

# Python OOP (Object Oriented Programming)

Gramsci Hermozo

# Course Content (1/3)

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

## Course Content (2/3)

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

# 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

1. Getting Started
2. Installation (Linux, Windows)
3. Virtual Environments
4. Datatype, Variables and Assigning values
5. Operators

# Getting Started

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



# Install Python on Linux

Open a terminal and execute the following command's

```
sudo apt-get update
```

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

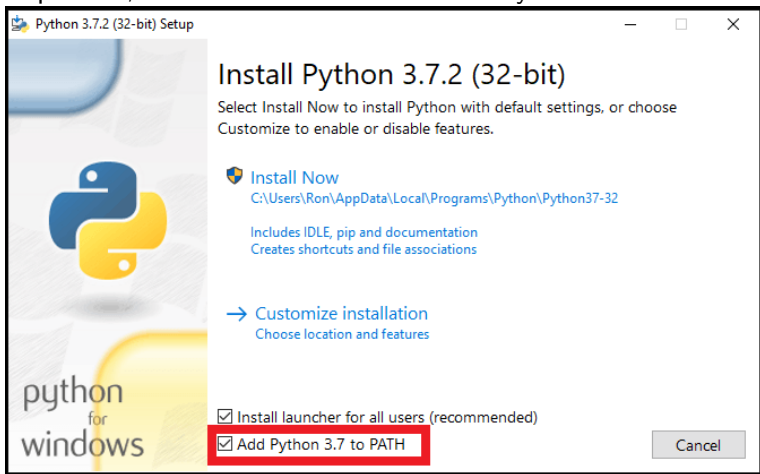
After that install PIP (package installer for python)

```
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”.



# 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 true                 |
| 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                   |
| ^        | XOR                  |
| ~        | NOT                  |
| «        | Zero fill left shift |
| »        | Signed right shift   |