2020

Free Python Course

O. Elijah

onlinepythonclass@gmail.com

O. ELIJAH (MSc, Ph.D)

1 INTRODUCTION

Welcome to chapter two of the free python course. In this chapter, we hope to achieve the following objectives.

1. Take a look at the following data types tuple, set and dictionary
2. Understand the basic operations in python

1.1 Review of chapter 1

In chapter one, we looked at variables and rules in creating a variable. We also studied the different data types such as integer, float, complex, Boolean, string and list. Now we shall take a look at the tuple, range, set and dictionary

1.2 Date type

We shall look at the following data types and see how to apply them in programming.

Tuple

A tuple is a collection which is ordered and unchangeable. In Python tuples are written with round brackets.

Create a tuple

```
fruits = ("apple", "banana", "cherry")
print(fruit)
Out: ('apple', 'banana', 'cherry')
```

Create a tuple with one item

Note: One item tuple, remember to add the comma:

```
thistuple = ("apple",)
print(type(thistuple))
Out: <class 'tuple'>
```

Index a tuple

```
fruits = ("apple", "banana", "cherry")
print(fruits[1])          # print out the first item in the tuple
Out: banana
```

Change a tuple value

Note you cannot directly change the item in a tuple. You need to convert to list first before you can change

Example change the banana to kiwi

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)
print(x)
Out: ("apple", "kiwi", "cherry")
```

Add Items to tuple

Note: Once a tuple is created, you cannot add items to it. Tuples are **unchangeable.**

Example – try to add orange to the item in thistuple

```
thistuple = ("apple", "banana", "cherry")
thistuple[3] = "orange" # This will raise an error
print(thistuple)
```

Remove item in a tuple

Note: You cannot remove items in a tuple.

Join two tuples

```
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)
tuple3 = tuple1 + tuple2
print(tuple3)
Out: ('a', 'b', 'c', 1, 2, 3)
```

Range

A range is a useful data type which most commonly encountered when using a for loop. It finds all integers x starting with a such a ≤ x < b and where two consecutive values are separated by i. range can be called with 1 or two parameters – range(a,b) is the same as range(a,b,1) and range(b) is the same as range(0,b,1).

```
b = range(10)

type(b)

Out: <class 'range'>

print(b)

range(0, 10)


list(b)

Out: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]


b = range(0,10,1)

list (b)

Out: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Note range(10) is the same as range(0,10) and range(0,10,1). range(0,10,1) creates integer starting with 0 to 9 with a step size of 1. You can change the step size

```
b = range(0,10,2)

list(b)

Out: [0, 2, 4, 6, 8]

c = range(3,10,3)

list(c)

Out: [3, 6, 9]
```

Set

Sets are collections which contain all unique elements of a collection. set and **frozenset** only differ in that the latter is immutable (and so has higher performance), and so set is similar to a unique list while **frozenset** is similar to a unique tuple.

Set functions

A number of methods are available for manipulating sets. The most useful are

Function	Method	Description
`set.add(x,`*element*`)`	`x.add(`*element*`)`	Appends *element* to a set.
`len(x)`	–	Returns the number of elements in the set.
`set.difference(x,`*set*`)`	`x.difference(`*set*`)`	Returns the elements in x which are not in *set*.
`set.intersection(x,`*set*`)`	`x.intersection(`*set*`)`	Returns the elements of x which are also in *set*.
`set.remove(x,`*element*`)`	`x.remove(`*element*`)`	Removes *element* from the set.
`set.union(x,`*set*`)`	`x.union(`*set*`)`	Returns the set containing all elements of x and *set*.

Examples

```
x = set(['MSFT','GOOG','AAPL','HPQ','MSFT'])

print(x)

Out: {'HPQ', 'AAPL', 'MSFT', 'GOOG'}
```

Note that ’**MSFT**’ is repeated in the list used to initialize the set, but only appears once in the set since all elements must be unique.

```
x.add('CSCO')


print( x)

Out: {'CSCO', 'MSFT', 'GOOG', 'AAPL', 'HPQ'}

x.add('CSCO')

print(x)
{'CSCO', 'MSFT', 'GOOG', 'AAPL', 'HPQ'}

y = set(['XOM', 'GOOG'])

print(x.intersection(y))
{'GOOG'}

x = x.union(y)
print( x)
{'HPQ', 'CSCO', 'MSFT', 'GOOG', 'XOM', 'AAPL'}

x.remove('XOM')
print(x)
{'HPQ', 'CSCO', 'MSFT', 'GOOG', 'AAPL'}
```

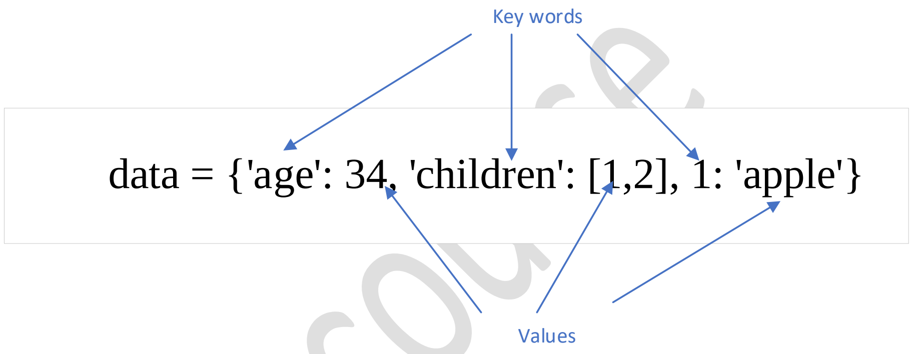
Dictionary

Dictionaries in Python are composed of keys (words) and values (definitions). Dictionaries keys must be unique primitive data types (e.g. strings, the most common key), and values can contain any valid Python data type.

Examples

#key words are ‘age’, ‘children’, 1

# values are 34, [1,2], ‘apple’



print(type(data))

<class 'dict'>

# to get the value in data

print(data['age'])

Out: 34

# to replace the value in ‘age’

data['age'] = 20

# check the update value

print(data['age'])

Out: 20

New key-value pairs can be added by defining a new key and assigning a value to it.

print(data)

{'age': 20, 'children': [1, 2], 1: 'apple'}

data['name'] = 'abc'   # add ‘name: abc’ to data

print(data)

{'age': 20, 'children': [1, 2], 1: 'apple', 'name': 'abc'}

Key-value pairs can be deleted using the reserved keyword del.

```
del data['children']

print(data)

{'age': 20, 1: 'apple', 'name': 'abc'}
```

1.3 Python operators

Python supports the following operators

- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

Let us look at the arithmetic, comparison or relational and assignment operators. The rest of the operators will be discussed in Chapter 3

Arithmetic Operators

Note: 4 + 5 (Here, 4 and 5 are called operands and + is called operator)

Operator	Description	Example
+	**Addition**	**23 +34 = 57**
-	**Subtraction**	**34 – 20 = 14**
*	**Multiplication**	**3*3 = 9**
/	**Division**	**8/4 = 2**
%	**Modulus (returns the remainder)**	**9%2 = 1**
**	**Exponent**	**2**3 = 8**
//	**Floor or round off**	**9//2 = 4**

Comparison operators

The comparison operators are also called relational operators. Values on either side of the operator are compared and the relationship is decided. Examples of the comparison operators are as follows.

Operator	Description	Example
= =	**Equal**	**a = 5 # store 5 in variable a** **b = 5 # store 5 in variable b** **a==5 # compare data in a with the value 5** **True # the computer returns a True**
!=	**Not equal**	**4 ! = 5 # check if 4 is not equal to 5** **True # the computer returns a True**
>	**Greater than**	**a= 5 # store the value 5 in a** **c = 4 # store the value 4 in c** **a>c #check if the data in a is greater than in c** **True # computer returns a True**
<	**Less than**	**a= 5 # store the value 5 in a** **c = 4 # store the value 4 in c** **a<c # check if the value in a is less than in c** **False**
>=	**Greater than or equal to**	**a= 10** **b = 10** **a>=10 # check if a is greater than or equal to 10** **True**
<=	**Less than or equal to**	**b = 10** **9<=b # check if 9 is less than or equal to value in b** **True**

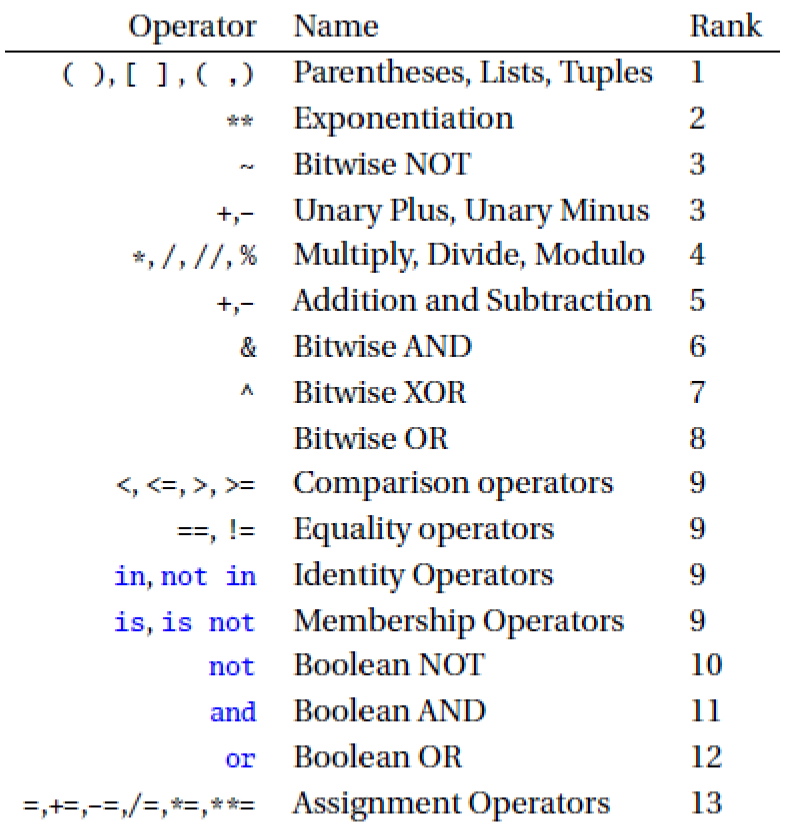
Assignment operators

Operator	Description	Example
=	Assigns values from right side operands to left side operand	**a = 3** **b = 4** **c = a+b** **print(c)** **7**
+=	It adds right operand to the left operand and assign the result to left operand	**c+=10 # means c = c +10** **print(c)** **17**
-=	It subtracts right operand from the left operand and assign the result to left operand	**c-=10 # means c = c- 10** **print(c)** **7**
/=	It divides left operand with the	**c/=2 # means c = c/2** **print(c)**

	right operand and assign the result to left operand	**3.5**
%=	It takes modulus using two operands and assign the result to left operand	**c%=3.5 # means c = c% 3.5** **print(c)** **0.0**
**=	Performs exponential (power) calculation on operators and assign value to the left operand	**a**=2 # means a = a**2** **print(a)** **16**
//=	It performs floor division on operators and assign value to the left operand	**a//=3 # means a= a//3** **print(a)** **5**

Operators Precedence

operators from highest precedence to lowest are shown as follows.



Operator	Name	Rank
( ), [ ], (,)	Parentheses, Lists, Tuples	1
**	Exponentiation	2
~	Bitwise NOT	3
+,-	Unary Plus, Unary Minus	3
*, /, //, %	Multiply, Divide, Modulo	4
+,-	Addition and Subtraction	5
&	Bitwise AND	6
^	Bitwise XOR	7
	Bitwise OR	8
<, <=, >, >=	Comparison operators	9
==, !=	Equality operators	9
in, not in	Identity Operators	9
is, is not	Membership Operators	9
not	Boolean NOT	10
and	Boolean AND	11
or	Boolean OR	12
=,+=,-=,/=,*=,**=	Assignment Operators	13

Note that some rows of the table have the same precedence and are only separated since they are conceptually different. In the case of a tie, operations are executed left-to-right. For example, x**y**z is interpreted as (x**y)**z. This table has omitted some operators available in Python which are not generally useful in numerical analysis (e.g. shift operators).

Note: Unary operators are + or - operations that apply to a single element. For example, consider the expression (-4). This is an instance of a unary negation since there is only a single operation and so (-4)**2 produces 16. On the other hand, 4**2 produces -16 since ** has higher precedence than unary negation and so is interpreted as (-4**2). 4* 4 produces 16 since it is interpreted as (4)*(4), since unary negation has higher precedence than multiplication.

Examples

```
a = 20
b = 10
c = 15
d = 5
e = 0
```

```
e = (a + b) * c / d       #( 30 * 15 ) / 5

print ("Value of (a + b) * c / d is ",  e)
```

```
e = ((a + b) * c) / d      # (30 * 15 ) / 5

print ("Value of ((a + b) * c) / d is ",  e)


e = (a + b) * (c / d);     # (30) * (15/5)

print ("Value of (a + b) * (c / d) is ",  e)


e = a + (b * c) / d;       #  20 + (150/5)

print ("Value of a + (b * c) / d is ",  e)
```

When you execute the above program, it produces the following result −

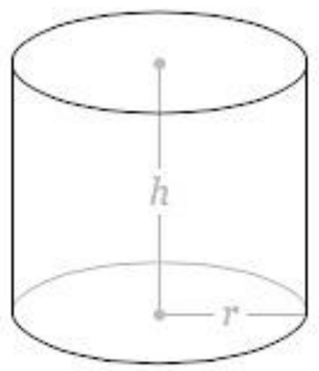
```
Value of (a + b) * c / d is 90

Value of ((a + b) * c) / d is 90

Value of (a + b) * (c / d) is 90

Value of a + (b * c) / d is 50
```

1.4 Application

Now let us see how we can apply what we have just learnt from this chapter.

Example 1.

I want to write a program to calculate and display the volume of a cylinder using the following algorithm



```
1. Get the height, let say h
2. Get the radius, let say r
3. Calculate the result let say V, V = π * h * r²
4. Display the result, V
```

Solution

To get the input of height, radius from the key board you need to use the function called **input().** Note whatever you enter as input, the input() function always converts it into a string.

To carry out a calculation, we need to convert the input value to integer data type to be able to perform our calculation.

```
# Example 1, program to calculate the volume of a cylinder
import math   # this imports the math library.
Pi = math.pi   # takes the value of pi from math library and saves it in the Variable Pi
height = (input('Enter a height of cylinder: ')
radius = input('Enter a radius of cylinder: ')
# convert from string to integer
height = int(height)
radius = int (radius)
# calculate the volume
volume = Pi* height* radius**2   # volume = π * h *r^2

print("The volume of Cylinder is: ", volume )
```

Result

```
Enter a height of cylinder: 5

Enter a radius of cylinder: 20
The volume of Cylinder is:  6283.185307179586
```

Example 2

I want to create a registration form for my customers with the following details

First name	
Last name	
Age	
Address	
State	
Marital status	

There are several ways to do this. In this example I will do this using the Dictionary

First step is to create variable to store the data of the user

```
first_name = input('Enter first name : ')
last_name = input('Enter Last name : ')
age = input('Enter your age  : ')
state = input('Enter your Sate   : ')
Address = input('Enter your address  : ')
marital_status = input('Enter your Marital status  : ')
```

Second step is to create a dictionary of the variables

```
# convert the variable into a dictionary
customer_one = {'First name': first_name, 'Last name': last_name, 'Age': age, 'State': state,
'Address':address, 'Status':marital_status }
print(customer_one)
```

Result

```
Enter first name : ola
Enter Last name : Elijah
Enter your age  : 18
Enter your Sate   : Johor
Enter your address  : 23 Taman Utama
Enter your Marital status  : Single

{'First name': 'ola', 'Last name': 'Elijah', 'Age': '18', 'State': 'Johor', 'Address': '23 Taman
Utama', 'Status': 'Single '}
```

Next, I am interested in finding out the first name and last name of the customer

```
# get the first name and last name of the customer

print('first name is : ', customer_one['First name'])
print('last name is : ', customer_one['Last name'])
```

Result

```
# get the first name and last name of the customer

first name is :  ola
last name is :  Elijah
```

Next, I want to add 20 to the current age of the customer, to do this we write the following code

```
# increase the customer age by 20

new_age = int(customer_one['Age']) + 20
print('New age is :', new_age)
```

Result

```
New age is : 38
```

Example 3

I want to create a data for years starting from 1980 to 2020 with intervals of 2 years, that is

1980, 1982, 1984, 1986 …………….2020.

To do this write the following codes

```
#Example 3  - Create range of years from 1980 to 2020 with interval of 2 years
b = range(1980,2020,2)
list(b)     # list the years in b
```

Result

```
[1980,
 1982,
 1984,
 1986,
 1988,
 1990,
 1992,
 1994,
 1996,
 1998,
 2000,
 2002,
 2004,
 2006,
 2008,
 2010,
 2012,
 2014,
 2016,
 2018]
```

Example 4

I want to create username and save it as a tuple. I will use a tuple because you cannot easily change the data in the tuple

Username

To do this I will write the following codes

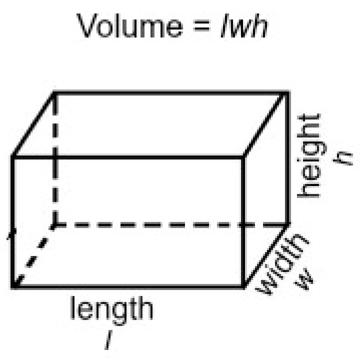
```
# Example 4  creating a user name with tuple
user_name = input('Enter your user name : ')
user_name = (user_name,)
print(type(user_name))
```

1.5 Summary

1. First, we have learnt more about the following data types tuple, range, set, dictionary
2. We have learnt some of the basic operators in python
3. We have practiced some example and seen applications of the theory learnt.

1.6 Exercise

1. Consider a given problem below to calculate and display the volume of a cuboid.



Write a program to calculate the volume of the cuboid

2. As a programmer, you have been given the task to create a user registration form with the following details.

User name	
Password	
First name	
Last name	
Age	
Address	
State	
Marital status	

For security purpose, make the user name and the password a tuple. Then create a dictionary named **user_record** to store all the variable from the registration form. Print the user_record out to show the details of the user.

Exercise is due on the 29th of January

1.7 Reference

[1] http://www.tutorialspoint.com/python/python_basic_operators.htm

[2] https://www.w3schools.com/python/python_tuples.asp

Chinese Proverbs.

“And remember each 10,000 mile journey begins with just 1 step” (千里之行，始於足下 Qiānlǐ zhī xíng, shǐyú zú xià. Laozi.)