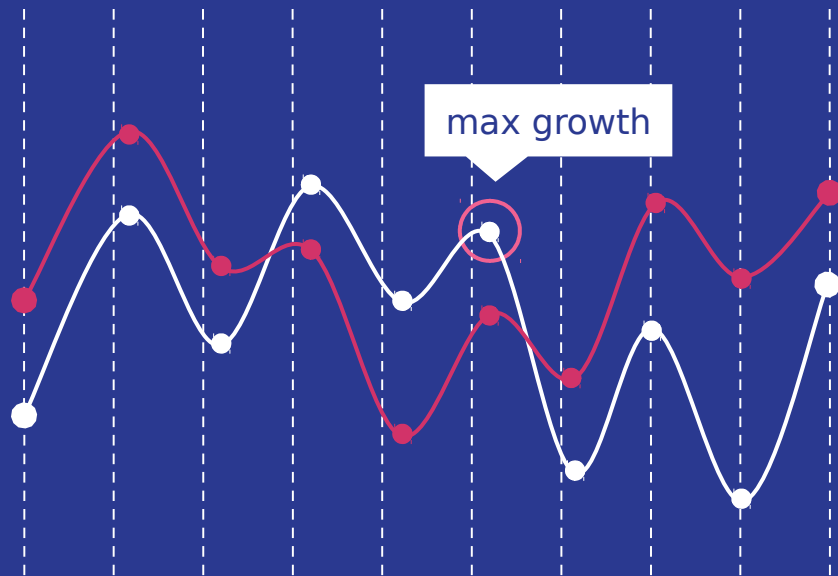




Introduction to Python

CMP 201 (2019/2020)

SORINOLU, Babafemi Gabriel




LESSON 7:

Be a Ninja Coder!

Objectives

The aim of this lesson is to introduce the **python programming language**.

Content(Week

- 
- **Functions**
 - Exception handling
 - File handling

Python Functions:

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result



Creating a function:

a function is defined using the def keyword

```
#Get the value of the "model" key
```

```
def my_function():  
    print("Hello from a function")
```

Calling a function:

a function is called by writing the function name followed by it's parenthesis

#Function definition

```
def my_function():  
    print("Hello from a function")
```

#Function calling

```
my_function():
```

PUZZLE:

```
def func(x):
```

```
    return x + 1
```

```
f = func
```

```
print ( f(2) + func(2) )
```


Function parameters:

Information can be passed to functions as parameter.

Parameters are specified after the function name, inside the parentheses.

You can add as many parameters as you want, just separate them with a comma.

The following example has a function with one parameter (fname).

When the function is called, we pass along a first name, which is used inside the function to print Hello to the name

Function parameters:

```
def say_hello(fname):  
    print( "Hello "+ fname)
```

```
say_hello ("Tobi")  
say_hello("Pelumi")  
say_hello ("John")
```

Default parameters values:

The default parameter value allows a value to be used when a parameter is not passed to the function.

```
def my_function(country = "Norway"):  
    print("I am from " + country)
```

```
my_function("Sweden")  
my_function("India")  
my_function()  
my_function("Brazil")
```

Passing a list parameters:

You can send any data types of parameter to a function (string, number, list, dictionary etc.), and it will be treated as the same data type inside the function. E.g. if you send a List as a parameter, it will still be a List when it reaches the function:

```
def my_function(countries):  
    for country in countries:  
        print("I am from " + country)  
  
my_list=["Nigeria", "Togo", "Sweden", "India"]  
my_function(my_list)
```

Returning values :

A function can return a value after being called. Such functions are referred to as **return type function**.

```
def my_function(x):  
    return 5 * x
```


```
print(my_function(3))  
print(my_function(5))  
print(my_function(9))
```

Exception handling:

When an error occurs, or exception as we call it,

Python will normally stop and generate an error message.

These exceptions can be handled using the try statement


A solid blue horizontal bar located at the bottom left of the slide.

Try .. Except:

The **try** block lets you test a block of code for errors.

The **except** block lets you handle the error.

The **finally** block lets you execute code, regardless of the result of the try- and except blocks.



Try .. Except:

The try block will generate an exception, because x is not defined

```
try:  
    print (x)  
except:  
    print("An exception occurred")
```

Since the try block raises an error, the except block will be executed.
Without the try block, the program will crash and raise an error

File handling:

File handling is an important part of any web application.

Python has several functions for creating, reading, updating, and deleting files.

The key function for working with files in Python is the `open()` function.

A solid blue horizontal bar located at the bottom left of the slide.

File handling:

The open() function takes two parameters; filename, and mode. There are four different methods (modes) for opening a file

"r"	Read - Default value. Opens a file for reading, error if the file does not exist
"a"	Append - Opens a file for appending, creates the file if it does not exist
"w"	Write - Opens a file for writing, creates the file if it does not exist
"x"	Create - Creates the specified file, returns an error if the file exists

File handling:

In addition you can specify if the file should be handled as binary or text mode

"t"	Text - Default value. Text mode
"b"	Binary - Binary mode (e.g. images)

File handling:

To open a file for reading it is enough to specify the name of the file:

Syntax:

```
f = open("demofile.txt")
```

#"r" for read, and "t" for text are the default values

```
f = open("demofile.txt", "rt")
```

Reading Files:

The `open()` function returns a file object, which has a `read()` method for reading the content of the file:

Syntax:

```
f = open("demofile.txt")  
print(f.read())
```

Reading part of files:

By default the `read()` method returns the whole text, but you can also specify how many characters you want to return:

Syntax:

```
f = open("demofile.txt")  
print(f.read(10))
```

Reading lines:

You can return one line by using the `readline()` method:

Syntax:

```
f = open("demofile.txt")  
print(f.readline())
```

By calling `readline()` two times, you can read the two first lines:

Reading line by line:

By looping through the lines of the file, you can read the whole file, line by line :

Syntax:

```
f = open("demofile.txt")  
for x in f:  
    print(f.readline())
```


Closing files:

It is a good practice to always close the file when you are done with it :

Syntax:

```
f = open("demofile.txt")  
    print(f.readline())  
f.close()
```

Note: You should always close your files, in some cases, due to buffering, changes made to a file may not show until you close the file.

Closing files:

It is a good practice to always close the file when you are done with it :

Syntax:

```
f = open("demofile.txt")  
    print(f.readline())  
f.close()
```

Note: You should always close your files, in some cases, due to buffering, changes made to a file may not show until you close the file.

Writing files:

To create a new file in Python, use the `open()` method, with one of the following parameters:

"x" - Create - will create a file, returns an error if the file exist

"a" - Append - will create a file if the specified file does not exist or will append to the end of the file if it does exist

"w" - Write - will create a file if the specified file does not exist or will overwrite if it does exist

Creating new files:

Example:

#a new empty file is created , returns error if it exist

```
f = open("demofile2.txt", "x")  
f.write("Now the file has more content!")  
f.close()
```

Task

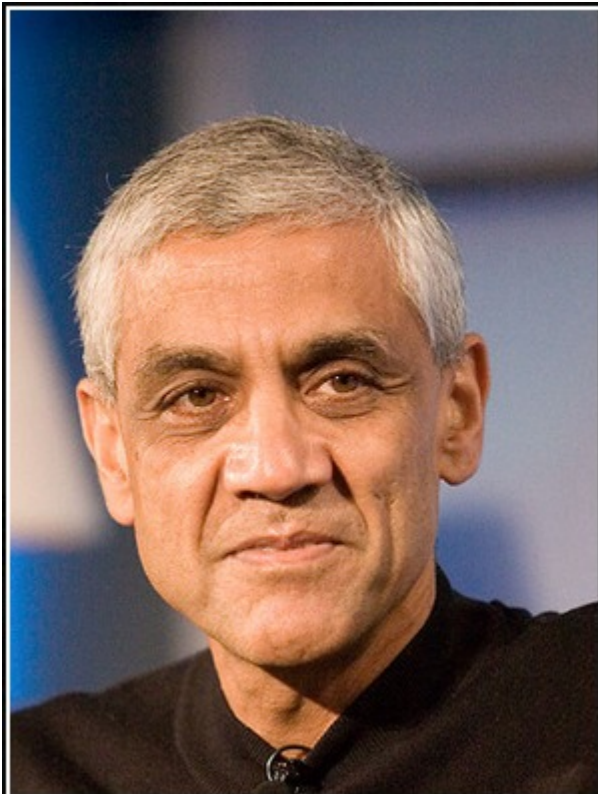
To do in class



EXERCISE

(Uniqueness in code earns extra credit).

- 1. Each question should be kept in a single .py file**
- 2. then all zipped in a file**
- 3. with your matric No. as name of the file**



Doctors can be replaced by software
– 80% of them can. I'd much rather
have a good machine learning
system diagnose my disease than
the median or average doctor.

— *Vinod Khosla* —

AZ QUOTES