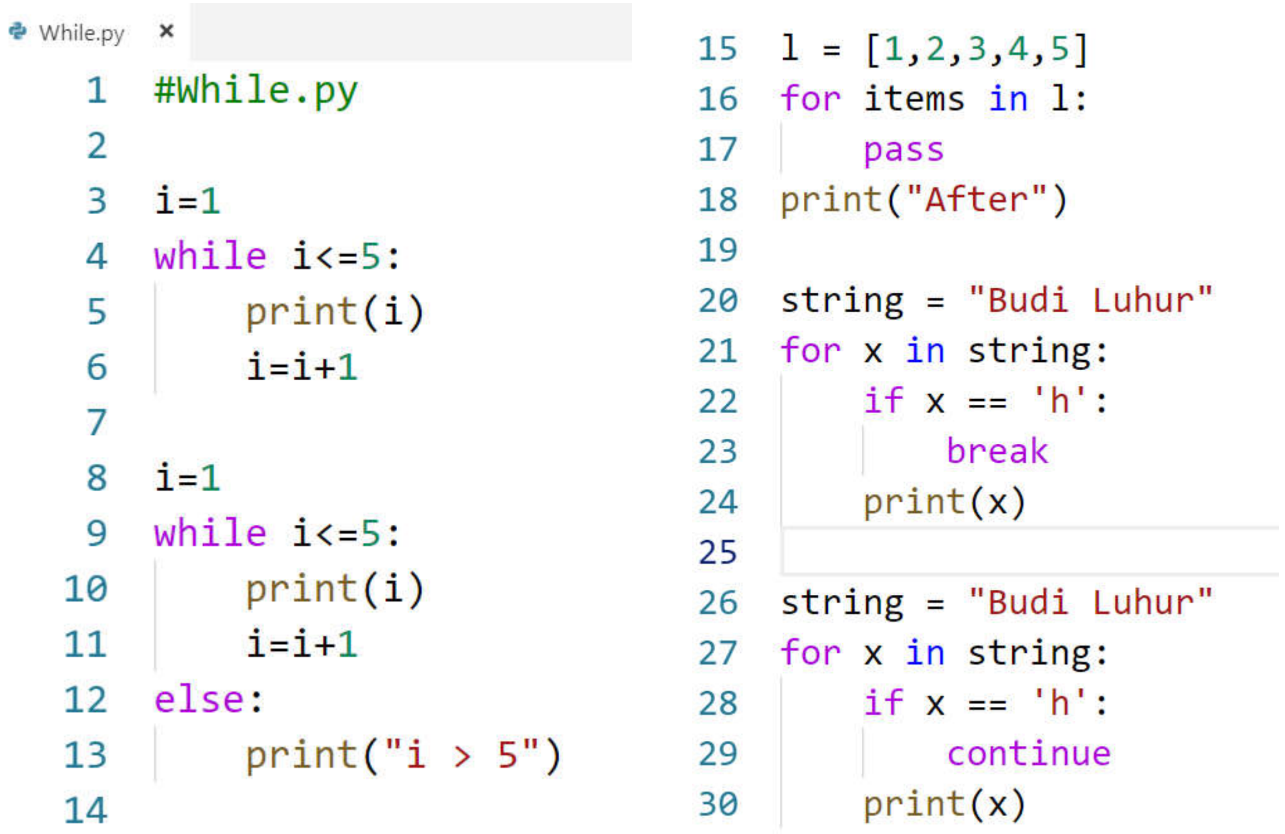
```python
#For.py

my_list = [1,2,3,4,5]

for x in my_list:
    print(x)

for x in my_list:
    if x % 2 == 0:
        print(x)

string = "Budi Luhur"
for x in string:
    print(x)

my_list = [(1,2), (3,4), (5,6), (7,8)]
print(len(my_list))

for tup in my_list:
    print(tup)

for a,b in my_list:
    print(a,b)

for a,b in my_list:
    print(a)

my_dict = {"k1":1, "k2":2, "k3":3}
for i in my_dict:
    print(i)

for i in my_dict.items():
    print(i)

for a,b in my_dict.items():
    print(b)
```

While Loop

Syntax:

While expression:

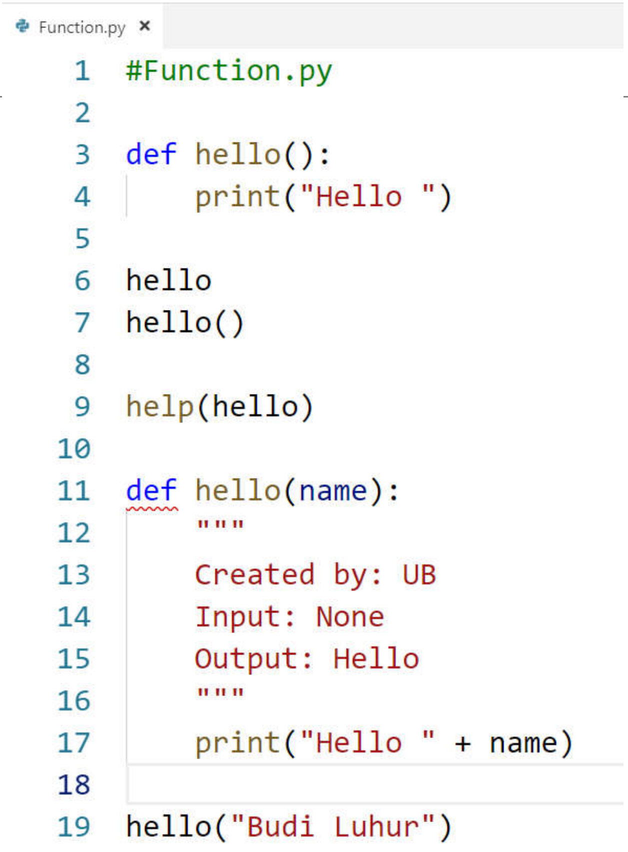
statement(s)



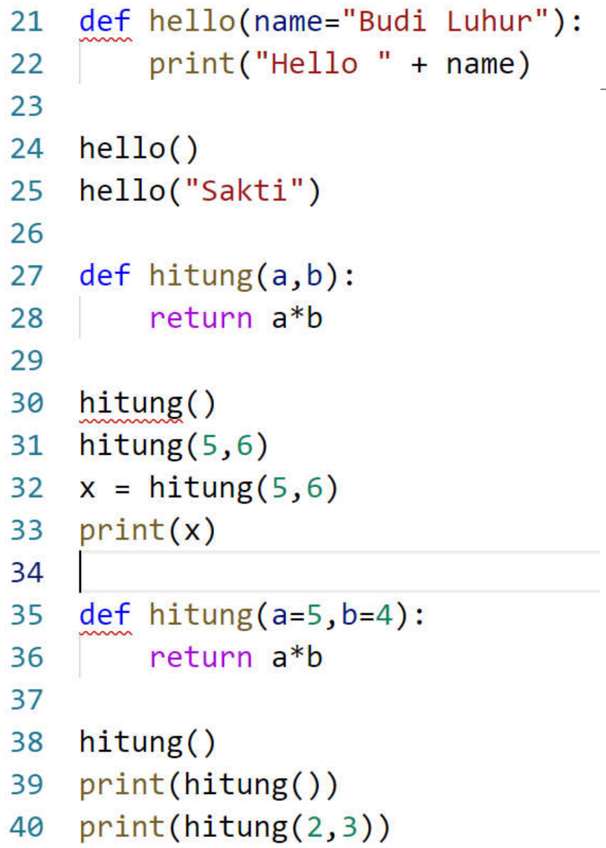
While.py

```python
#While.py

i=1
while i<=5:
    print(i)
    i=i+1

i=1
while i<=5:
    print(i)
    i=i+1
else:
    print("i > 5")

l = [1,2,3,4,5]
for items in l:
    pass
print("After")

string = "Budi Luhur"
for x in string:
    if x == 'h':
        break
    print(x)

string = "Budi Luhur"
for x in string:
    if x == 'h':
        continue
    print(x)
```

Function



```python
#Function.py

def hello():
    print("Hello ")

hello
hello()

help(hello)

def hello(name):
    """
    Created by: UB
    Input: None
    Output: Hello
    """
    print("Hello " + name)

hello("Budi Luhur")
```



```python
def hello(name="Budi Luhur"):
    print("Hello " + name)

hello()
hello("Sakti")

def hitung(a,b):
    return a*b

hitung()
hitung(5,6)
x = hitung(5,6)
print(x)

def hitung(a=5,b=4):
    return a*b

hitung()
print(hitung())
print(hitung(2,3))
```

Map, Filter, Lambda Expression

Map and Filter

- The map function is the simplest one in the Python built-in functions

- It applies to the inerrables

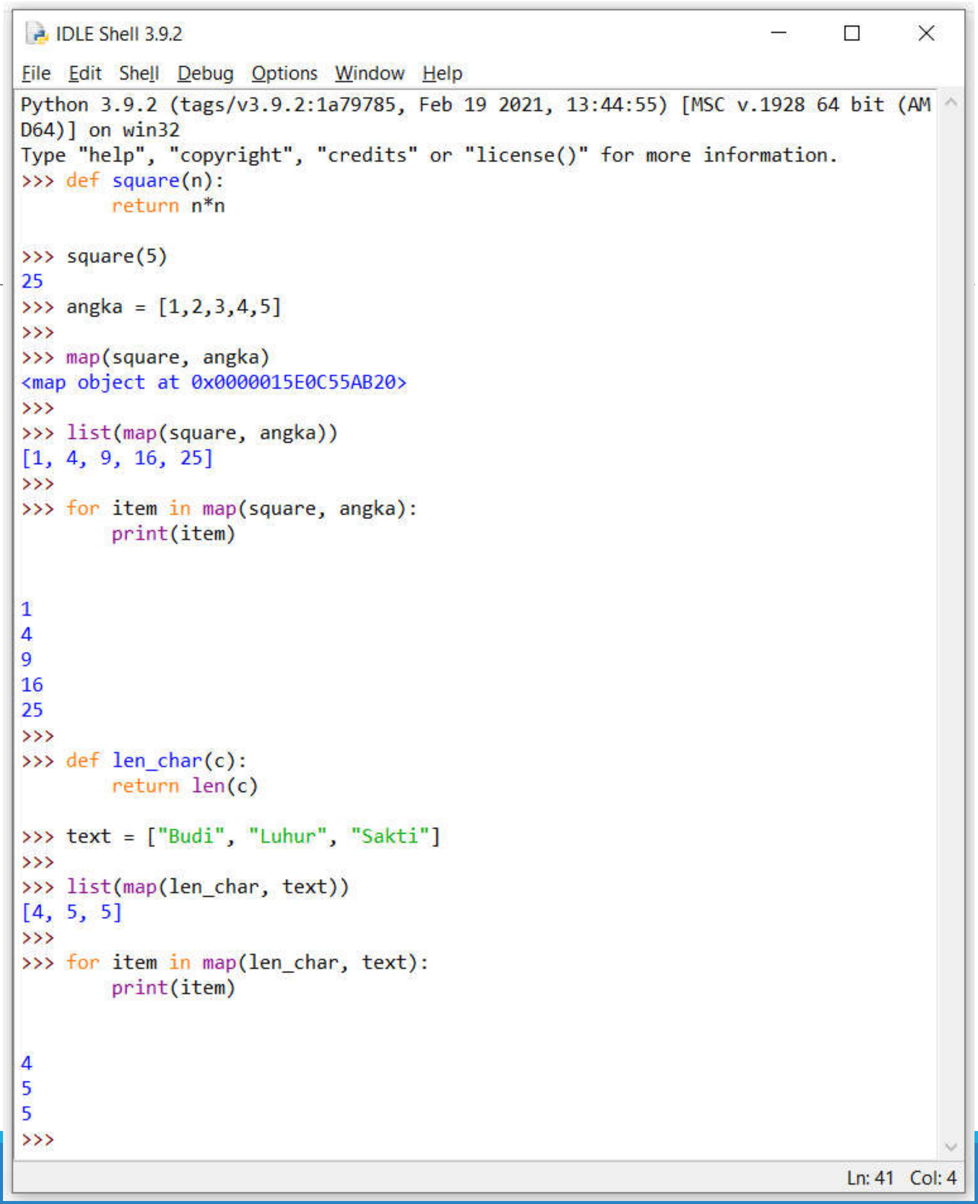
- The filter function filters out items based on a test condition that has been given in the function

Lambda Expression

- used to create small, one time and anonymous function objects in Python

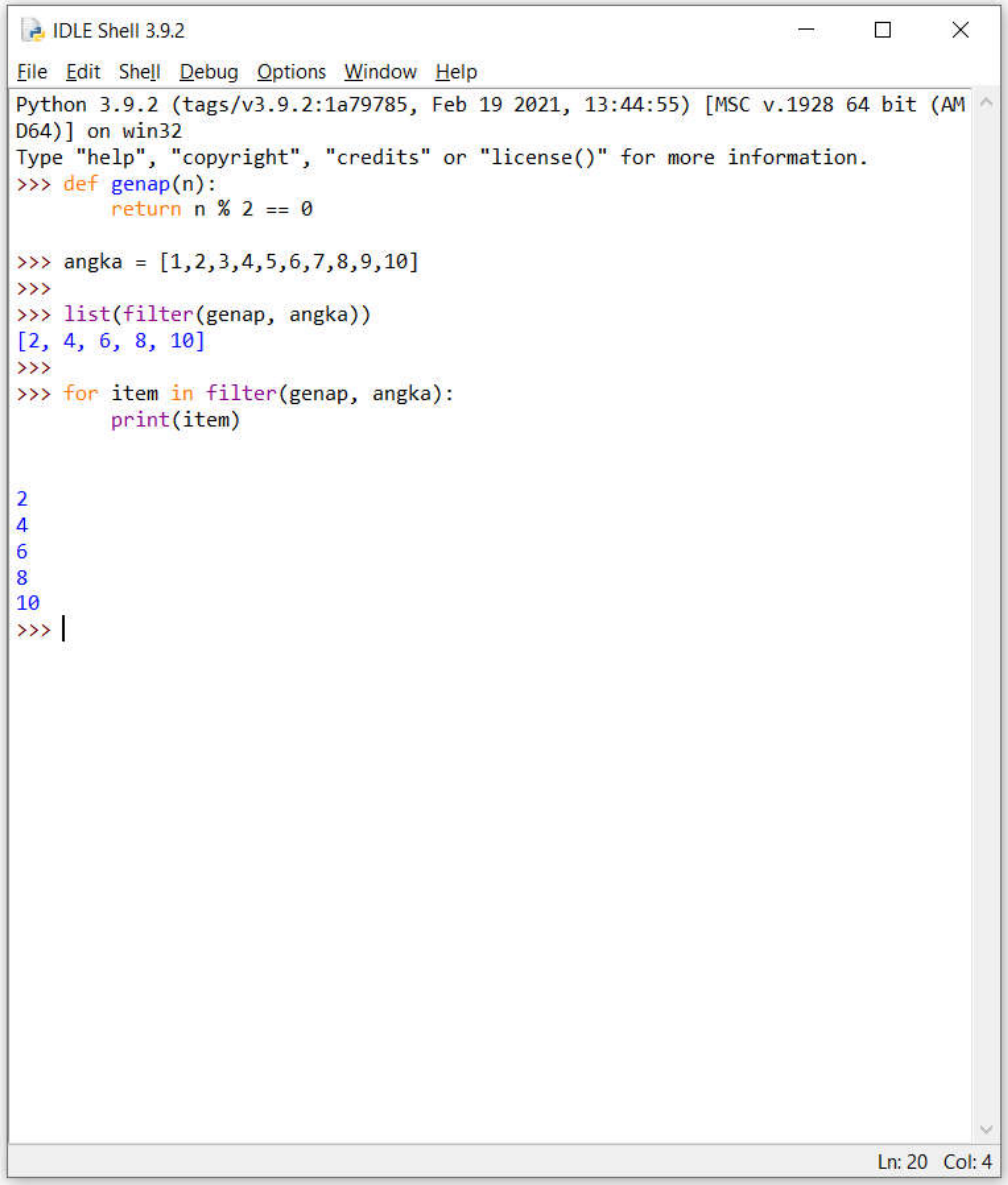
- It can contain any number of arguments, but it can have only one expression

Map Function



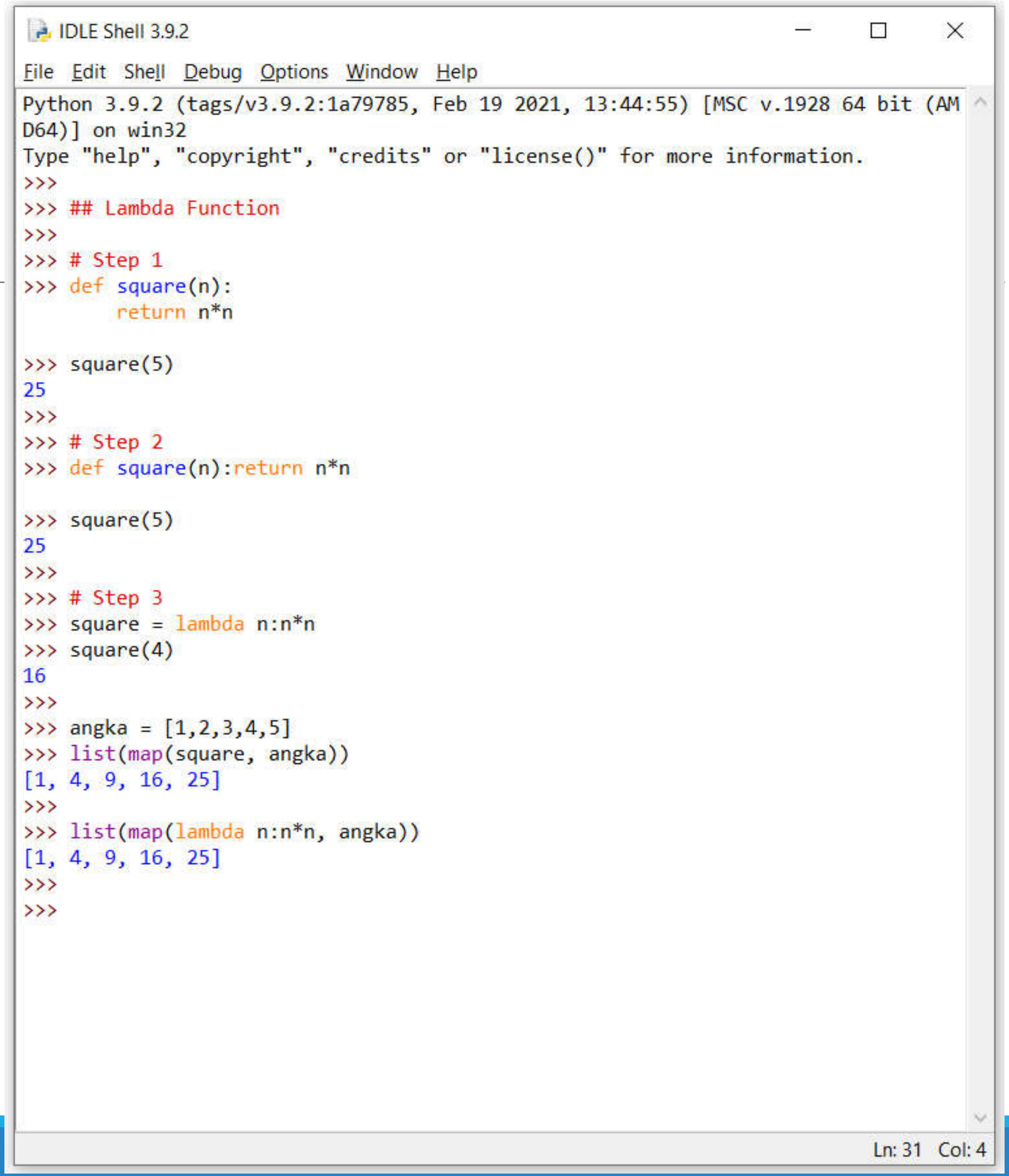
```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def square(n):
        return n*n

>>> square(5)
25
>>> angka = [1,2,3,4,5]
>>>
>>> map(square, angka)
<map object at 0x0000015E0C55AB20>
>>>
>>> list(map(square, angka))
[1, 4, 9, 16, 25]
>>>
>>> for item in map(square, angka):
        print(item)


1
4
9
16
25
>>>
>>> def len_char(c):
        return len(c)

>>> text = ["Budi", "Luhur", "Sakti"]
>>>
>>> list(map(len_char, text))
[4, 5, 5]
>>>
>>> for item in map(len_char, text):
        print(item)


4
5
5
>>>
```

Filter Function



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> def genap(n):
        return n % 2 == 0

>>> angka = [1,2,3,4,5,6,7,8,9,10]
>>>
>>> list(filter(genap, angka))
[2, 4, 6, 8, 10]
>>>
>>> for item in filter(genap, angka):
        print(item)


2
4
6
8
10
>>> 
```

Lambda Function



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
>>> ## Lambda Function
>>>
>>> # Step 1
>>> def square(n):
        return n*n

>>> square(5)
25
>>>
>>> # Step 2
>>> def square(n):return n*n

>>> square(5)
25
>>>
>>> # Step 3
>>> square = lambda n:n*n
>>> square(4)
16
>>>
>>> angka = [1,2,3,4,5]
>>> list(map(square, angka))
[1, 4, 9, 16, 25]
>>>
>>> list(map(lambda n:n*n, angka))
[1, 4, 9, 16, 25]
>>>
>>>
```