SOFTWARE: COMPUTER PROGRAMMING BASICS I

OT 699 - Week 1

Sook-Lei Liew, PhD, OTR/L, sliew@usc.edu

OBJECTIVES

- By the end of this session, students will be able to:
  - Understand why computer programming is useful
  - Describe common principles for computer programming
  - Install Anaconda
  - Open Jupyter Notebook
  - Write their first lines of code (in Python)

SOFTWARE VS HARDWARE

- Software = programs, code, commands to the computer, files
- Hardware = physical components of the computer (or associated machine parts, like sensors, arduinos, robots, etc.)
- We will learn to use them together, but first we will start with software so we understand how to talk to computers through programming

WHY COMPUTER PROGRAMMING?

- If you want to work with technology, you need to learn to tell a computer or machine what you want it to do in terms it understands
- Computers have their own language(s) and syntax(es); learning to program is essentially like learning a foreign language
- Programming will allow you to create web pages, apps, robots, devices, etc.

BINARY

- “Human-readable” inputs, like letters, numbers, and symbols from our keyboards, have to be translated for the computer
- Machine language essentially has two states (binary)
  - 0 (off/down switch) and 1 (on/up switch)
  - https://www.youtube.com/watch?v=Xpk67YzOn5w

  - Try it yourself: Convert text to binary:
  - http://www.unit-conversion.info/texttools/convert-text-to-binary/

BINARY EXAMPLES

A – 01000001

a – 01100001

Hi! - 01001000 01101001 00100001

[space] – 00100000

Hi A! - 01001000 01101001 00100000 01000001 00100001

PROGRAMMING LANGUAGES

- There are many different types of programming languages and environments, depending on what you want to do.
- Examples (not exhaustive!):
  - Websites → HTML, CSS, javascript
  - Data science → Python, R (which is really a statistical language)
  - Data analysis, signal processing → Bash/Linux, Matlab
  - Software → C++, Visual Basic, Java
- Each of these converts letters into binary for the computer

PICKING A LANGUAGE

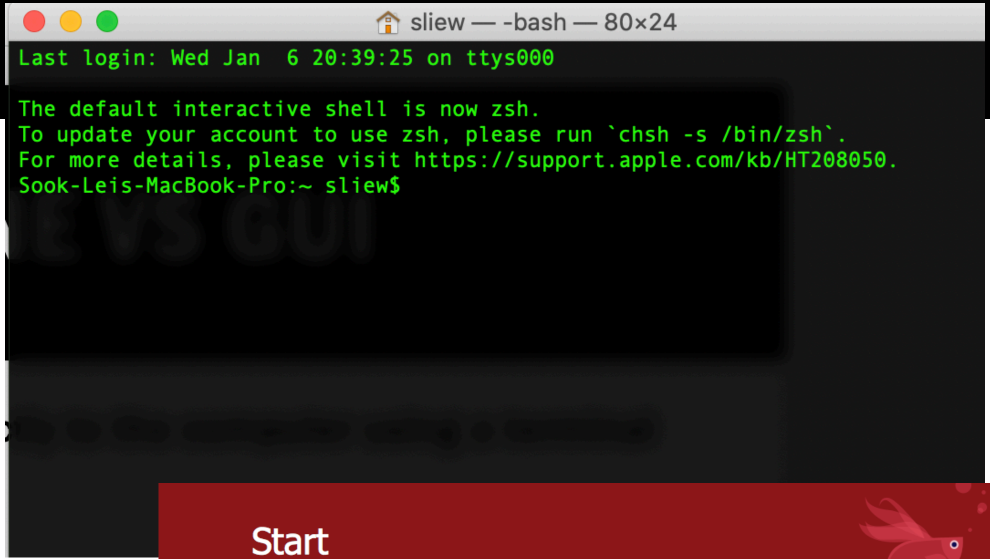
- Identify what you want to do, then find the language that is best suited for that
- Because each language has a limited number of commands and libraries (e.g., lots of commands that do specific tasks), each language will have different functionalities – such as commands to interface with the internet, to read and write text files, etc.
- In this course we will learn some basics in Python because it is a relatively intuitive language that is used for many purposes

COMMON PRINCIPLES

- Each language has its own syntax and commands, but most concepts are the same
  - “I’m hungry” → different in different languages, but the intent is the same
- Every digit matters! (e.g., spaces, capitalization, etc. are important because remember, each of these is encoded by a different sequence of 0s and 1s!)
- Learning basics of one language will allow you to learn other languages more easily

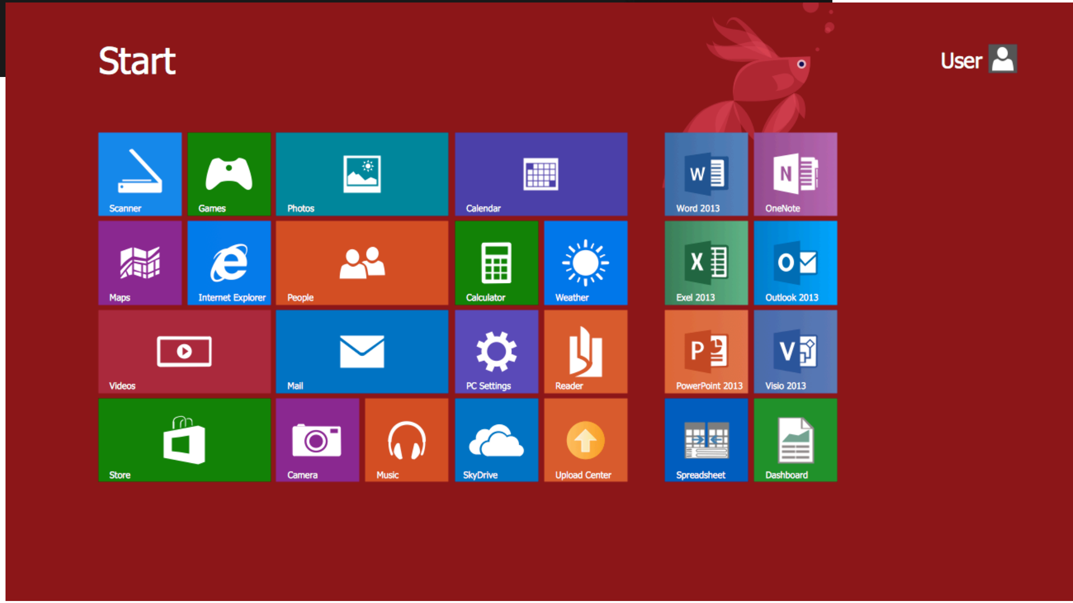
COMMAND LINE VS GUI

- Command line interface (CLI)
  - CMD prompt on Windows
  - Terminal on Mac
  - Tends to be more powerful/flexible and lightweight
- However, we most commonly use graphical user interfaces (GUIs) (e.g., visual representations of commands)



```
Last login: Wed Jan  6 20:39:25 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Sook-Leis-MacBook-Pro:~ sliew$
```



GUIS → IDES

- For many programming languages, people have developed GUI-based systems to make programming easier. These are called interactive development environments (IDEs).
- An IDE typically has an editor where we type the code, along with tools for debugging the code, visualizing the variables in a list, and seeing the output.
- Popular Python IDEs include IDLE, Spyder, etc.
- More about IDEs: https://www.codecademy.com/articles/what-is-an-ide
- Here is an online Python compiler:
  - https://www.programiz.com/python-programming/online-compiler/

ANACONDA / PYTHON / JUPYTER

- In this class, we will use a little bit of both – we will download a package called Anaconda which provides both GUI and CLI options, for both Windows and Mac (and Linux)
- Within Anaconda, we will use a program called Jupyter Notebook, which allows us to type Python (language) commands and see the output/result of each command right away
- This is great for initial learning (and debugging)!
- In this class we will use Python 3.

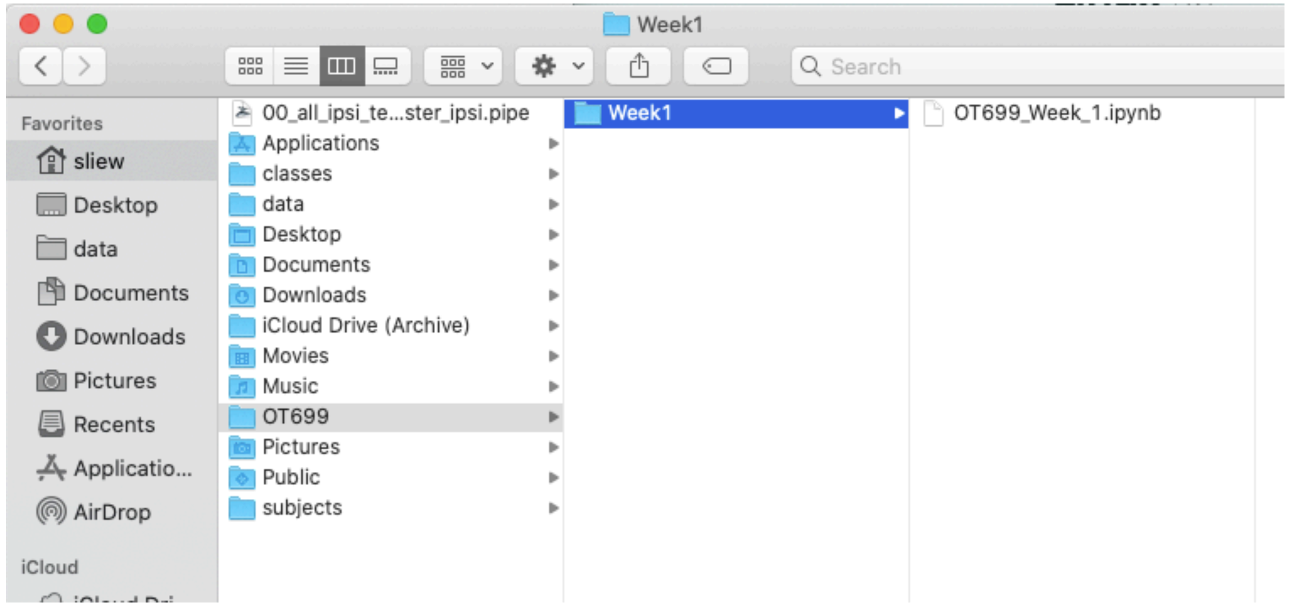
DEBUGGING

- The only feedback you will get from the computer is the execution of your command, or an error
- Debugging is where we fix errors (or unexpected results)
- This is a huge part of programming – writing code, and then debugging it! Ideally, you will debug it as you go so you can identify mistakes early on (versus after you’ve written 1000 lines of code)
- It is basically problem solving. We will do it together in class if you have errors!

LET'S GET STARTED!

Let's talk about File Structure!

1. In your main folder (C:/ on Windows, or your username on Mac):
2. Create a folder called "OT699" and within that, a folder called Week1
3. Download our OT699_Week1.ipynb file from blackboard and put it in username/OT699/Week1/



LET’S GET STARTED!

Next, let’s install the software:

1. Download Anaconda for your computer:

   https://www.anaconda.com/products/individual

   Choose your operating system + the graphical installer

2. Install Anaconda for your computer
3. Open the Anaconda GUI (search: Anaconda Navigator)
4. Click on Jupyter Notebook

LET’S GET STARTED!

5. Open the OT699_Week1.ipynb file in Jupyter Notebook

Congrats, you completed the first part of software download and installation! We will continue the tutorial from Jupyter Notebook! :)