

Introduction To Python Programming



Hello!

I am Eslam Ahmed

I am a software engineer.

You can find me at jeksogsa@gmail.com



Hello!

I am Eman Ehab

I am a ML research engineer.

You can find me at
emanehab.ieee@gmail.com



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- **Input & Output**
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Input & Output



```
1 # Input  
2 name = input('Please enter your name: ')
```



```
1 # Output  
2 print('hello world, Python is awesome ^_^')
```



Input & Output

Comments

```
1 # making a single line comment
2 name = 'ahmed'
3
4
5 print('hello') # inline comment
6
7
8 """
9 multi line comments
10 you can make many lines
11 """
```



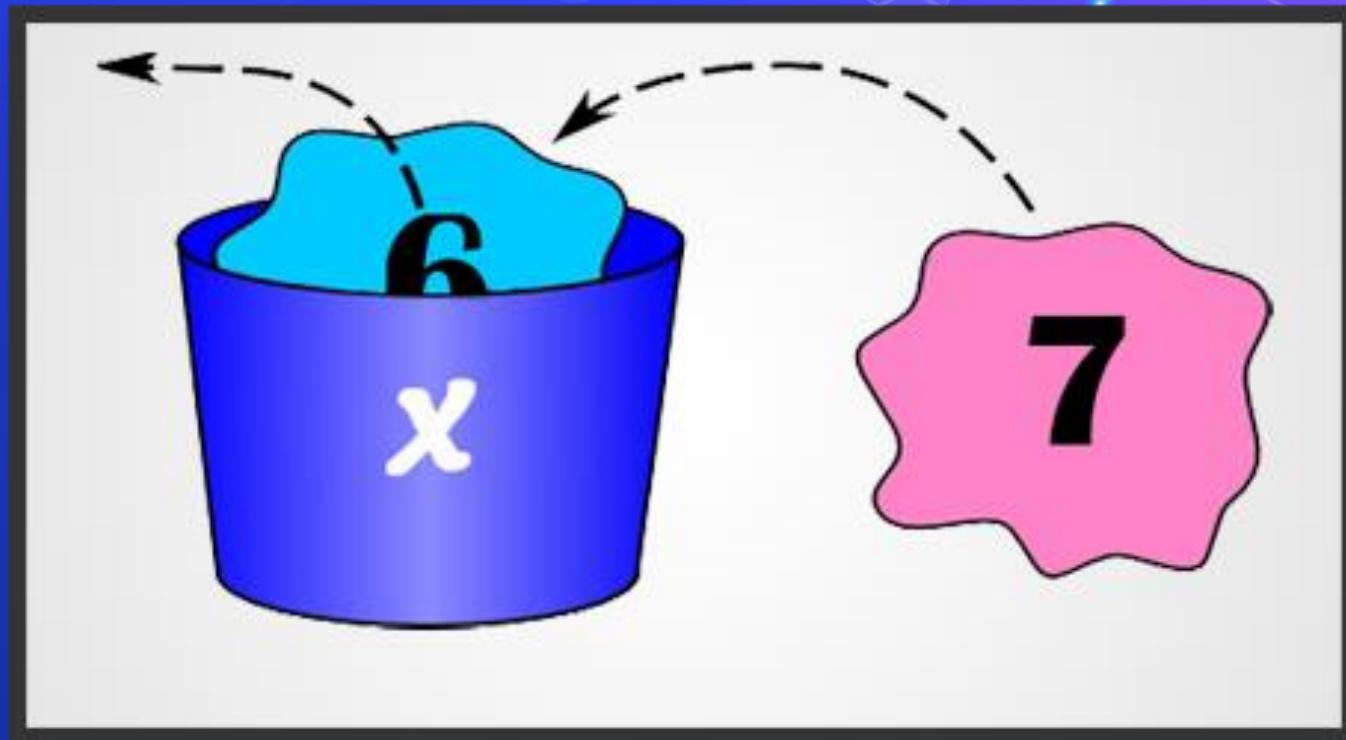
Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- **Variables**
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Variables



Variables

```
1 # there is a location in RAM called 'x' and has value = 6
2 x = 6
3
4 # now x is now 7
5 x = 7
```

Variables

```
1 # variables can be anything (numbers, text, lists, ...)  
2 number_1 = 20  
3 number_2 = 2.5  
4 name = 'Eman'  
5 is_online = True  
6 is_online = False  
7 fruits = ['apples', 'oranges', 'grapes']  
8 user = {'name': 'Eslam', 'age': 30, 'gender': 'male'}
```

Variables

Rules for variables assignment

- Names cannot start with numbers.
- There can be no spaces, use _ instead.
- Can't use these symbols :'"<>/?|()!@#\$%^&*~-+=[],.
- Avoid using words that have special meaning in Python like list.
- For best practice user lowercase letters.
- Use easy and meaningful name and related to the problem.

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- **Data types**
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Data types

Name	Type	Description
Integers	int	Whole numbers, such as: 3 300 200
Floating point	float	Numbers with a decimal point: 2.3 4.6 100.0
Strings	str	Ordered sequence of characters: "hello" 'Sammy' "2000" "楽しい"
Lists	list	Ordered sequence of objects: [10,"hello",200.3]
Dictionaries	dict	Unordered Key:Value pairs: {"mykey": "value", "name": "Frankie"}
Tuples	tup	Ordered immutable sequence of objects: (10,"hello",200.3)
Sets	set	Unordered collection of unique objects: {"a","b"}
Booleans	bool	Logical value indicating True or False

Data types

```
1 # use type() function to check what is the type of a variable
2 x = 5
3 y = "python is awesome"
4 z = [1, 2, 3]
5
6 print(type(x)) # int
7 print(type(y)) # str
8 print(type(z)) # list
```



100

001

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - **Numbers & Math**
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Numbers & Math

```
1 # this is integer values (int)
2 5
3 1000
4 -5000
5 0
6
7
8 # this is float values (float)
9 5.25
10 1000.75
11 -5000.3
12 5.0
```



Numbers & Math

```
1 3 + 5      # result is 8
2 10 - 7     # result is 3
3 2 * 5      # result is 10
4 15 / 5     # result is 3
5 3 / 2      # result is 1.5
6 3 // 2     # result is 1
7 32 % 3     # result is 2
8 2 ** 3     # result is 8
9 4 ** 0.5   # result is 2
```



Numbers & Math

```
1 x += 5 # x = x + 5  
2 x -= 5 # x = x - 5  
3 x *= 5 # x = x * 5  
4 x /= 5 # x = x / 5  
5 x %= 5 # x = x % 5  
6 x //= 5 # x = x // 5  
7 x **= 5 # x = x ** 5
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - **Boolean & Comparison and Logic**
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Boolean & Comparison and Logic

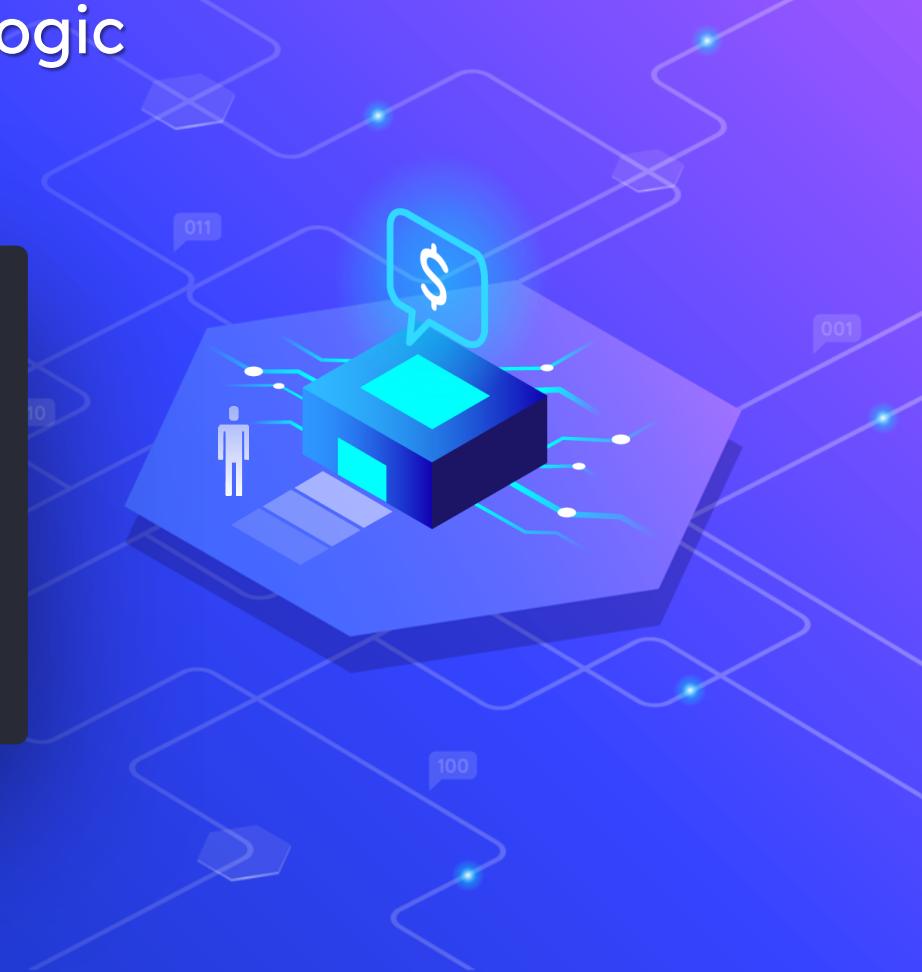


```
1 is_online = True  
2  
3 has_dog = False
```



Boolean & Comparison and Logic

```
1 5 == 5      # result is True  
2 5 != 5      # result is False  
3 10 > 7      # result is True  
4 2 >= 5       # result is False  
5 15 < 5       # result is False  
6 3 <= 3       # result is True
```



Boolean & Comparison and Logic

```
1 # AND
2 1 < 2 and 2 < 3          # Result is True
3 1 != 1 and 2 < 3         # Result is False
4 1 != 1 and 2 > 3         # Result is False
5
6
7 # OR
8 1 < 2 or 2 < 3           # Result is True
9 1 != 1 or 2 < 3           # Result is True
10 1 != 1 or 2 > 3          # Result is False
11
12
13 # NOT
14 not 1 == 1                # Result is False
15 not 1 > 10               # Result is True
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - **Strings**
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Strings



```
1 greeting = "Hello World"  
2 greeting = 'Hello World'
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - **Lists**
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Lists

```
1 my_list = ['A string', 23, 100.232 , 'p', True]
2
3 print(my_list[0])    # 'A string'
4 print(my_list[1])    # 23
5 print(my_list[2])    # 100.232
6 print(my_list[3])    # 'p'
7 print(my_list[4])    # True
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - **Tuples**
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Tuples (faster and immutable lists)

Used when you have immutable values and need faster processing on them

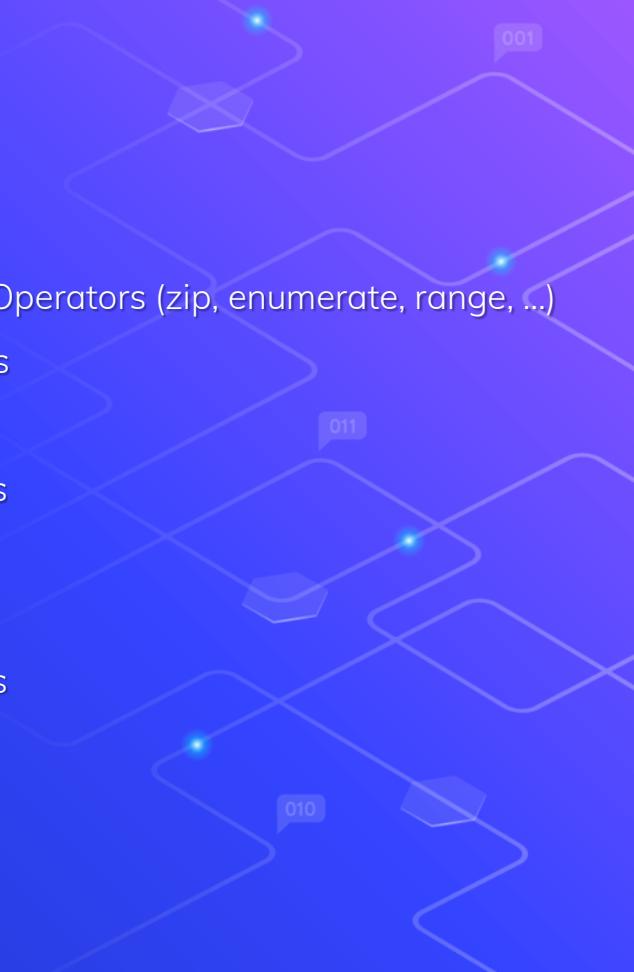
```
1 my_tuple = ('A string', 23, 100.232, 'p', True)
2
3 print(my_tuple[0]) # 'A string'
4 print(my_tuple[1]) # 23
5 print(my_tuple[2]) # 100.232
6 print(my_tuple[3]) # 'p'
7 print(my_tuple[4]) # True
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - **Sets**
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Sets (unique lists)

Used for intersections & union operations

```
1 my_list = [1,1,2,2,3,4,5,6,1,1]
2
3 my_set = set(my_list)
4 print(my_set)    # {1, 2, 3, 4, 5, 6}
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - **Dictionaries**

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Dictionaries

```
1 store = {'apples': 10, 'oranges': 20}  
2  
3 print(store['apples']) # result is 10  
4 print(store['oranges']) # result is 20
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

File Handling

```
1 # Read from file
2 my_file = open('test.txt', 'r')
3 print(my_file.read())    # or use readlines()
4 my_file.close()
5
6
7 # Write to a file
8 my_file = open('test.txt', 'w')  # or w+ for read & write
9 print(my_file.write('Hello Python'))
10 my_file.close()
11
12
13 # Append to a file
14 my_file = open('test.txt', 'a')  # or a+ for read & append
15 print(my_file.write('Hello Python'))
16 my_file.close()
17
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



If Conditions

```
1 person = 'George'  
2  
3 if person == 'Sammy':  
4     print('Welcome Sammy!')  
5 elif person =='George':  
6     print('Welcome George!')  
7 else:  
8     print("Welcome, what's your name?")  
9  
10 # Welcome George!
```



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- **For Loops**
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

For Loops

```
1 list1 =  
2 [1,2,3,4,5,6,7,8,9,10]  
3 for num in list1:  
4     print(num)  
5  
6 """  
7 1  
8 2  
9 3  
10 4  
11 5  
12 6  
13 7  
14 8  
15 9  
16 10  
17 """
```

While Loops

```
1 x = 0  
2  
3 while x < 10:  
4     print(x)  
5     x+=1  
6  
7 """  
8 0  
9 1  
10 2  
11 3  
12 4  
13 5  
14 6  
15 7  
16 8  
17 9  
18 """
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (`zip`, `enumerate`, `range`, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Built-in functions & Operators

- Range
- Enumerate
- Zip
- In
- Sorted
- Min & max
- ...



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

List Comprehensions

```
1 lst = [x**2 for x in range(0,11)]
2 print(lst)
3
4 # [0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- **Functions**
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Functions

```
1 def say_hello():
2     print('hello')
3
4
5 say_hello()
6
7 # hello
```



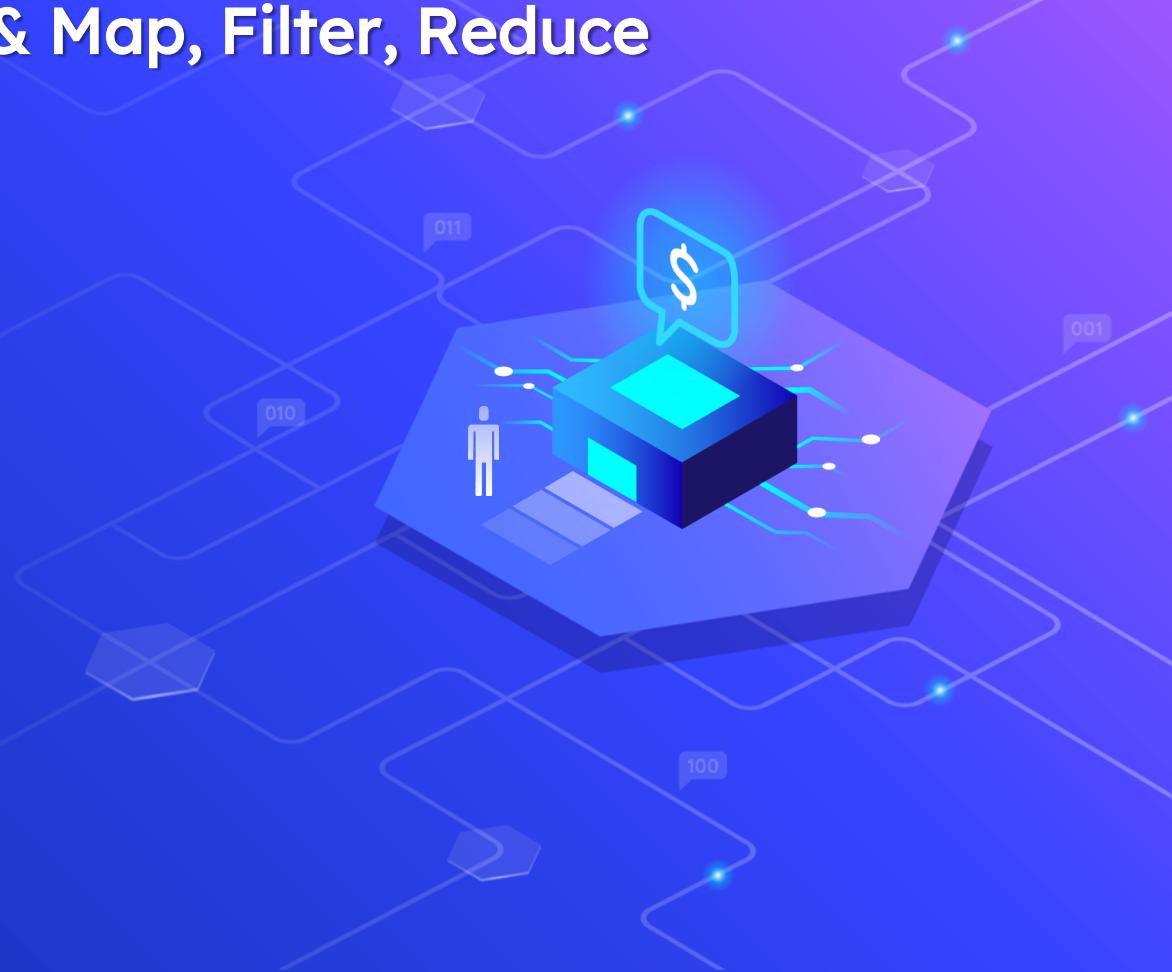
Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Lambda Expressions & Map, Filter, Reduce

- Map
- Filter
- Reduce
- Lambda Expression



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Variables Scope

- Local
- Global



Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
 - Numbers & Math
 - Boolean & Comparison and Logic
 - Strings
 - Lists
 - Tuples
 - Sets
 - Dictionaries

- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages



Modules & Packages

- Module is a file which contains various Python functions and global variables. It is simply just .py extension file which has python executable code.
- Package is a collection of modules.
- Library is a collection of packages.
- Framework is a collection of libraries.

Modules & Packages

- Random
- OS
- Datetime
- How to make your custom module
- How to make your custom package



Conda environments

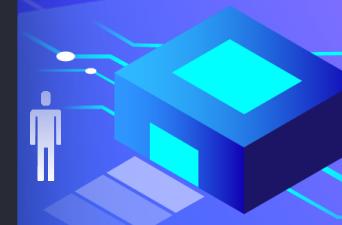
```
1 #show info  
2 conda info --envs  
3  
4  
5 # create environment  
6 conda create -n my-env python=3.7  
7 conda create -n my-env numpy requests  
8 conda create -n my-env python=3.7 numpy=1.16.1 requests=2.19.1  
9  
10  
11 # clone environment to a new one  
12 conda create -n myclone --clone my-env  
13  
14  
15 # activate, deactivate or remove  
16 conda activate my-env  
17 conda deactivate  
18 conda remove -n my-env --all  
19
```

Don't forget to add jupyter kernel for this newly created environment

<https://janakiev.com/blog/jupyter-virtual-envs/>

Conda environments

```
1 # generate requirements.txt from all environment packages  
2 pip freeze > requirements.txt  
3  
4  
5 # generate requirements.txt from project packages  
6 pip install pigar  
7 pigar  
8  
9  
10 # install packages from requirements.txt  
11 pip install -r requirements.txt
```



Time for Thanos.py >_



Questions ?!



Thanks!

>_ Live long and prosper

