# COMPUTER SCIENCE

## LESSON 11+12, TUESDAY OCT 17TH +THURSDAY OCT 19TH

WWW.CITA.UTORONTO.CA/~WOODFORD

WOODFORD@CITA.UTORONTO.CA

# EXTERNAL LEARNING OPPORTUNITY: THURSDAY OCT 19TH – URGENT!

- The Coming Revolution in Computational Astrophysics
  - Saul Teukolsky, Cornell University
  - 4:10 at 60 St George St – **need to leave Abelard by 3:45pm**
- Large-scale computer simulations are increasingly crucial in explaining astrophysical phenomena. Just one recent example is the stunning agreement of the full 3-d solution of Einstein's equations for colliding black holes with the observed signal from LIGO. For the past 50 years, the dominant computer method for solving these kinds of equations has remained essentially unchanged. To keep up with continuing advances in observation, simulations will require more fidelity and higher accuracy. One might think that with exascale machines becoming available in the next 5 years, this will be easy. This is not true: I will explain why Moore's Law is broken, and how the next generation of supercomputers will instead get their power by having millions of processors. Current codes will not be able to use these machines efficiently. I will describe new methods for harnessing the power of such exascale computers to solve some of the largest problems in astrophysics and other areas of science.

# THIS WEEK IN CS AND STEM

- LIGO announces BNS detected with EM and GW astronomy
  - Confims that all elements heavier than iron are formed by BNS mergers
  - https://www.ligo.caltech.edu/page/press-release-gw170817

- Programmable synthetic skin developed by engineers at Cornell U
  - https://futurism.com/scientists-have-created-programmable-synthetic-skin/

# DEFINING FUNCTIONS

- We've learned a lot of very valuable tools, however we've just been using python as a calculator thus far.
- Let's define functions that can be called again. Remember the steps to coding:
  1. **Identify the problem:** what do you need to do? What is being asked?
  2. **Find a solution:** research, read, formulate, write pseudo code either literally or in your mind.
  3. **Implement your solution, verify the result:** write a rudimentary code that gets the job done – even if it's messy at this stage. Try to double check your answer.
  4. **Improve efficiency:** go back and incrementally improve the efficiency of your code. This may be condensing variables, cutting lines, creating functions for repeated actions.
  5. **Improve aesthetics:** ensure legibility of your code, use variable names that make sense, make your code well commented

# DEF FORMAT

```
def name(argument):
        (some statements)
        return (item)
```

# GETTING INTO SCRIPTS

- Open Spyder!

# BASIC ARGUMENT PASSING AND USAGE

- Def vs a script
  - Still need defs in a script, but with a script you can make a file that contains many defs and import it into a new script as a library. This is called modularizing your code and is very useful for keeping your code clean and efficient.
  - Ex: import filename as fn ; from filename import func
- Using the command line

# THE MAIN FUNCTION

- In a script, you typically have a main function that houses what the script is supposed to iterate. You can modularize functions to be used within main, but main is the function that runs when you call the script.
- This is different than making a .py file to house a bunch of miscellaneous functions, which is what we would call a library.
- The format for a script would be:

```
import …                          #Call all your packages
def func(arg):                    #Define any functions you need to use in main()
        …
        return …
def main():                       #This defines the main function. Note there are no arguments.
        …
        …func(arg)…               #Call functions throughout
if __name__="__main__":           #For the compiler to know what to run, it needs to be able to see main.
        main()                    #This block tells the compiler to run the main function when the script is
                                  #called.
```

# ASSIGNMENT #4 = PROJECT 1

- 2 parter, code and short report (no refs necessary)

- Follow the instructions in Assignment4.ipynb and Assignment4.py. You'll need to submit both files for full credit plus a report document for full credit.
- Due Oct 29th by 11:59pm via email submission to woodford@cita.utoronto.ca