ISA 414 – Managing Big Data

Lecture 2 – Preliminaries (Part I)

VS Code, Jupyter Notebooks, Markdown

Dr. Arthur Carvalho

arthur.carvalho@miamioh.edu

MIAMI UNIVERSITY

Lecture Objectives

- Learn the basics of VS Code
- Learn about Jupyter Notebooks
- Learn basic Markdown notation

Introduction to Python

- What is Python?
  - General-purpose programming language
  - Contrast the above to R
    - An environment for statistical computing, data visualization, and data analytics

- Python is (arguably) the most popular programming language for data analytics
  - The programming language we use in this course

Introduction to Python

- But what is a programming language?
  - It is a **formal language** used to program computers
    - **Syntax**: set of rules that define the combinations of symbols that are considered to be correctly structured in a language
    - **Semantics**: meaning of constructs in a language

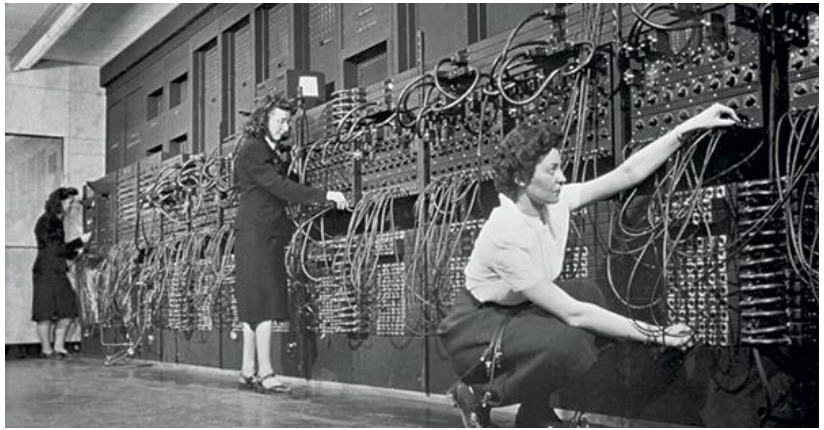
- Unlike **natural languages**, formal languages are very rigid
  - Any syntax violation will produce errors

Introduction to Python

- How can one use a language to program a computer? (CS 101)
  - Computers only understand 0s and 1s
    - Absence/presence of electricity
  - Sequences of 0s and 1s define numbers
    - *E.g.*, 0000000000000001 = 1, 0000000000000010 = 2
  - The central processing unit (CPU) of a computer contains arithmetic logic unit (ALU), capable of performing arithmetic operations
    - Arithmetic: basis of modern mathematics

Introduction to Python

- How can one use a language to program a computer?
  - Turn electricity on/off
    - Produce sequences of 0s and 1s
    - Not very productive



Introduction to Python

- How can one use a language to program a computer?
  - Low-level programming language
    - Easier to understand than 0s and 1s, but still incredibly hard to work with
    - Very strong correspondence between the language and machine code instructions
    - Very efficient
    - *E.g.*, Assembly X86

```
pushl   %ebp            // push ebp onto stack
movl    $1, %eax        // eax=1 (return value if n==0)
movl    %esp, %ebp      // ebp now points to top of stack
subl    $8, %esp        // allocate 8 bytes on top of stack
movl    %ebx, -4(%ebp)  // store ebx on stack (see .L1 for restoring)
movl    8(%ebp), %ebx   // ebx = n (1st argument of function)
testl   %ebx, %ebx      // if n==0, set zero flag (0 flag means equal, so no jump if n==0)
jne     .L5
```

Introduction to Python

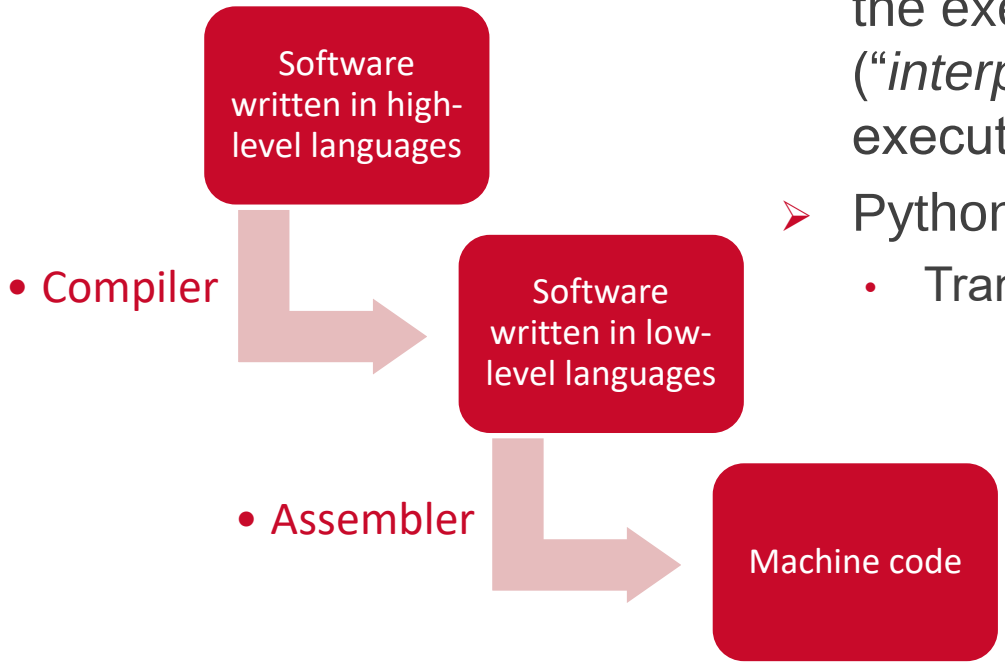
- How can one use a language to program a computer?
  - High-level programming language
    - Easier to understand than machine code and low-level languages
    - Low correspondence between the language and machine code instructions
    - Not as efficient as low-level languages, but easy to program
    - *E.g.*, Python:

```
x = 5
if x > 0:
    print("Non-negative number")
else:
    print("Negative number")
```

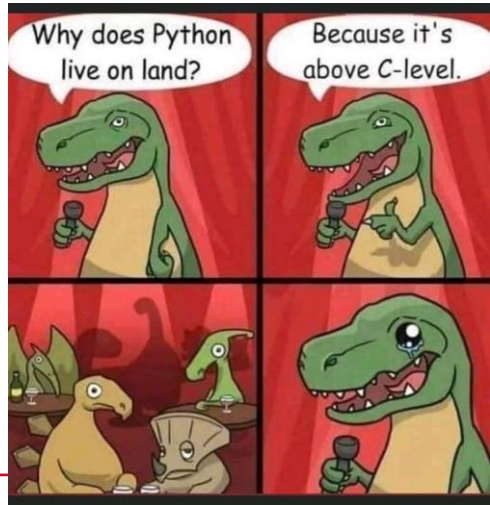
Introduction to Python

- How can one use a language to program a computer?
  - Software written in low-level programming languages are more efficient, but more difficult to program
    - Closer to machine code
  - Software written in high-level programming languages must be translated to low-level languages by specialized software
    - *Interpreters, Compilers,* and *Assemblers*

Introduction to Python



- This translation process can happen during the execution of the software ("*interpretation*") or only once before execution ("*ahead-of-time compilation*")
- Python is a high-level interpreted language
  - Translation "on-the-fly"



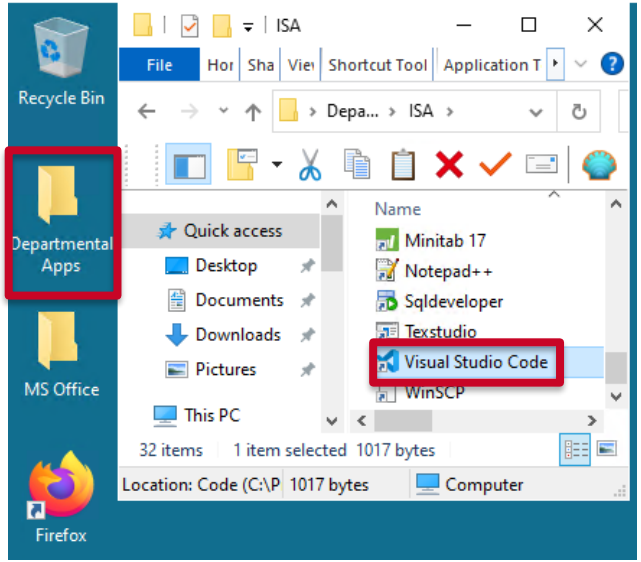
Introduction to Python

- It is commonplace to use specialized software known as **IDE** (Integrated Development Environment) to help with programming tasks
  - The *de facto* IDE for R is called **RStudio**
  - We will use the IDE called **Visual Studio (VS) Code** for Python
- Python and VS Code are already installed on the lab computers and remote desktop
- To run Python and VS Code on your own computer:
  1. Download and install Python (version 3.8+)
     - https://www.python.org/downloads/
  2. Download and install VS Code
     - https://code.visualstudio.com/download

INTRO TO VS CODE

Introduction to VS Code

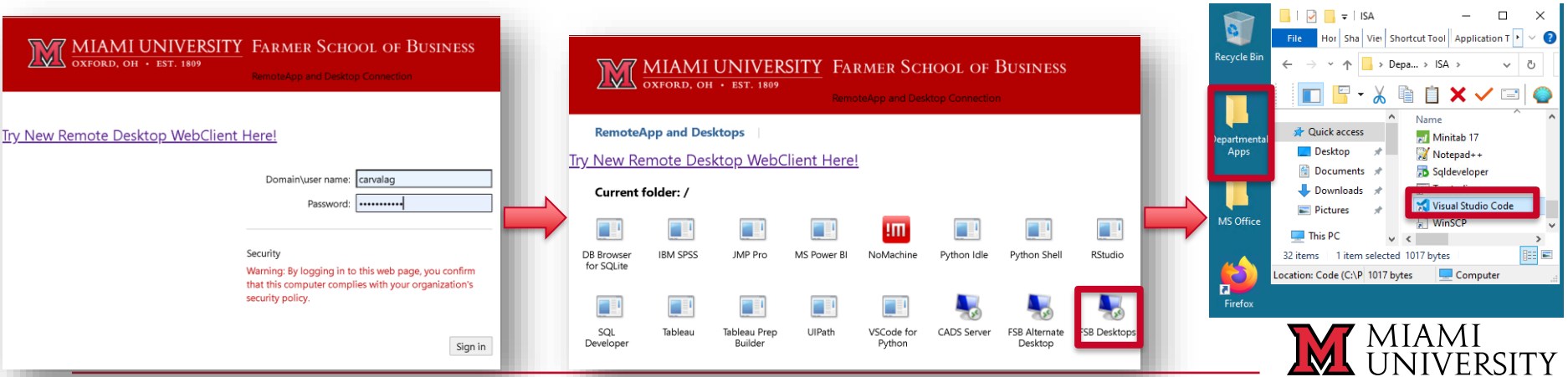
- Open VS Code now
  - Open the file at "*Departmental Apps*" -> *ISA* -> "*Visual Studio Code*"



MIAMI UNIVERSITY

Introduction to VS Code

- You can also access VS Code remotely
  - Go to https://virtualpc.fsb.miamioh.edu/ and sign in
  - Select "*FSB Desktops*"
  - Open the downloaded file and sign in again
  - Open VS Code at "*Departmental Apps*" -> *ISA* -> "*Visual Studio Code*"

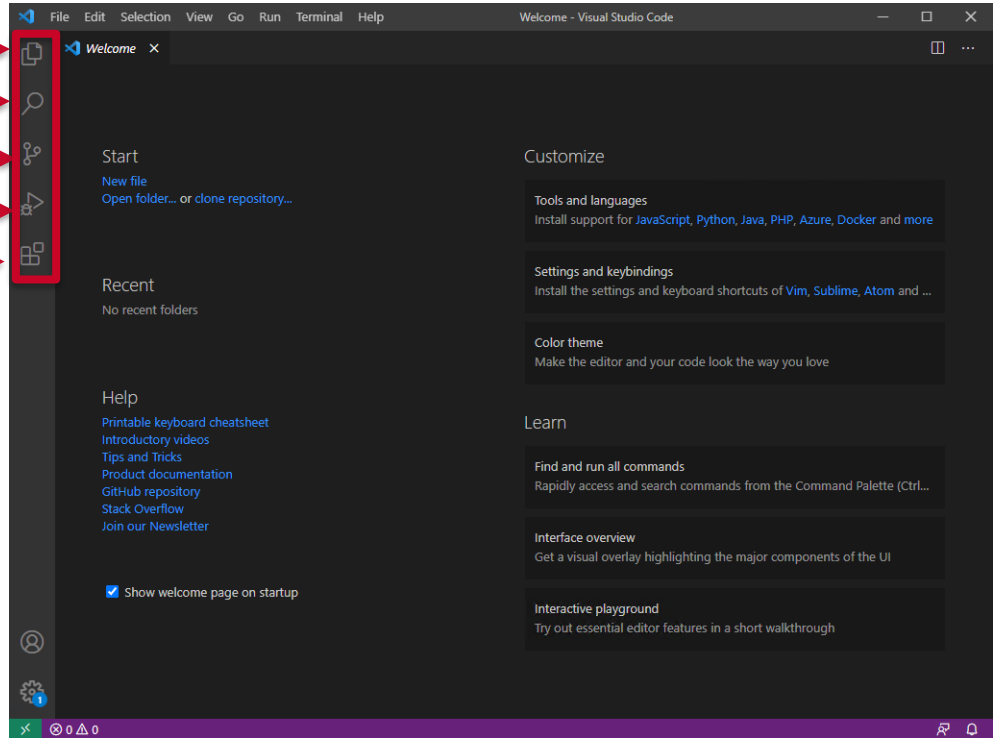


MIAMI UNIVERSITY

Introduction to VS Code

- This is the initial screen at the time of writing

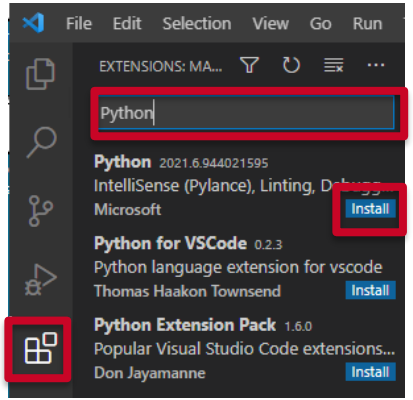
Project explorer
Search
Source Control
Run and Debug
Extensions

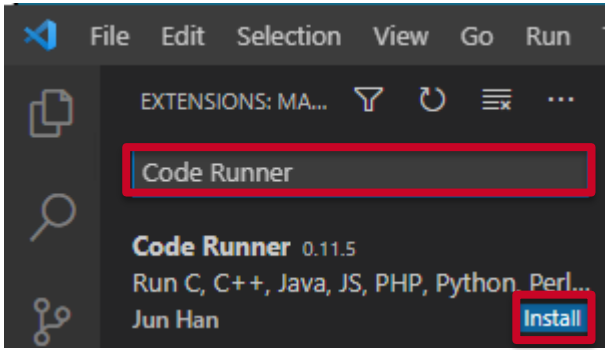


MIAMI UNIVERSITY

Introduction to VS Code

- VS Code can be used to program in virtually any programming language
  - The appropriate extensions must be installed first
  - Let's install three extensions: *Python*, *Jupyter*, and *Code Runner*
    - Click on the *Extensions* icon, type *Python*, and click on *Install*
    - Do the same for *Code Runner* and *Jupyter*





Introduction to VS Code

- You can always change the appearance of VS Code
  - For example, go to *File* -> *Preferences* -> "*Color Theme"* to change the background color

- You can always search for commands using the *search bar*
  - Press F1 for the search bar to appear

INTRO TO PYTHON IN VS CODE

Introduction to Python

- We will learn how to run Python programs in three different ways
  - Non-interactively
  - Interactively
  - Jupyter notebooks

1) Non-Interactive Python

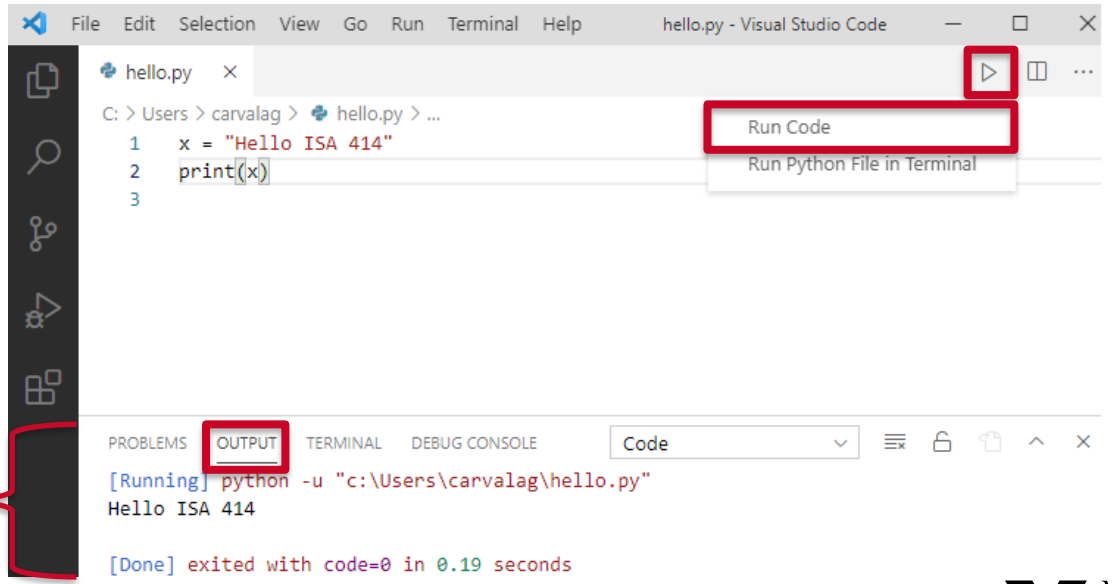
- Let's create our first python file
  - Go to *File* -> "*New File*"
  - Type the following:

```
x = "Hello ISA 414"
print(x)
```

  - Go to *File* -> *Save*
    - Save the file as *hello.py*
    - Because of the *.py extension*, VS Code now recognizes this is a Python file

1) Non-Interactive Python

- Let's run our first Python code now
  - Click on the "play" button (top right) and select "*Run Code*"



output from our code

MIAMI UNIVERSITY

1) Non-Interactive Python

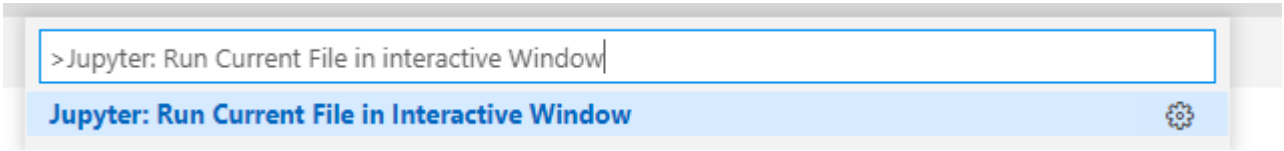
- The previous procedure will run the entire code
  - One can run only a few lines by selecting these lines and running the code
  - Highlight the first line and run the code again
    - Nothing is printed because the print function was not selected
  - What happens if one subsequently highlights the second line and run the code again?
    - An error occurs
    - *x* is not defined

```
[Running] python -u "c:\Users\carvalag\tempCodeRunnerFile.py"
Traceback (most recent call last):
  File "c:\Users\carvalag\tempCodeRunnerFile.py", line 1, in <module>
    print(x)
NameError: name 'x' is not defined
```

2) Interactive Python

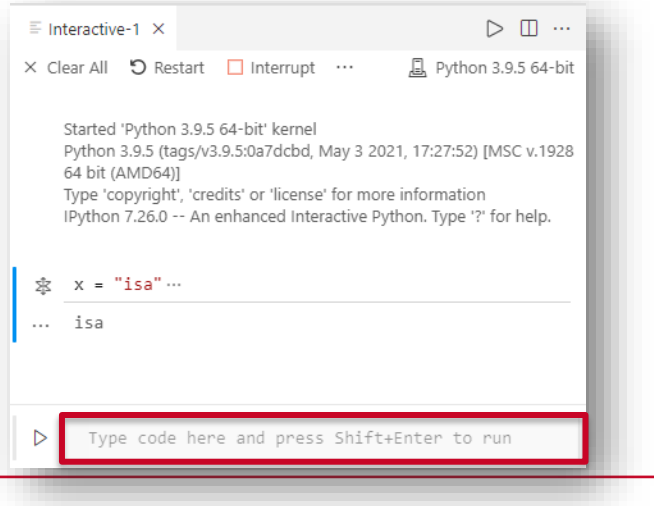
- The previous error happened because we ran Python non-interactively
  - The environment forgets everything in between runs

- Let's now run Python interactively
  - Press F1 and search for "*Jupyter: Run Current File in Interactive Window*"



2) Interactive Python

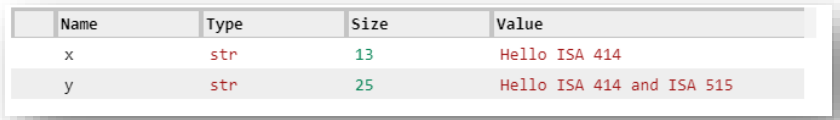
- You can not type commands at the bottom of the environment
  - For those familiar with R
    - This is similar to how the console in RStudio works



MIAMI UNIVERSITY

2) Interactive Python

- Let's test the previous environment
  - Type y = x + " and ISA 515" at the bottom of the environment
    - Press the keys *shift* + *enter* to run the code
    - See how the code now knows what *x* is
  - How do we know the current values of x and y?



Name	Type	Size	Value
x	str	13	Hello ISA 414
y	str	25	Hello ISA 414 and ISA 515

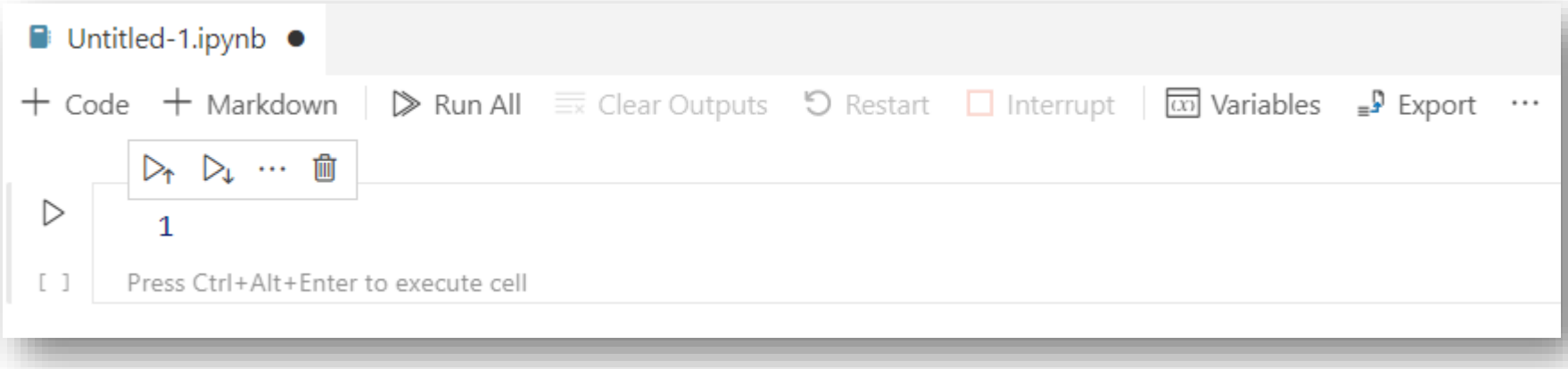
MIAMI UNIVERSITY

3) Jupyter Notebooks

- The third way of running Python applications will be through **Jupyter notebooks**
- What is a notebook?
  - Notebooks are incredibly popular in data science
  - Among many other things, they allow analysts to write well-formatted text, code, and results (graphs, tables, …) all in one place
    - This is great to enhance *readability* and *reproducibility*
- There are some notebooks in Python
  - We shall use Jupyter notebooks embedded in VS Code

3) Jupyter Notebooks

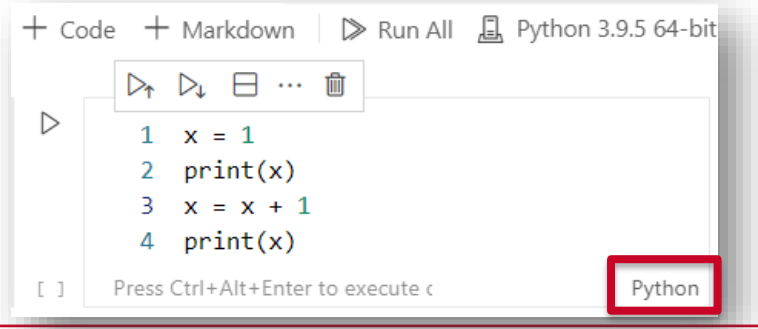
- Let's create our first Jupyter notebook
  - Press F1 and search for *Jupyter: Create New Blank Notebook*



3) Jupyter Notebooks

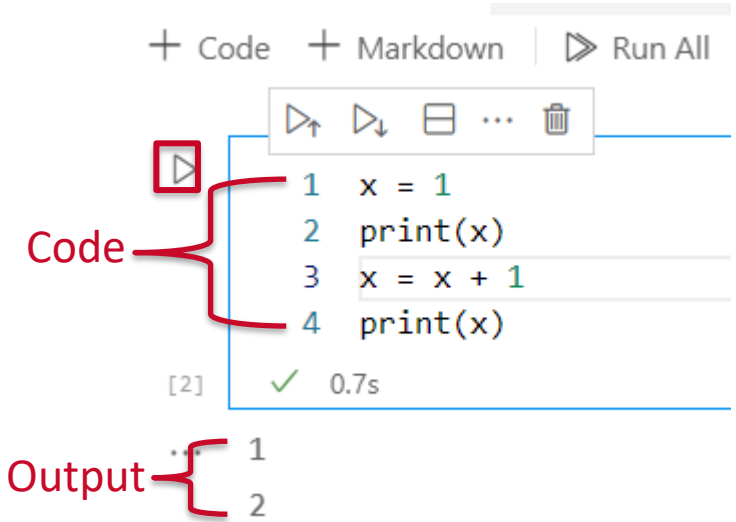
- Notebooks have **cells**
  - For our purposes in this course, each cell will be either text written in a language called **Markdown** or code written in **Python**
  - Our first cell is a Python cell
    - Let's add the following code to it

```
x = 1
print(x)
x = x + 1
print(x)
```



3) Jupyter Notebooks

- Click on the play button on the top-left of the cell to run the entire code (or press ctrl + enter)



- See how the code and output are close together

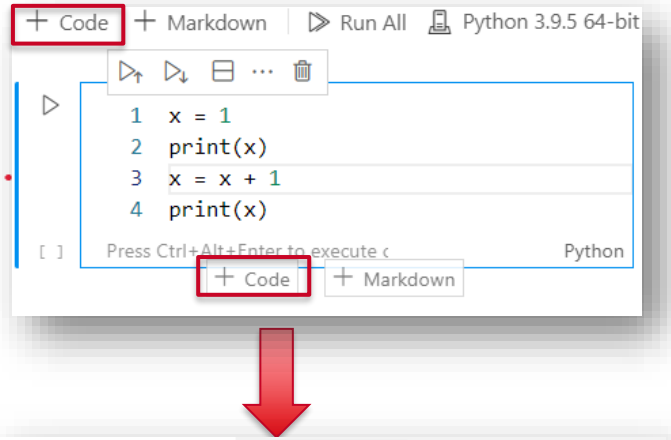
3) Jupyter Notebooks

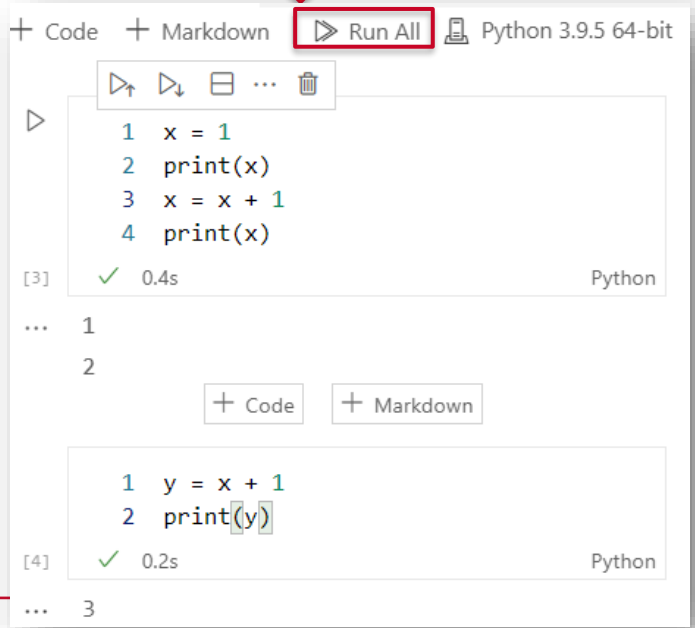
- Click on the “+ Code” button to add another cell to your notebook
- Add the following code to and run the second cell

  y = x + 1

  print(y)

- “Run All” cells
  - See how notebooks are interactive
  - A variable in one cell can be called from another





3) Jupyter Notebooks

- Understanding the top buttons

Add Python or Markdown code

Run all cells

Clear all the produced outputs

Restart session (erase all variables)

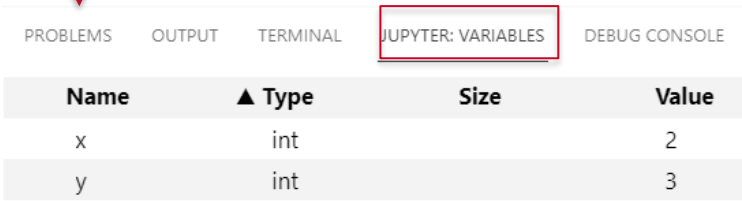
Interrupt the execution of the cells (similar to the Stop button in Rstudio)

Defined variables

Export the notebook to PDF or HTML

Customize notebook

Python Version



3) Jupyter Notebooks

- Saving your Jupyter Notebook to a *.ipynb* file
  - Go to the menu item *File* -> *Save*
- Opening a Jupyter Notebook
  - Go to the menu item *File* -> *Open* and select the file
- Moving forward
  - I will share a Jupyter notebook file with you every single class
  - Available on Canvas at least 12h before class

INTRO TO MARKDOWN

Markdown

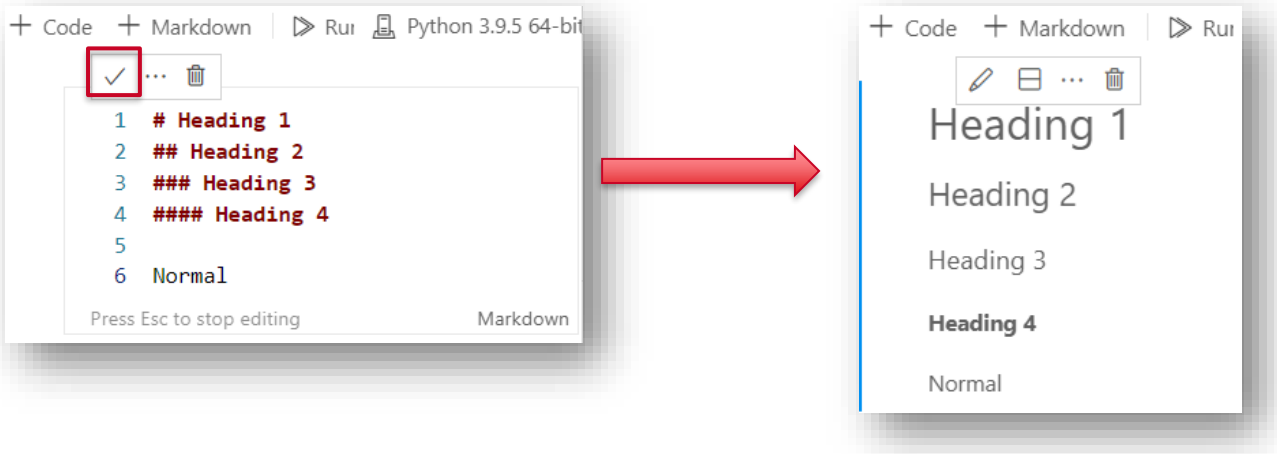
- Technically speaking, notebook cells can be of different types
  - We focus on two: Python and Markdown
- Markdown: markup language to format texts
  - We barely scratch the surface of what can be done with markdown
    - Enough to produce neat reports/deliverables
  - Quick reference: https://www.markdownguide.org/cheat-sheet/

Markdown

- Create a markdown cell
  - Add some code to your cell (look at the '#' symbol)
    - Run the code (ctrl + enter or check button)

```
# Heading 1
## Heading 2
### Heading 3
#### Heading 4

Normal text
```



Markdown

- Let's try to mix together Python and Markdown
  - Create a markdown, followed by a Python, and finally another Markdown cell
  - Add the following code to and run all the cells:

Cell #1
```
# This is my *first* **notebook**
```

Cell #2
```
x = 1
```

Cell #3
```
The code `x=1` assigns the value 1 to `x`
And here is an old image of ISA Faculty:
Link: [ISA Professors](https://www.miamioh.edu/fsb/_files/images/isa/isameet-800x400.jpg)
![ISA Professors](https://www.miamioh.edu/fsb/_files/images/isa/isameet-800x400.jpg)
```

Markdown

- Why use Markdown?
  - *De facto* language to report data analysis and for technical documentation
    - For example, it is used on GitHub
  - Markdown is portable
    - Unlike Microsoft Word or similar software that lock content into a proprietary file format
  - Markdown is platform independent
  - You will see the power of Python + Markdown inside Jupyter notebooks as we progress

Summary

- We have learned the basics of VS Code and Markdown
  - More about Markdown: https://www.markdownguide.org/
- Next lecture: Preliminaries (part II)
  - How to use built-in functions
  - How to declare variables
  - How to control the flow of a Python code