



**K. J. Somaiya Polytechnic, Mumbai-77**

**Batch No: C1**

**Enrollment No: FCOW19118**

**Experiment No.: 02**

**Experiment Name: Develop programs to learn different types of structures (list, dictionary, tuples, arrays) in python.**

**Experiment No: 02**

**Experiment Name: Develop programs to learn different types of structures (list, dictionary, tuples, arrays) in python.**

**Course Outcome:****O18RA72.1****Understand basics of Python programming.****Theory:****Data types in Python**

Every value in Python has a datatype. Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.

There are various data types in Python. Some of the important types are listed below:

**1. Python Numbers**

Integers, floating point numbers and complex numbers fall under Python numbers category. They are defined as int, float and complex classes in Python.

We can use the type() function to know which class a variable or a value belongs to. Similarly, the isinstance() function is used to check if an object belongs to a particular class.

Code	Output
<pre>a = 5 print(a, "is of type", type(a)) a = 2.0 print(a, "is of type", type(a)) a = 1+2j print(a, "is complex number?", isinstance(1+2j, complex))</pre>	<pre>5 is of type &lt;class 'int'&gt; 2.0 is of type &lt;class 'float'&gt; (1+2j) is complex number? True</pre>

**Note:**

- Integers can be of any length; it is only limited by the memory available.
- A floating-point number is accurate up to 15 decimal places. Integer and floating points are separated by decimal points. 1 is an integer, 1.0 is a floating-point number.
- Complex numbers are written in the form,  $x + yj$ , where  $x$  is the real part and  $y$  is the imaginary part. Here are some examples:

Code	Output
<pre>&gt;&gt;&gt; a = 1234567890123456789 &gt;&gt;&gt; a</pre>	1234567890123456789
<pre>&gt;&gt;&gt; b = 0.1234567890123456789 &gt;&gt;&gt; b</pre>	0.12345678901234568
<pre>&gt;&gt;&gt; c = 1+2j &gt;&gt;&gt; c</pre>	(1+2j)

Notice that the float variable `b` got truncated.

## 2. Python List

List is an ordered sequence of items. It is one of the most used datatype in Python and is very flexible. All the items in a list do not need to be of the same type.

Declaring a list is pretty straight forward. Items separated by commas are enclosed within brackets `[ ]`.

```
a = [1, 2.2, 'python']
```

We can use the slicing operator `[ ]` to extract an item or a range of items from a list. The index starts from 0 in Python.

```
a = [5,10,15,20,25,30,35,40]
```

Code	Output
<pre># a[2] = 15 print("a[2] = ", a[2]) # a[0:3] = [5, 10, 15] print("a[0:3] = ", a[0:3]) # a[5:] = [30, 35, 40] print("a[5:] = ", a[5:])</pre>	<pre>a[2] = 15 a[0:3] = [5, 10, 15] a[5:] = [30, 35, 40]</pre>

Lists are mutable, meaning, the value of elements of a list can be altered.

Code	Output
<pre>a = [1, 2, 3] a[2] = 4 print(a)</pre>	[1, 2, 4]

### 3. Python Tuple

Tuple is an ordered sequence of items same as a list. The only difference is that tuples are immutable. Tuples once created cannot be modified.

Tuples are used to write-protect data and are usually faster than lists as they cannot change dynamically.

It is defined within parentheses () where items are separated by commas.

```
t = (5,'program', 1+3j)
```

We can use the slicing operator [] to extract items but we cannot change its value.

```
t = (5,'program', 1+3j)
```

Code	Output
<pre># t[1] = 'program' print("t[1] = ", t[1]) # t[0:3] = (5, 'program', (1+3j)) print("t[0:3] = ", t[0:3]) # Generates error # Tuples are immutable t[0] = 10</pre>	<pre>t[1] = program t[0:3] = (5, 'program', (1+3j)) Traceback (most recent call last) :   File "test.py", line 11, in &lt;module&gt;     t[0] = 10 TypeError: 'tuple' object does not support item assignment</pre>

### 4. Python Strings

String is sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ''' or ''''.

Code	Output
<pre>s = "This is a string" print(s) s = '''A multiline</pre>	<pre>This is a string A multiline string</pre>

```
string'''
print(s)
```

Just like a list and tuple, the slicing operator [ ] can be used with strings. Strings, however, are immutable.

Code	Output
<pre>s = 'Hello world!' # s[4] = 'o' print("s[4] = ", s[4]) # s[6:11] = 'world' print("s[6:11] = ", s[6:11]) # Generates error # Strings are immutable in Python s[5] = 'd'</pre>	<pre>s[4] = o s[6:11] = world Traceback (most recent call last):   File "&lt;string&gt;", line 11, in &lt;module&gt; TypeError: 'str' object does not support item assignment</pre>

## 5. Python Set

Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

Code	Output
<pre>a = {5,2,3,1,4} # printing set variable print("a = ", a)  # data type of variable a print(type(a))</pre>	<pre>a = {1, 2, 3, 4, 5} &lt;class 'set'&gt;</pre>

We can perform set operations like union, intersection on two sets. Sets have unique values. They eliminate duplicates.

Code	Output
<pre>a = {1,2,2,3,3,3} print(a)</pre>	<pre>{1, 2, 3}</pre>

Since, set are unordered collection, indexing has no meaning. Hence, the slicing operator [ ] does not work.

Code	Output
<pre>&gt;&gt;&gt; a = {1,2,3} &gt;&gt;&gt; a[1]</pre>	<pre>Traceback (most recent call last):   File "&lt;string&gt;", line 301, in runcode   File "&lt;interactive input&gt;", line 1, in &lt;module&gt; TypeError: 'set' object does not support indexing</pre>

## 6. Python Dictionary

Dictionary is an unordered collection of key-value pairs.

It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.

In Python, dictionaries are defined within braces { } with each item being a pair in the form key:value. Key and value can be of any type.

Code	Output
<pre>&gt;&gt;&gt; d = {1:'value','key':2} &gt;&gt;&gt; type(d)</pre>	<pre>&lt;class 'dict'&gt;</pre>

We use key to retrieve the respective value. But not the other way around.

Code	Output
<pre>d = {1:'value','key':2} print(type(d)) print("d[1] = ", d[1]); print("d['key'] = ", d['key']); # Generates error print("d[2] = ", d[2]);</pre>	<pre>&lt;class 'dict'&gt; d[1] = value d['key'] = 2 Traceback (most recent call last):   File "&lt;string&gt;", line 9, in &lt;module&gt; KeyError: 2</pre>

## 7. Creating Python Arrays

An array is a special variable, which can hold more than one value at a time.

If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this :

```
car1 = "Ford"
```

```
car2 = "Volvo"
```

```
car3 = "BMW"
```

- However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?
- The solution is an array!

An array can hold many values under a single name, and you can access the values by referring to an index number.

Note : To work with arrays in Python you will have to import a library, like the NumPy library.

OR

Import a module named as 'array'

Code	Output
<pre>import array as arr a = arr.array('d', [1.1, 3.5, 4.5]) print(a) <b>OR</b> import numpy as np a = np.array('d', [1.1, 3.5, 4.5]) print(a)</pre>	<pre>array('d', [1.1, 3.5, 4.5])</pre>

Here, we created an array of float type. The letter d is a type code. This determines the type of the array during creation.

## 8. Conversion between data types

- a. We can convert between different data types by using different type conversion functions like int(), float(), str(), etc.

```
>>> float(5)
```

```
5.0
```

- b. Conversion from float to int will truncate the value (make it closer to zero).

```
>>> int(10.6)
```

```
10
```

```
>>> int(-10.6)
```

```
-10
```

c. Conversion to and from string must contain compatible values.

```
>>> float('2.5')
```

```
2.5
```

```
>>> str(25)
```

```
'25'
```

```
>>> int('1p')
```

Traceback (most recent call last):

File "<string>", line 301, in runcode

File "<interactive input>", line 1, in <module>

ValueError: invalid literal for int() with base 10: '1p'

d. We can even convert one sequence to another.

```
>>> set([1,2,3])
```

```
{1, 2, 3}
```

```
>>> tuple({5,6,7})
```

```
(5, 6, 7)
```

```
>>> list('hello')
```

```
['h', 'e', 'l', 'l', 'o']
```

e. To convert to dictionary, each element must be a pair :

```
>>> dict([[1,2],[3,4]])
```

```
{1: 2, 3: 4}
```

```
>>> dict([(3,26),(4,44)])
```

```
{3: 26, 4: 44}
```

## Implementation and Output

### 1. Types of Structures in Python

#### Program:

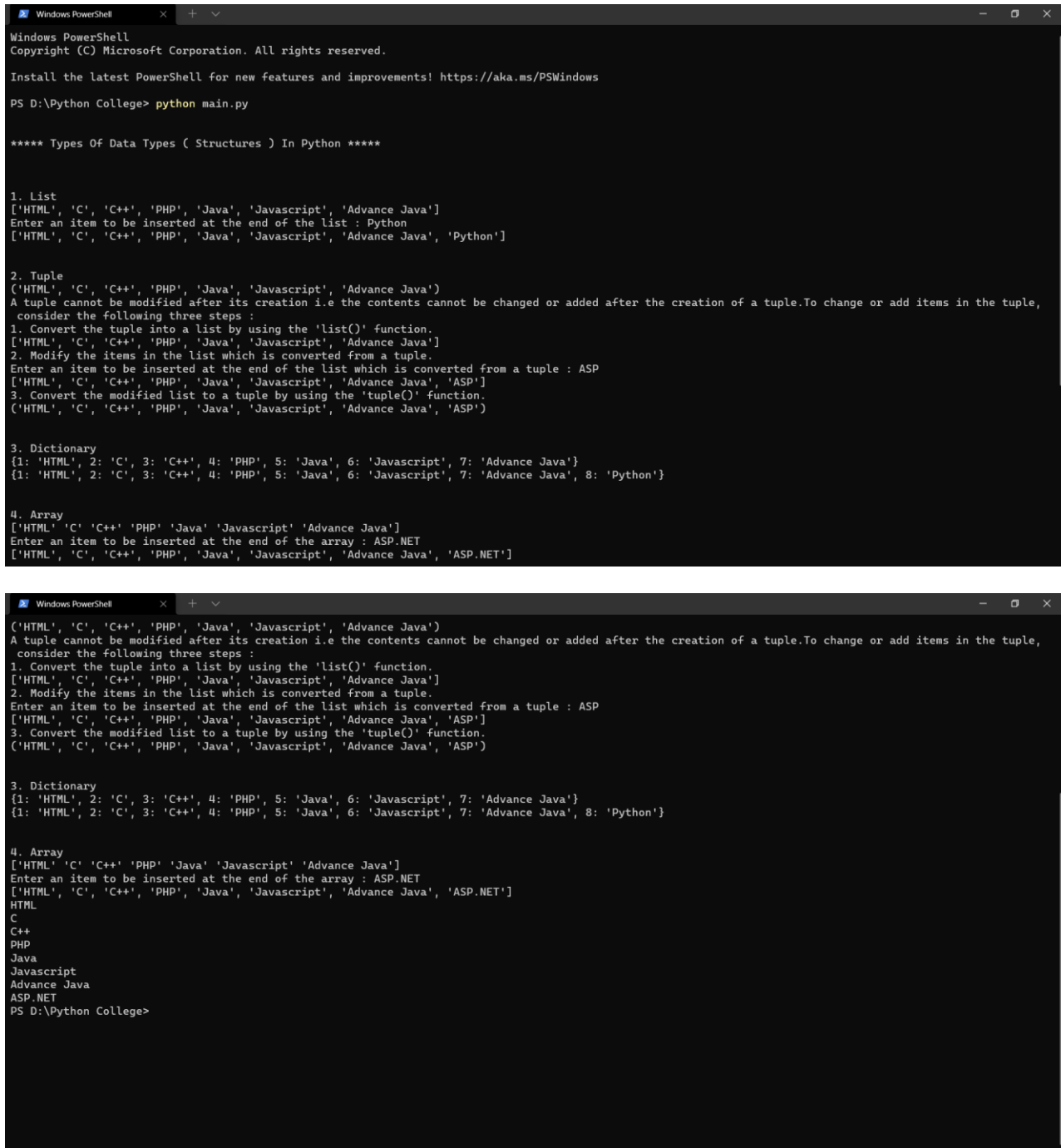
```
print("\n")
print("***** Types Of Data Types ( Structures ) In Python *****")
print("\n\n")
```



```
print("1. List")
list_variable=["HTML","C","C++","PHP","Java","Javascript","Advance Java"]
print(list_variable)
list_variable.append(input("Enter an item to be inserted at the end of the
list : "))
print(list_variable)
print("\n")
print("2. Tuple")
tuple_variable=("HTML","C","C++","PHP","Java","Javascript","Advance Java")
print(tuple_variable)
print("A tuple cannot be modified after its creation i.e the contents cannot
be changed or added after the creation of a tuple.")
print("To change or add items in the tuple, consider the following three steps
:")
print("1. Convert the tuple into a list by using the 'list()' function.")
tuple_to_list=(list(tuple_variable))
print(tuple_to_list)
print("2. Modify the items in the list which is converted from a tuple.")
tuple_to_list.append(input("Enter an item to be inserted at the end of the
list which is converted from a tuple : "))
print(tuple_to_list)
print("3. Convert the modified list to a tuple by using the 'tuple()'
function.")
list_to_tuple=tuple(tuple_to_list)
print(list_to_tuple)
print("\n")
print("3. Dictionary")
dictionary_object={1:"HTML",2:"C",3:"C++",4:"PHP",5:"Java",6:"Javascript",7:"
Advance Java"}
print(dictionary_object)
dictionary_object[8]="Python"
print(dictionary_object)
print("\n")
print("4. Array")
languages=["HTML","C","C++","PHP","Java","Javascript","Advance Java"]
import numpy as np
languages_array=np.array(languages)
print(languages_array)
languages.append(input("Enter an item to be inserted at the end of the array
: "))
print(languages)
for language in languages:
```

```
print(language)
```

## Output:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\Python College> python main.py

***** Types Of Data Types ( Structures ) In Python *****

1. List
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java']
Enter an item to be inserted at the end of the list : Python
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'Python']

2. Tuple
('HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java')
A tuple cannot be modified after its creation i.e the contents cannot be changed or added after the creation of a tuple. To change or add items in the tuple,
consider the following three steps :
1. Convert the tuple into a list by using the 'list()' function.
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java']
2. Modify the items in the list which is converted from a tuple.
Enter an item to be inserted at the end of the list which is converted from a tuple : ASP
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP']
3. Convert the modified list to a tuple by using the 'tuple()' function.
('HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP')

3. Dictionary
{1: 'HTML', 2: 'C', 3: 'C++', 4: 'PHP', 5: 'Java', 6: 'Javascript', 7: 'Advance Java'}
{1: 'HTML', 2: 'C', 3: 'C++', 4: 'PHP', 5: 'Java', 6: 'Javascript', 7: 'Advance Java', 8: 'Python'}

4. Array
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java']
Enter an item to be inserted at the end of the array : ASP.NET
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP.NET']

('HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java')
A tuple cannot be modified after its creation i.e the contents cannot be changed or added after the creation of a tuple. To change or add items in the tuple,
consider the following three steps :
1. Convert the tuple into a list by using the 'list()' function.
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java']
2. Modify the items in the list which is converted from a tuple.
Enter an item to be inserted at the end of the list which is converted from a tuple : ASP
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP']
3. Convert the modified list to a tuple by using the 'tuple()' function.
('HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP')

3. Dictionary
{1: 'HTML', 2: 'C', 3: 'C++', 4: 'PHP', 5: 'Java', 6: 'Javascript', 7: 'Advance Java'}
{1: 'HTML', 2: 'C', 3: 'C++', 4: 'PHP', 5: 'Java', 6: 'Javascript', 7: 'Advance Java', 8: 'Python'}

4. Array
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java']
Enter an item to be inserted at the end of the array : ASP.NET
['HTML', 'C', 'C++', 'PHP', 'Java', 'Javascript', 'Advance Java', 'ASP.NET']
HTML
C
C++
PHP
Java
Javascript
Advance Java
ASP.NET
PS D:\Python College>
```

**Conclusion:** Thus, we have developed programs to learn different types of structures (list, dictionary, tuples, arrays) in python.