COMPUTER SCIENCE

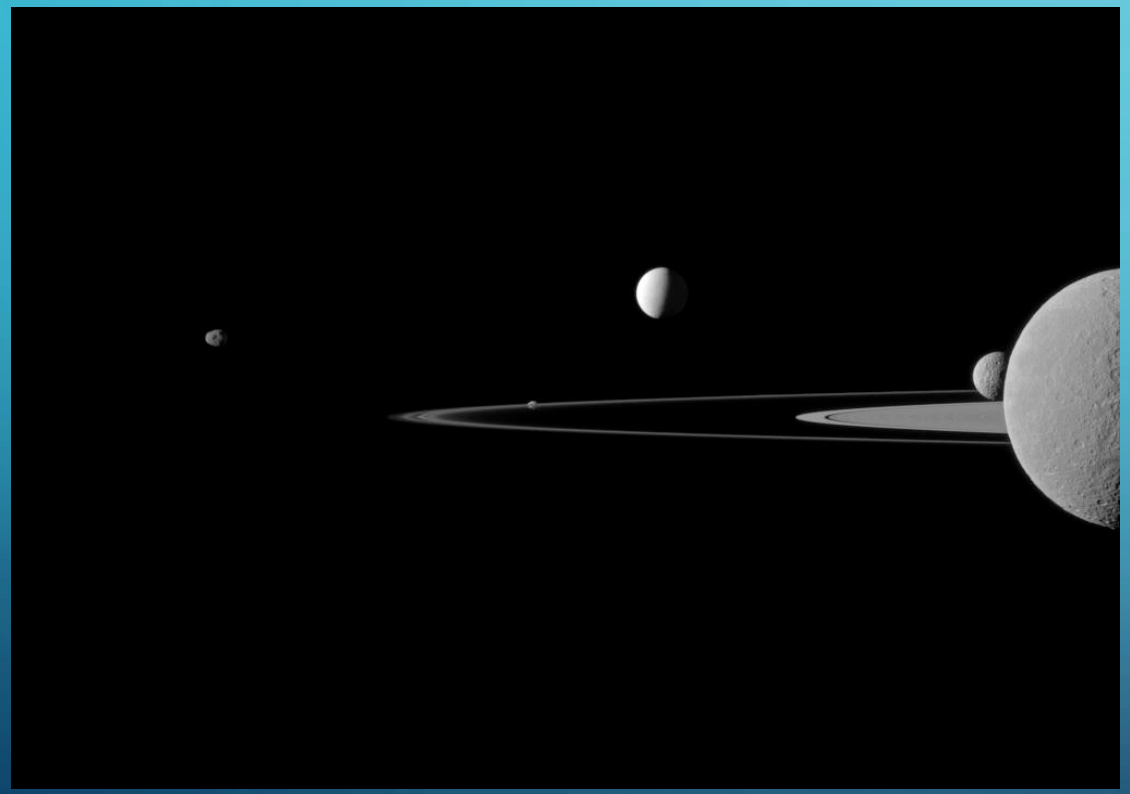
LESSON 4+5: TUESDAY SEPTEMBER 19TH AND THURSDAY SEPTEMBER 21ST , 2017

WWW.CITA.UTORONTO.CA/~WOODFORD

WOODFORD@CITA.UTORONTO.CA

THIS PAST WEEK IN CS AND STEM

- Cassini space probe ended it's 20 year mission, by disintegrating into the planet it was sent to orbit



Images from nasa.gov/ebooks

THE TERMINAL

- In order to code, you have to know how to communicate with your computer in a basic, simple way
- Open the terminal window (terminal for Mac, MobaXterm for Windows)

SOME BASIC COMMANDS

- ls – list all items in a directory

pwd – tell you your current working directory

- cd – change directory
- mkdir – make a new directory
- Try it out – verify with your GUI!
- Now try removing it - rm

- flags
  - Sometimes you want to maybe view hidden folders, or remove everything in a directory, or only remove files of a certain type. Here are a few common flags for the ls and rm commands:
  - ls:
    - -a = list ALL files/folders, including hidden folders
    - --help = help (most commands have this – really useful!! Can also be –h, --h, -help)
    - -X = sort by extension
    - * = replaces everything – Let's try an example!
  - rm:
    - -rf = remove everything, even recursively down through directories (BE CAREFUL, you cannot bring back files or directories removed this way!)

EDITING IN THE TERMINAL

- There are a number of editors you can use, and it's all up to preference. For simplicity, let's try VIM
  - Type {vim filename } into the command line to open a file
  - i to insert
  - ESC to stop inserting items
  - :q to quit
  - :w to save (:wq to save and quit)
- Cat – print your file to screen
- Tail – print the bottom lines of your file to screen

- Give it a test run:
  - Make a new directory
  - Inside that new directory, create a text file with one line of text and save
  - Find that file and open it with your default GUI text editor

  - Any issues so far?

PIPING

- You can “pipe” the output of a command in bash as the input for another command.

- Try ls –l |less
  - What do you think will happen?
  - What does the –l flag do?

FINDING FILES FROM COMMAND LINE

- find – very powerful command, can help locate files and directories
  - Flags:
    - name (look for files with a particular name or part of a name)
    - type (look for file type)
  - Try it out! What happens for “find –name *.txt”?

CANCELLING COMMANDS

- Say you've started a search command or a program and realize you don't want it afterall – how do you stop it?
  - CTRL-C
  - exit
  - stop

MISCELLANEOUS

- Tab completion
- Going up the directory tree (../../)
- “hot keys”
  - CTRL-a = go to beginning of line
  - CTRL-e = go to end of line

COPY AND MOVE

- These are also very common, you can copy files and whole directories to another location or move them, which could be considered rename or cut/paste.
  - cp {file to copy} {where to put new copy} (include –r for directories)
  - mv {file to move/rename} {new location and name}

FILE TYPES AND AMBIGUITY

- Working in the terminal gives you a lot of freedom and control – and of course responsibility comes with that.
  - Your programs and apps are no longer naming and sorting files for you – YOU have to make sure you organize and name files appropriately.
  - Extensions are NOT important here. You can create a text file with a ".jpg" or ".pdf" and it will still be a text file – try it out!
    - What do your GUI apps do?

BEGINNINGS OF BASH

- Bash is the language you're using in the terminal – all of the commands we've tried thus far count as bash!

- Of course, things can get much more complicated – let's try putting things inside a program that the terminal can then read and use.

- Open a .txt file in a directory of your choosing
- Edit it to contain: “#!/bin/bash
  echo Hello World
- What do you think this will do? What is echo?

- Open a .txt file in a directory of your choosing
- Edit it to contain: “#!/bin/bash
  echo Hello World
- What do you think this will do? What is echo?
- Now save and exit. Run with “bash [].txt”. What happens?

OUTPUT TO A FILE

- Everything you can do in a bash file you can also do in the command line.
- One very useful command is “>”:
  - Use this to output to a file, where you’d do: “command_or_program > file.txt” to capture all of the output of whatever command or program you decide to run.

WHAT IS A BYTE?

- We've heard of GB, TB, KB, etc but what is B = byte?

WHAT IS A BYTE?

- We've heard of GB, TB, KB, etc but what is B = byte?
  - 8 bits
  - What's a bit?

WHAT IS A BYTE?

- We've heard of GB, TB, KB, etc but what is B = byte?
  - 8 bits
  - What's a bit?
  - A single digit in a number, so in the binary number 1011 there are 4 bits and half of a byte. For a full byte, it would be represented as 00001011.

BASICS OF TELLING LANGUAGES APART

- What are some main differences in languages? How can we tell them apart?

BASICS OF TELLING LANGUAGES APART

- What are some main differences in languages? How can we tell them apart?
  - Accent
  - Alphabet
  - Diction/vocabulary
  - Phonetics
- But how do you tell computer/coding language apart?

SYNTAX

- Fortran: Tab dependant, “!” indicate comments (ie. not read by the computer)
- C/C++: Statements end with “;”, not tab dependent, “//” indicate comments
- Python: Tab semi-dependant (doesn’t change the code but creates warnings), “#” indicates comments.

FILE TYPES

- Fortran: *.f90
- C/C++: *.cc or *.cpp
- Python: *.py

HOW TO RUN A CODE FILE

- Depends on the language, of course. For C/C++ and Fortran, you cannot run them as they are. They first need to be compiled and translated into computer speak (eg binary) beforehand. The new computer speak file that contains the tasks for your code is called an “executable” (usually has the extension .out)

HOW TO RUN A CODE FILE

- Depends on the language, of course. For C/C++ and Fortran, you cannot run them as they are. They first need to be compiled and translated into computer speak (eg binary) beforehand. The new computer speak file that contains the tasks for your code is called an “executable” (usually has the extension .out)

- Python is simpler: simply run “python *.py” in the command line and it will compile and run for you – no extra files, no extra space, and no extra worry. Any errors or warnings you have will appear on the command line.

BUT WAIT, THERE'S MORE!

- Python already sounds a lot easier than the other languages (it is), but it gets better.
  - You don't even need the command line to run simple python programs – you can run line by line in ipython, notebooks, or multi-function editors like Spyder or Enthought Canopy.
  - Python is sometimes straight-up used as a calculator in the command line. Let's try it out!

- Type “ipython” into the command line. If “no command found”, try “python”.
- You should have “>>>” at the bottom of the terminal
- Try some basic math operations!

- Let's try out "Hello World!" first
- What might you need to print to the command line?
- Note that # indicate comments!

PRINT

- To print to screen in python, you need to use the “print” command.

- It can take multiple inputs, separated by commas. Passing a variable leads to printing whatever is in memory for that variable. Doing something like “print “Hello!” ” will print the string “Hello!”.

VARIABLE TYPES

- What kind of variables are there? What can we use and pass in python?

VARIABLE TYPES

- What kind of variables are there? What can we use and pass in python?
  - Integers (ex 1,2,3)
  - Floats (ex 1.000, 2.15, 3e-10)
  - Strings (ex “Hello”)
  - Booleans (True/False, present or not present)

ASSIGNMENT #2

- Code
- Fix/Fill in the code snippets in “Assignment2.txt”, found at www.cita.utoronto.ca/~woodford
  - Note that you can try out commands now on your laptops to verify!
- Submit your updated version by email to woodford@cita.utoronto.ca by Sept 24 11:59 pm. Feel free to leave in .txt format, but please change the name of the file to include your name!

REFERENCES

- https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/3/html/Step_by_Step_Guide/s1-managing-shells.html
- http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html#toc2