## Class 01

#### Class Plan

- 1. Introduction
- 2. Syllabus
- 3. Python as our main language
- 4. Google Colaboratory (Colab) as the main platform
- 5. Python Output
- 6. Variables

# Object\_Oriented Programming for Business Applications

Data collection and preprocessing

Applications in Business

Python Libraries

## Syllabus

• M, W: 2:00 pm – 3:15 pm

#### Python compared to other languages

- 1. Easier to learn code syntax
- 2. Interpreted language, no need to compile, faster development
- 3. More than 100,000 packages (modules) available to use in Python (<a href="https://pypi.org/">https://pypi.org/</a>), and everything is open source.

#### Google Colab

Google Colab:

https://colab.research.google.com/

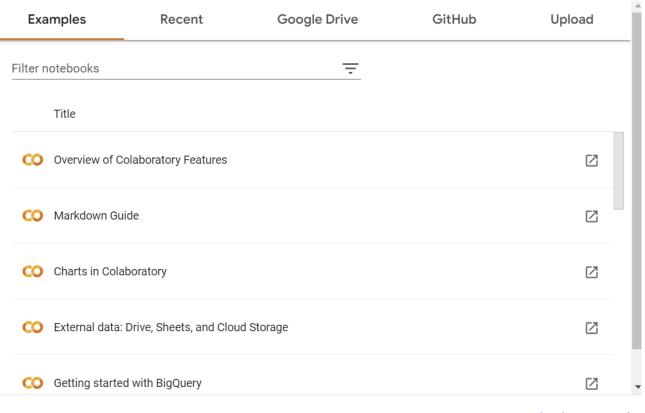
- Require a Google account
- No need to install Python or many packages
- Works the same on Windows and Mac
- Easy to share files
- You need to set up Autosave to save your work on you Google Drive

#### Intro to Google Colab

- Open a new notebook from inside Colab or Google Drive
- Rename your newly created notebook
- Create a "text" cell and add some content.
- Shift+Enter or Ctrl+Enter to run the text cell.

• Create a "code" cell, write a simple code and execute it (Numbers and Strings).

#### Intro to Google Colab



New notebook Cancel

### Google Colab

• Review code execution in Google Colab

#### Python Output

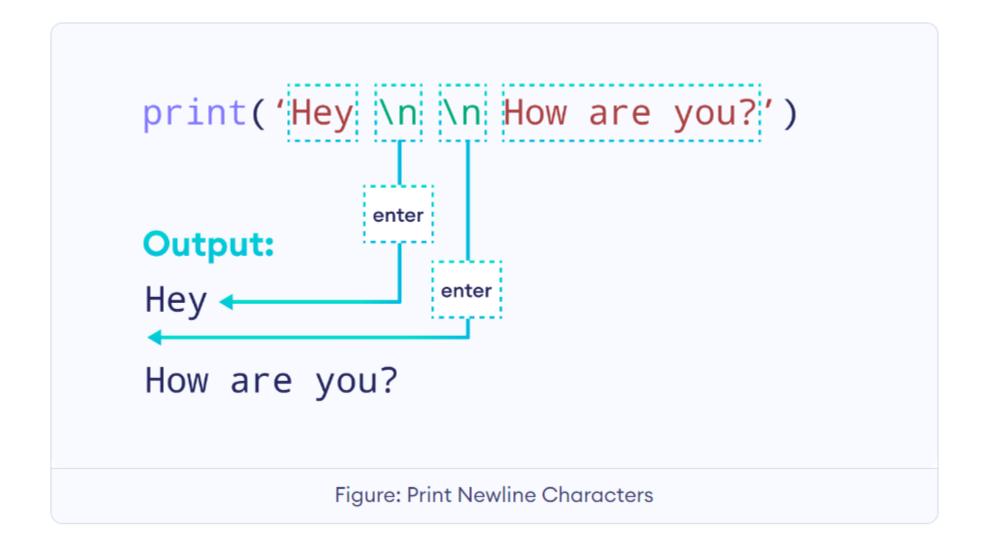
```
# print an integer
print(34) # Output: 34
# print a string
print('MIS 515') # Output: MIS 515
# print multiple items at once
print(22, 'Batman') # Output: 22 Batman
```

#### Print

Print Output

```
print(67)
print(67, -5)
print('Hey.', 'Are you enjoying the course?')
print(Hey there)
print('Hey')
print('How are you?')
```

#### Print Newline (Enter Key)



#### Variables

Variables are containers to store data like numbers and strings.

#### **Assignment**

#### Printing Variables

```
age = 25
print('age') # Output: age
print(age) # Output: 25
```

#### Practice

Problem Description:

- Create a variable named salary and assign 4950.5 to it.
- Print the variable.

## 8/28/2023

## Class 02

#### Class Plan

- 1. Data types
- 2. Conditions
- 3. Iterations

## Questions

#### Changing Values Stored in Variables

age = 25

print(age)

age = 26

print(age)

age is 25

25 is printed

age is 26

26 is printed

#### Practice

#### Problem description

- Declare a variable named country
- Assign string 'United States' to it
- Print the variable
- Change the value of the variable to 'Canada'
- Print the variable again

#### Value of One Variable to Another

```
print(color1) # Output: blue

color2 = "pink"

# Assigning the value stored in color2 to color1

color1 = color2
print(color1) # Output: pink
```

color1 = "blue"

#### Variable Names



## Notebook Time!

- Numbers
- Strings
- Boolean
- Collections

Integer (int): Represents whole numbers.

- age = 25
- quantity = 10

Float (float): Represents decimal numbers.

pi = 3.14159

price = 19.99

**String (str)**: Represents sequences of characters enclosed in single or double quotes.

- 'This is a string.' But be careful with sentences like this: Let's go!
- "This is also a string."

- name = "Ellie"
- message = 'Hello, world!'

Boolean (bool): Represents a binary value - True or False.

- is\_valid = True
- has\_permission = False

#### Collections

**List:** Represents an ordered collection of elements. A set of data enclosed in [] where elements are separated by commas.

#### Lists

Access list items by index

- fruits = ["apple", "banana", "orange"]
- numbers = [1, 2, 3, 4, 5]

List items do not have to be of the same type

school = ["Book", "Laptop", 12, 'coffee']

#### Collections

• Tuple:

Similar to a list but immutable (cannot be changed after creation). Elements are enclosed in ().

- coordinates = (5, 10)
- rgb\_color = (255, 0, 0)

#### Tuples

•Example: fruits = ("apple", "banana", "orange")
Elements are separated by commas and enclosed in parentheses.

Tuples can contain different data types: strings, numbers, etc.

•Similar to list, accessing tuple elements by Indexing: Elements can be accessed using their position (index)

•Example: **fruits[0]** returns "apple"

#### Collections

• Dictionary: Represents a collection of key-value pairs.

Similar to Lists but elements are enclosed in {} and accessed using a key/value pair.

```
person = {"name": "Bob", "age": 30, "city": "New York"}
grades = {"Math": 95, "History": 85, "Science": 78}
```

#### Dictionaries

A Dictionary allows you to store key, value pairs. It is also known as:
 Maps, Hash tables, Associate Arrays

#### Example:

• a real-life dictionary where you can look up a word (key) to find its corresponding definition (value). Dictionaries are mutable, meaning they can be modified after creation.

There are other types, too. Learn more? Set, None Type, Complex, ...

in Python, you can use the type() function to determine the data type of a variable. For example:

- value = 42
- print(type(value)) # Output: <class 'int'>

## Notebook Time!

#### Conditions

Use if statements to control which code is executed.

If a statement is True:

do something

Else:

do something else

==, !=, >, >=, <, <=, and, or,

## Notebook Time!

#### Iterations (Looping)

#### While loop:

The while loop will execute the block of code over and over until the conditional statement after the word "while" is true.

#### For loop:

The for loop is typically used in two situations. The first is when you know how many time you want to execute a loop. The second is when you want to loop through a collection.

## Notebook Time!