

# Computer Science for the Physical Sciences

*Week 3*

---

*Craig Rasmussen (Research Support Services, University of Oregon)*

# Computer Science Minor: *Review*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures
- Upper-division courses (8 credits)
  - [Computer Architecture](#)
  - Introduction to Algorithms
  - C/C++ and Unix
  - [Operating Systems](#)
  - Automata Theory
  - Software Methodology I-II
  - [Introduction to Compilers](#)
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning

# Computer Science Minor: *This week*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures Lists and Maps
- Upper-division courses (8 credits)
  - Computer Architecture
  - Introduction to Algorithms Complexity
  - C/C++ and Unix Python and Shell
  - Operating Systems
  - Automata Theory
  - Software Methodology I-II  
Revision Control and Make Files
  - Introduction to Compilers
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning

# Computer Science Minor: *Shell commands*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures [Lists and Maps](#)
- Upper-division courses (8 credits)
  - Computer Architecture
  - Introduction to Algorithms [Complexity](#)
  - C/C++ and Unix [Python and Shell](#)
  - Operating Systems
  - Automata Theory
  - Software Methodology I-II  
[Revision Control and Make Files](#)
  - Introduction to Compilers
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning

# Unix Shell Commands: *Taxonomy*

---

- Help
- Directories and navigation
- Files
- Permissions and resources
- Network
- Users and groups
- Processes
- Terminal
- Data discovery

# Shell Command Taxonomy: *Help*

---

- `man` - format and display the on-line manual pages
- `whatis` - search the `whatis` database for complete words
- `apropos` - search the `whatis` database for strings
- `which` - locate a program file in the user's path

# Shell Command Taxonomy: *Directories and Navigation*

---

- mkdir - make directories
- rmdir - remove directories
- cd - change current directory
- ls - list directory contents

# Shell Command Taxonomy: *Files /*

---

- ls - list directory contents
- touch - change file access and modification times
- rm, unlink - remove directory entries
- file - determine file type
- cp - copy files
- mv - move files
- cat - concatenate and print files
- more - display a file
- less - better version of more



# Shell Command Taxonomy: *Files II*

---

- head - display first lines of a file
- tail - display the last part of a file
- cut - cut out selected portions of each line of a file
- make - utility to maintain groups of programs
- gzip - compression tool using Lempel-Ziv coding
- gunzip - decompression tool using Lempel-Ziv coding
- zip - package and compress (archive) files
- unzip - list, test and extract compressed files in a ZIP archive

# Shell Command Taxonomy: *Permissions and resources*

---

- chmod - change file modes or Access Control Lists
  - look at file permissions (mode) with “`ls -l`”
- quota - display disk usage and limits
- df - display free disk space
- du - display disk usage statistics

# Shell Command Taxonomy: *Network*

---

- ssh - remote login program
- scp - secure copy (remote file copy program)
- ftp - internet file transfer program
- wget - non-interactive network downloader

# Shell Command Taxonomy: *Users and groups*

---

- finger - user information lookup program
- chgrp - change group
- groups - show group memberships

# Shell Command Taxonomy: *Processes*

---

- top - display and update sorted information about processes
- ps - process status
- kill - terminate or signal a process

# Shell Command Taxonomy: *Terminal*

---

- clear - clear the terminal screen

# Shell Command Taxonomy: *Data discover*

---

- echo - write arguments to the standard output
- wc - word, line, character, and byte count
- grep - file pattern searcher
- sort - sort lines of text files
- awk - pattern-directed scanning and processing language
- cut - cut out selected portions of each line of a file
- find - walk a file hierarchy

# Computer Science Minor: *Make files*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures **Lists and Maps**
- Upper-division courses (8 credits)
  - Computer Architecture
  - Introduction to Algorithms **Complexity**
  - C/C++ and Unix **Python and Shell**
  - Operating Systems
  - Automata Theory
  - Software Methodology I-II  
**Revision Control and Make Files**
  - Introduction to Compilers
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning



A *makefile* maintains groups of programs based on dependencies being satisfied

---

```
#
# this is a comment

#
# define environment variables (compilers/linker/libraries...)

CC = gcc

#
# define targets
all: hello

hello.o: hello.c
    $(CC) -c hello.c -o hello.o

hello: hello.o
    $(CC) -o hello hello.o

# run tests
check:

# clean up
clean:
    rm -f hello.o hello
```

target

dependency

tab

# Computer Science Minor: *Algorithmic complexity*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures [Lists and Maps](#)
- Upper-division courses (8 credits)
  - Computer Architecture
  - Introduction to Algorithms [Complexity](#)
  - C/C++ and Unix [Python and Shell](#)
  - Operating Systems
  - Automata Theory
  - Software Methodology I-II  
[Revision Control and Make Files](#)
  - Introduction to Compilers
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning

# Computational Complexity Theory

---

- The complexity of an algorithm is how the runtime scales as the number of elements  $N$  in a collection (an array for example) increases
- $T(N) = a_0 + a_1 \times N^1 + a_2 \times N^2 + \dots$ 
  - the complexity is the superscript of the leading term
  - call big O notation
- Array access is constant,  $O(N^0)$
- The inner product to two vectors is  $O(N^1)$
- Building a correlation matrix is  $O(N^2)$
- The order of an algorithm using an array data structure is *the number of loops passing over the entire array*

# Computer Science Minor: *IPython Notebook*

---

- Required courses (24 credits)
  - Introduction to Computer Science I-II-III
  - Elements of Discrete Mathematics I-II
  - Introduction to Data Structures **Lists and Maps**
- Upper-division courses (8 credits)
  - Computer Architecture
  - Introduction to Algorithms **Complexity**
  - C/C++ and Unix **Python and Shell**
  - Operating Systems
  - Automata Theory
  - **Software Methodology I-II**  
**Revision Control and Make Files**
  - Introduction to Compilers
  - Computational Science
  - Bioinformatics
  - Data Mining
  - Introduction to Artificial Intelligence
  - Machine Learning

# IPython Notebook

---

- Download and install the Anaconda Python distribution from:
  - <http://continuum.io/downloads>
- The Python Notebook combines within a single document:
  - code execution
  - text
  - mathematics
  - plots
  - rich media

# Running the IPython Notebook

---

- `ipython notebook`
  - this will open a notebook and run it from your browser