1. Introduction

Goal of Project:

Learn as much as possible about basic python

TABLE OF CONTENTS

2. Body

Need to show	How to show
Compiling and Debugging	Console
	.Py File
	Jupyter Labs
Basic programing Structures	Variables
	Operations
	Whitespace

Need to show	How to show
	imports
	Scope
Data structures	Arrays
Functions	
Input	
Output	

2.1. Basic Information

This solution simply presents the basic information of python AND will not have any asociated code.

2.1.1. Building and Compiling

Runing python is different to other languages because you can run python through multiple means.

2.1.1.1. Command Line Python Interpreter

Python can be run in the command line. To get started type **py** into the command line.

You will then see the following text in your command line, informing you have just launched the command line python interpreter.

> Python 3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32

> Warning:
> This Python interpreter is in a conda environment, but the environment has
> not been activated. Libraries may fail to load. To activate this environment
> please see https://conda.io/activation

> Type "help", "copyright", "credits" or "license" for more information.

If you did not see this text it is advised that you check your instalation of python.

Here you can type any line of code such as

```
>>>print("Hello World")
Hello World
```

2.1.1.2. .py Files

Another way to work with python is to write the code into a text file with the extension .py. Any .py file can be quickly read.

One example is the code below

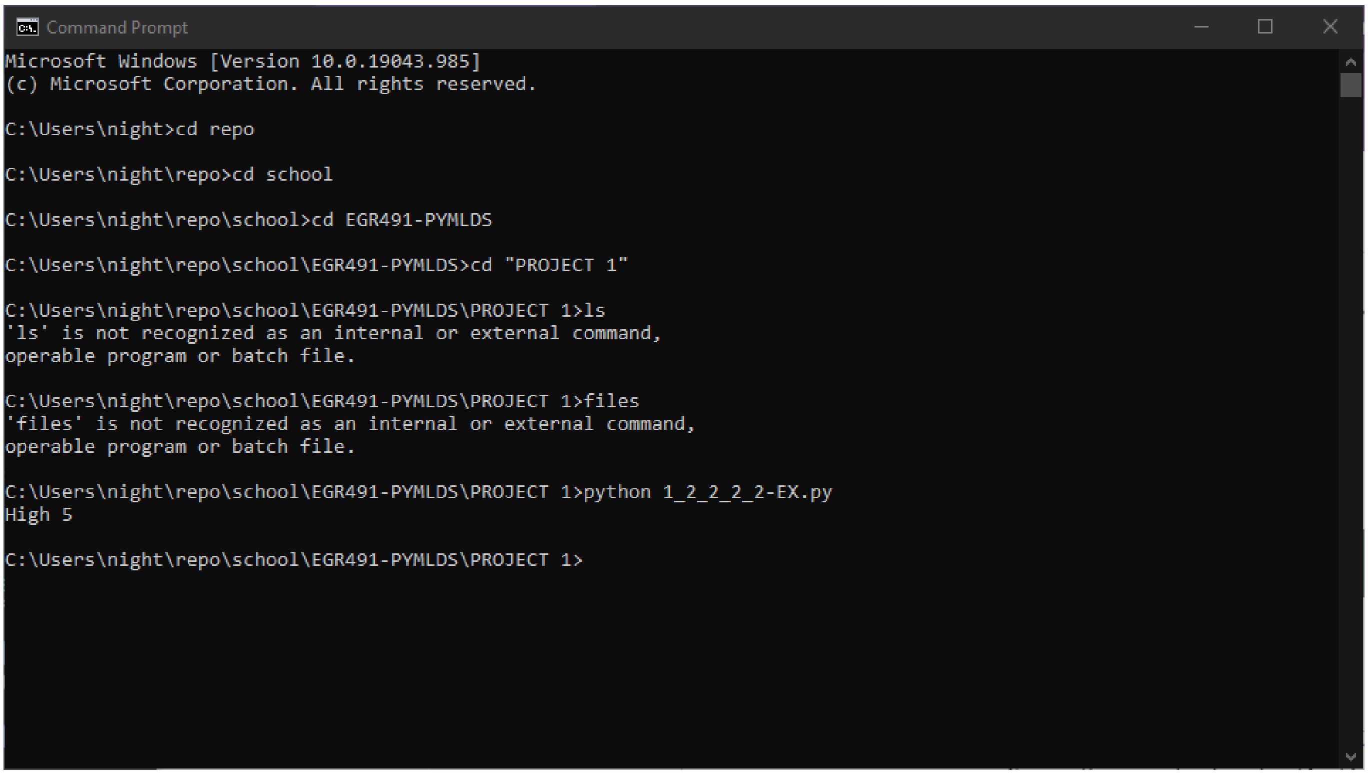
```
x = 0
x += 1
x *= 5
print("High", 5)
```

This will output

```
High 5
```

There are two ways to run these files,

1. Open the File in an IDE such as Juyyter Lab IDE or Visual Studio Code
   1. Right Click the file and select OPEN WITH
   2. Select the IDE of your choice
   3. These IDE's will allow you to run the file. In VSCode you will need to install the proper extension to run python files.
2. Run the files via command line.
   1. Loacte the file in command line (you can use cd < folder name > if it is not in your root direcotry)
   2. Type python < file name >
   3. A python interpreter will launch showing your output

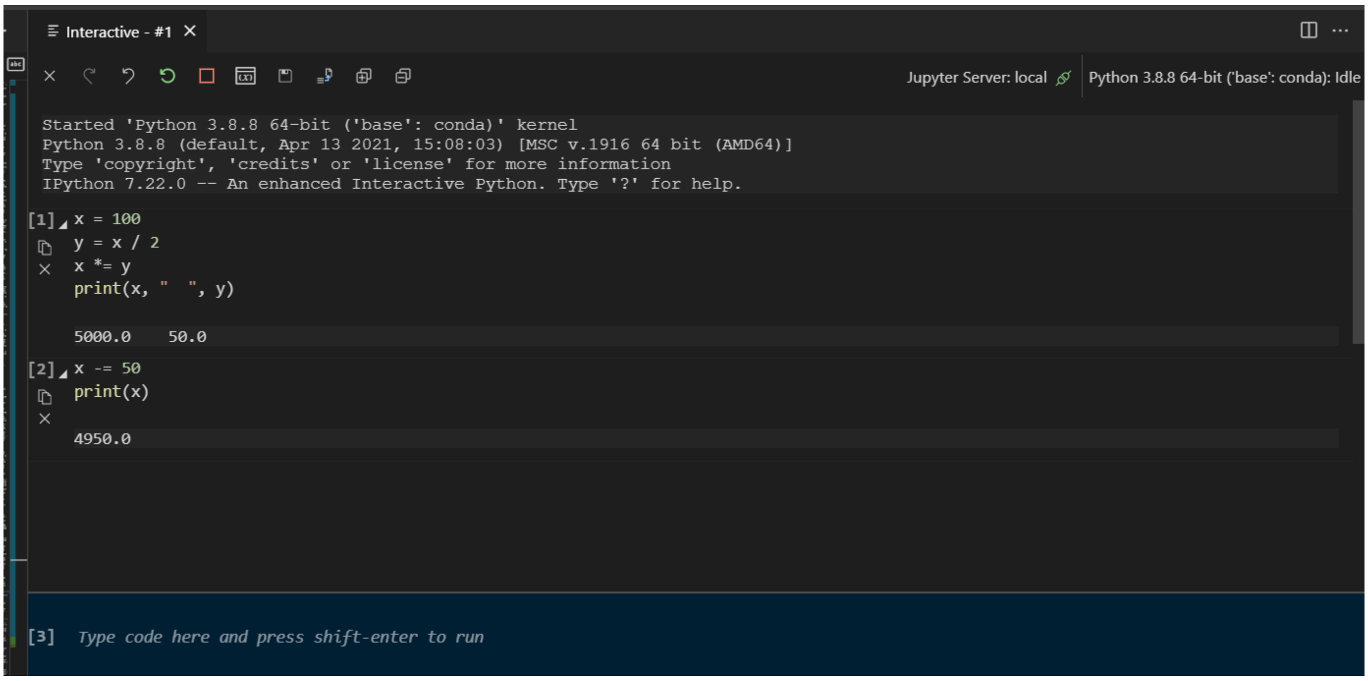


```
Microsoft Windows [Version 10.0.19043.985]
(c) Microsoft Corporation. All rights reserved.

C:\Users\night>cd repo

C:\Users\night\repo>cd school

C:\Users\night\repo\school>cd EGR491-PYMLDS

C:\Users\night\repo\school\EGR491-PYMLDS>cd "PROJECT 1"

C:\Users\night\repo\school\EGR491-PYMLDS\PROJECT 1>ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\night\repo\school\EGR491-PYMLDS\PROJECT 1>files
'files' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\night\repo\school\EGR491-PYMLDS\PROJECT 1>python 1_2_2_2_2-EX.py
High 5

C:\Users\night\repo\school\EGR491-PYMLDS\PROJECT 1>
```

2.1.1.3. Advanced Interpreter

There are other interpreters and IDE's for python. The best of these come in the conda package which includes the Spyder IDE and Jupyter Labs interpreter.

The difference between an IDE and Interpreter when it comes to python is that the IDE is more tradional. You write your code then you run it. In an interpreter you can write and run your code line by line or in short segments. Below is an example image from the Juypter Labs extension for Visual Studio Code.



```
Started 'Python 3.8.8 64-bit ('base': conda)' kernel
Python 3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.22.0 -- An enhanced Interactive Python. Type '?' for help.

[1] x = 100
    y = x / 2
    x *= y
    print(x, "  ", y)

    5000.0    50.0

[2] x -= 50
    print(x)

    4950.0

[3] Type code here and press shift-enter to run
```

2.2. Syntax

Python has a very lax syntax compared to many languages. The worst or best of these is that python does not require a semicolon at the end of each line. But there are a few differences between python and languages like Java.

2.2.1. White Space

In Java white space is generally ignored and is simplied used to increase readability. But python white space is used to distinguis blocks of code. For example the code below is invalid.

```
def sayHi():
print("hi")
```

And the output of the file makes it clear that line 2 is borken.

```
  File "c:/Users/night/repo/school/EGR491-PYMLDS/examples/2-2_exmpl.py", line 2
print("HI")
^
```

To fix this simply add a couple spaces before your

```
def sayHi():
  print("Hi")
```

Another attribute of python is that every funtion and variable's scope begins at its defintion in the file or interpreter.

2.3. Variables

Python is an object oriented programing lanuages and all variables avalible are objects. Unlike Java python variables are acutally pointers. Because of this it does not matter the type of your variable. Consider the code below,

```
x = 5 # x is an integer
x = 'BONJOUR' # x is a string
```

Those familiar with languages like Java and Ada will know that this does not happen normally. This is because of variables in python being pointers.

2.3.1. Built in Scalar Types

Python includes common built in types such as

- int
- float
- bool
- str (String type)

Python also includes a few uncomon types

- complex: Complex Number such as 1 + 1j
- NoneType: This represents a Null value

2.3.2. Data Structures

Arrays are ubiqutus to programing and are neccesary for many applications. Thus many languages have many types of arrays and other related data structures.
Pythons four built in data stuctures cover many of the needs of data science and mathmatics.

- **Lists** are common ordered lists and can be assed by using a command separated list with ","'s between each item between squre brackets.
- **Tuples** are immuatable (unchanging) list. This is assigned list a normal list but with
- **Sets** This are unordered lists but sets can only have unique values. To assign a set put a comma separated list between curly brackets {}
- **Dicts** or dictionaries are sets with contain a key value mappings. to assign a dict to a variable each key value parir will be setparated by commands with a colon between the key and value like this "x = {'key1':1,'key2':2}"

Below is an example of how to use and access each of these arrays

2.3.2.1. Examples

```
lst = [1,"two",3, [4,5]]

print(lst)      # Prints the whole list
print(lst[0])   # Prints list item 1
print(lst[-1])  # Prints the last Item of the list
print(lst[2:])  # Prints Items from index to on
print(lst[:3])  # Prints Items 0 to three
print(lst[1:3]) # Prints Items 1, 2 and 3
print(lst[-3:]) # Prints the last three items
lst.append("THIS WORKS")
print(lst)
```

OUTPUT

```
[1, 'two', 3, [4, 5]]
1
[4, 5]
[3, [4, 5]]
[1, 'two', 3]
['two', 3]
['two', 3, [4, 5]]
[1, 'two', 3, [4, 5], 'THIS WORKS']
```

```
tup = (0,1,2)
# you can do the same things as you do with lists
print(tup)
print(tup[1])
# you cannot modify the tupple in any way
tup[1] = 7 # This gives a TypeError because
```

OUTPUT

```
(0, 1, 2)
1
Traceback (most recent call last):
  File "c:/Users/night/repo/school/EGR491-PYMLDS/PROJECT 1/Example3.py", line 6,
in <module>
    tup[1] = 7
TypeError: 'tuple' object does not support item assignment
```

```
setObj = {0,1,2}
#The Same as a tupple or a list
print(setObj)
```

```
# But you do have access to unique operations
print(setObj.union({2,3,4,5})) # Union: outputs all items contained in the sets
duplicates are ignored you can also use |
print(setObj.intersection({1,7,9})) # Intersection: outputs the items shared in
each set: you can also use &
print(setObj.difference({0,2})) # Difference: outputs the items in setObj  but not
in the interal set you can also use -
print(setObj.symmetric_difference({0,2,9})) #Symetric Difference outputs items
that are in either list but not both you can also use ^
```

OUTPUT

```
{0, 1, 2}
{0, 1, 2, 3, 4, 5}
{1}
{1}
{9, 1}
```

```
dic = {1:"THIS",2:"THAT",3:"NOT"}
pritn(dic[1]) # In this case you must access by a key not by the index
#Adding a new key value pair
dic["SWITCH"] = 99
print(dic)
print(dic["SWITCH"])
```

OUTPUT

```
THIS
{1: 'THIS', 2: 'THAT', 3: 'NOT', 'SWITCH': 99}
99
```

2.4. Math

In this solution we will show the basic mathmatical operations of python. This include the following:

Operation		
+	Addition	Adds the two operators
-	Subtraction	Subtracts the first operator from the second
*	Multiplication	Multiplies the two operators
**	Power	Raser the first operator to the second
/	Nomal Dvision	Divides the frist operator by the second

Operation		
//	Integer Division	Divides the first operator by the second with no remander

- CODE

```
x = 7
print(x)
y = x + 5 # y = 12
print(y)
y = x * 2  # y = 14
print(y)
y = x ** 2 # y = 49
print(y)
y =  x / 2 # y = 3.5
print(y)
y = x // 2 # y = 3
print(y)
```

- OUTPUT

```
7
12
14
49
3.5
3
```

2.5. Logic and Loops

2.5.1. Conditional Statments

Python conditional staments uses the following keywords

- *if*: opening statment for a logic block
- *elif*: the quivalent to else if in other languages
- *else* The final else stments

Python includes server bitwise operators that can be used in logic statments such as

&	AND
|	OR
^	XOR
~	NOT

If statments are formed as follows

```
if x > 100:
  print("x is greater than 100")
elif x == 100:
   print("x is 100")
else:
  print("x is less than 100")
```

2.6. For loops

A python for loops looks like the code below,

```
for x in range(90, 100): #Prints all values from 90 to 100
    print(x)
```

2.7. While Loops

2.8. Input and Output

2.8.1. Screen

2.8.2. Files

2.9. Functions

A function in python is declared by using the *Def* keyword followed by the name of your function, then place all parameters by name in a list after

2.10. Imports

In Java you can get predesigned packages by doing, "import package_name."
In python there are not packages but **modules** which must first be installed. There are some good preinstalled such as numpy adds extended mathmatical abilities.

If you wish to use thrid party modules you follow the same syntax just ensure that they are installed. The most common way to instal python modules is using pip, below is an example of how to install the pygame modules which adds fun ways to make games.

```
pip install pygame
```

A warning to those who are just installing python for the first time. Make sure to install python to the PATH during the install process this will make using pip much easier.

Code Project