Python OOP (Object Oriented Programming)

Gramsci Hermozo

Course Content (1/3)

1. Introduction to Python
2. Introduction to OOP
3. UML
4. Implement our own objects
5. Design our own objects

Course Content (2/3)

6. Exception Handling
7. Manage Errors friendly
8. Comparing Objects
9. Polymorphism
10. Testing

Course Content (3/3)

11. Inheritance
12.

Scoring

- Practices (40/100)
- Project (60/100)
  - UML
  - Logic
  - Code
- Written Test
- Test Code

Tools

- Python installed
- Any text editor or IDE, could be PyCharm or Visual Studio Code
- Diagram Software, could be Draw.io or any used by you.

Introduction to Python

Content

Getting Started

- Created by Guido Van Rossum
- Is high level programming language
- Features
  - Dynamic type system
  - Automatic memeory management
  - Support multiple Programming paradigms
    - Object Oriented
    - Imperative
    - Functional Programming
    - Procedural Styles
  - Have a large and Comprenhensive standard library

Install Python on Linux

Open a terminal and execute the following command's

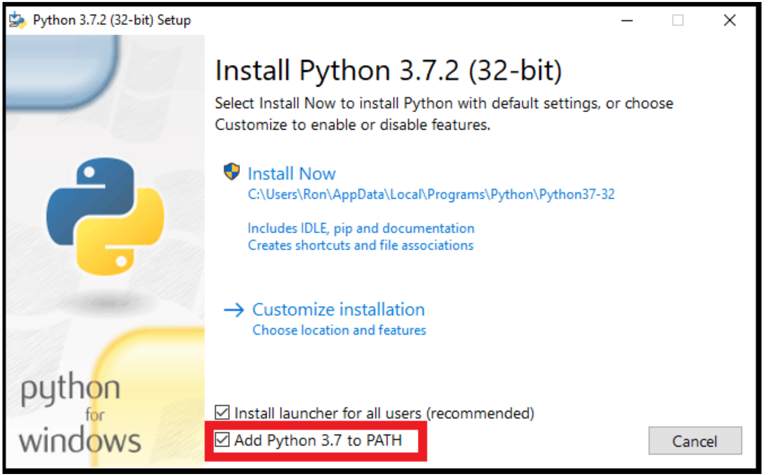
```
sudo apt-get update
sudo apt-get install python
```

After that install PIP (package installer for python)

```
sudo apt-get update
sudo apt-get install python-pip
```

Install Python on Windows

- Go to python web page here
- Download Windows installer.
- Important, check the checkbox to “Add Python 3.x to PATH”.


Python 3.7.2 (32-bit) Setup

Install Python 3.7.2 (32-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\Ron\AppData\Local\Programs\Python\Python37-32

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

Virtual Environments for Python (virtualenv)

Is a way that you can separate different python environments for different projects.

How to install

```
pip install virtualenv
```

Create

```
virtualenv <project_name>
```

Activate

```
source <project_name>/bin/activate
```

To get out

```
deactivate
```

Datatype

- Integer type
- Floating type
- Boolean type
- Null type

Variables And Assigning Values

- All you need to do is specify the variable name and then assign a value

```
<variable_name> = <value>
```

Operators

Arithmetic Operators

Operator	Name
+	Addition
-	Subtraction
	Multiplication
/	Division
%	Modulus
	Exponentiation
//	Floor division

Comparison Operators

Operator	Name
==	Equal
!=	Not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Logical Operators

Operator	Description
and	Return True if both statements are tru
or	Return True if one of the statements is true
not	Reverse the result, returns False if the result is true

Identity Operators

Operator	Description
is	Returns True if both variables are the same object
is not	Returns True if both variables are not the same object

Membership Operators

Operator	Description
in	Returns True if a sequence with the specified value is present in the object
not in	Returns True if a sequence with the specified value is not present in the object

Bitwise Operators

- Used to compare (binary) numbers

Operator	Name
&	AND
|	OR
ˆ	XOR
∼	NOT
«	Zero fill left shift
»	Signed right shift