bianca.phelan@tudublin.ie

O Object
O Oriented
P Programming

S1 = Python with Bianca

S2 = Java with Bryan

DT228(TU856)/DT282(TU858) - 2

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TU DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN

COMPUTER SCIENCE

Lab3 Solution Background

Object Attributes

[1]

- Identity
  - Address in memory
  - Unchangeable
  - `id()`
- Type
  - Defines possible values and operations
  - Unchangeable
  - `type()`
- Value
  - Some objects can change their value: **mutable**
  - Some objects cannot change their value: **immutable**

The mutability of an object is determined by its type.

An Object's ID

- ID can be seen as an address in memory
- `id()` returns the memory address location, it's a built-in Python function

```
name = "Bianca"
print(id(name))
```

```
140426659034352

Process finished with exit code 0
```

Python Data Types

Mutable	Immutable
List	Int
Dictionary	Float
Set	String
User-defined Classes	Bool
	Decimal
	Tuple
	Range

Once immutable variables are initialised, their values cannot be changed.

Comparing List (mutable) to Tuple (immutable) [1]

1

```
names = ["Bianca", "Bryan", "Susan"]
cities = ("Dublin", "Galway", "Cork")

print(names)
print(id(names))

print(cities)
print(id(cities))
```

```
/Users/bianca.schoenphelan/Doc
['Bianca', 'Bryan', 'Susan']
140425298527488
('Dublin', 'Galway', 'Cork')
140425298576576
```

2

```
print(names[0])
print(cities[0])
```

```
Bianca
Dublin
```

3

```
names[0] = "Lucas"
print(names[0])

cities[0] = "Portlaoise"
print(cities[0])
```

```
/Users/bianca.schoenphelan/Documents/OOP_Class/Code/venv/bin
Lucas
Traceback (most recent call last):
  File "/Users/bianca.schoenphelan/Documents/OOP_Class/Code/
    cities[0] = "Portlaoise"
TypeError: 'tuple' object does not support item assignment
```

Comparing Lists and Tuples cont'd

```
print("names location: ",id(names))
print("cities location: ", id(cities))
names += ["Bianca", "Paul"]
cities += ("Portlaoise", "Swords")
print("after change names location: ",id(names))
print("after change cities location: ", id(cities))
```

```
names location:  140610939471040
cities location:  140610939520128
after change names location:  140610939471040
after change cities location:  140610910351424
```

Same address in memory!

New address in memory!

Other Immutable Data Types

1

```
age = 21
print(id(age))
age += 1
print(age)
print(id(age))
```

```
/Users/bianca.
4304203072
22
4304203104
```

2

```
name = "Bianca"
print(id(name))
name += " Phelan"
print(id(name))
```

```
/Users/bianca.sch
140683584291952
140683584292592
```

3

```
name[0] = "D"
```

You probably noticed this in lab 3!

```
/Users/bianca.schoenphelan/Documents/OOP_Class/Code/venv/b
Traceback (most recent call last):
  File "/Users/bianca.schoenphelan/Documents/OOP_Class/Coc
    name[0] = "D"
TypeError: 'str' object does not support item assignment
```

Copying Mutable Objects by Reference [1]

```
names = ["Bianca", "Bryan", "Susan"]
names2 = names

print(id(names))
print(id(names2))

names.append("Paul")
print(names is names2)

print(names)
print(names2)
```

`is` checks for the memory address.

```
/Users/bianca.schoenphelan/Documents/C
140638535861504
140638535861504
True
['Bianca', 'Bryan', 'Susan', 'Paul']
['Bianca', 'Bryan', 'Susan', 'Paul']
```

Copying Lists **NOT** by Reference

[2]

```
names = ["Bianca", "Bryan", "Susan"]
names2 = names # creates a copy by reference
names3 = names[:] # not a copy by reference
names4 = list(names) # not a copy by reference

print(id(names))
print(id(names2))
print(id(names3))
print(id(names4))
```

```
140683290166528
140683290166528
140683290669376
140683290670784
```

```
names.append("Paul")
print(names is names2)
print(names is names3)
print(names is names4)

print(names)
print(names2)
print(names3)
print(names4)
```

```
True
False
False
['Bianca', 'Bryan', 'Susan', 'Paul']
['Bianca', 'Bryan', 'Susan', 'Paul']
['Bianca', 'Bryan', 'Susan']
['Bianca', 'Bryan', 'Susan']
```

Summary

- ★ **Mutable vs Immutable**
- ★ **Memory address vs value of variable**
- ★ **Copy by reference**

References

1. Python Basics: Mutable vs Immutable Objects, Towards Data Science, Ventislav Yordanov, 3 Feb 2019, https://towardsdatascience.com/https-towardsdatascience-com-python-basics-mutable-vs-immutable-objects-829a0cb1530a, accessed Oct 2020.
2. Python Basics for Data Science, Ventislav Yordanov, 1 Feb 2018, https://towardsdatascience.com/python-basics-for-data-science-6a6c987f2755, accessed Oct 2020.