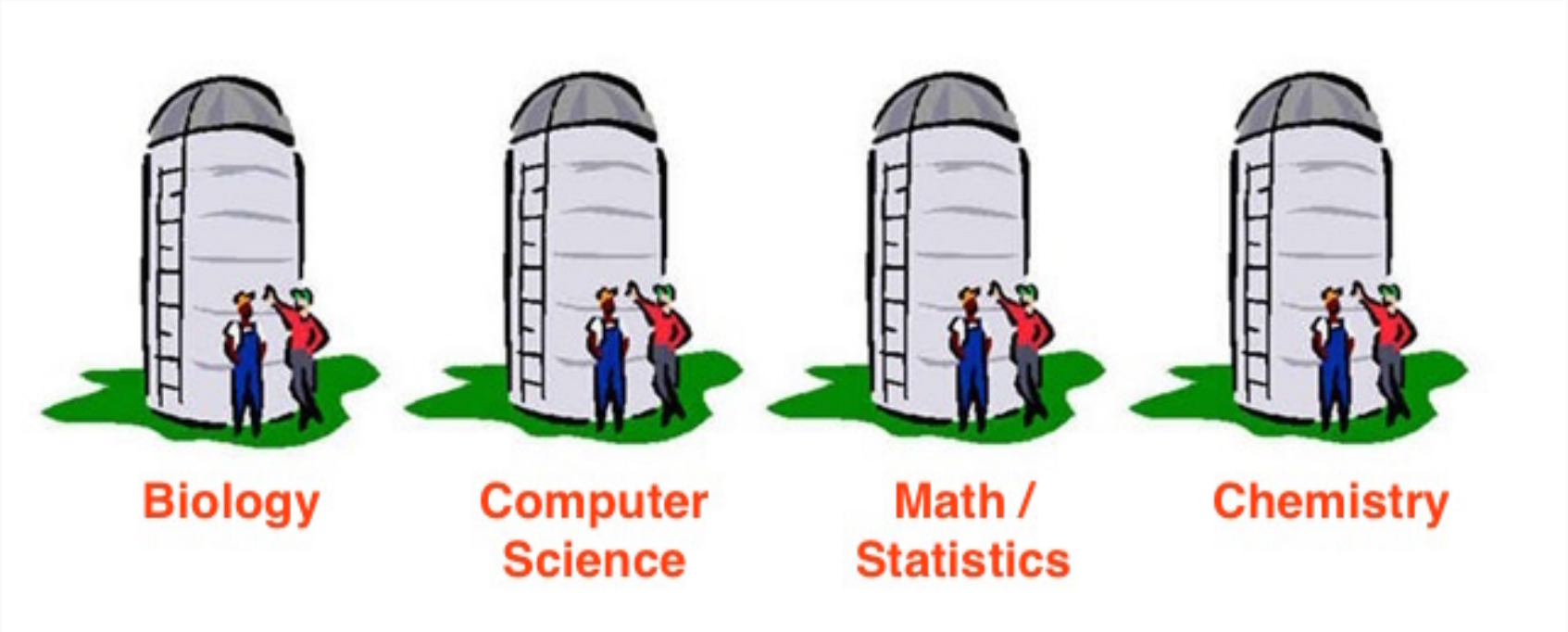
COMP 383 Computational Biology

Introductory Lecture

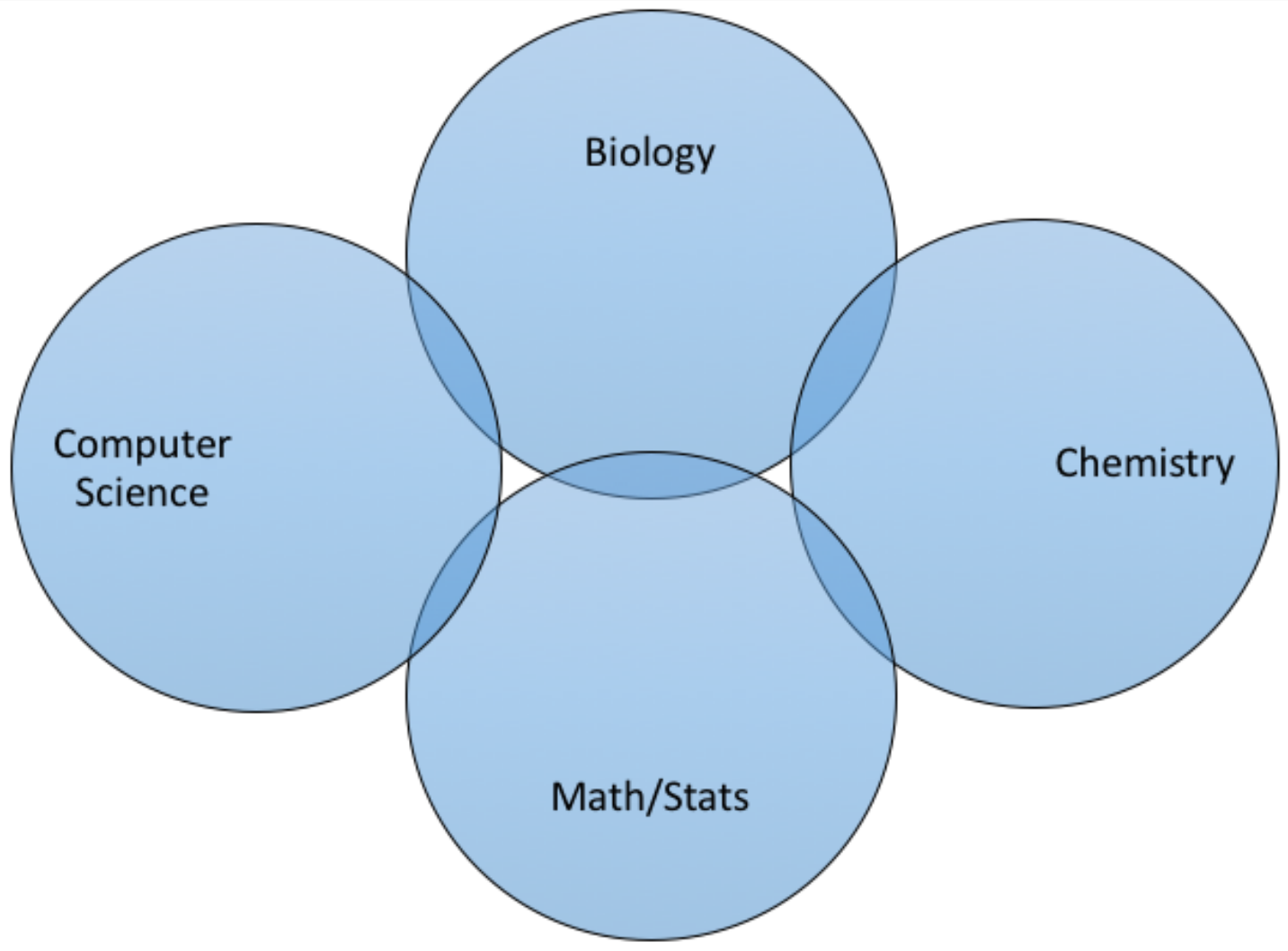
Spring 2017

Loyola University Chicago
hwheeler1@luc.edu

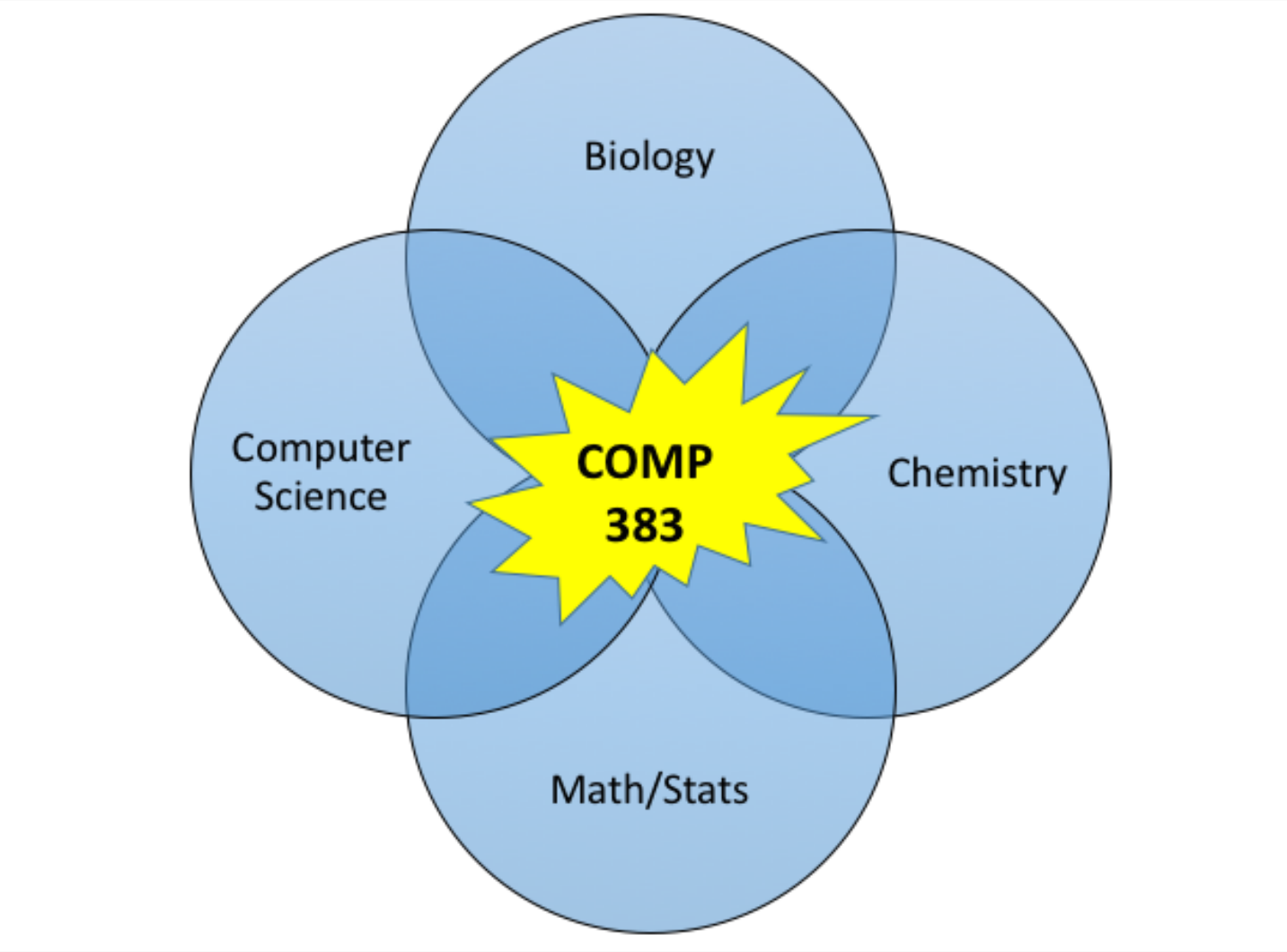
Bioinformatics Major Courses



In BIOL 388 Bioinformatics, you started to see an intersection of these fields



COMP 383 Computational Biology integrates all you've learned



Research must be reproducible

http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003285

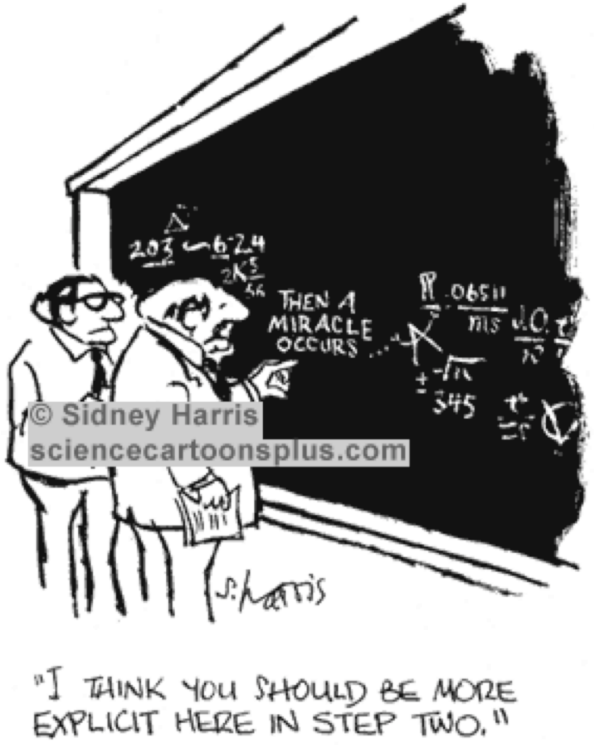
OPEN ACCESS Freely available online

PLOS | COMPUTATIONAL BIOLOGY

Editorial

Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve[1,2]*, Anton Nekrutenko[3], James Taylor[4], Eivind Hovig[1,5,6]



Rule 1: For Every Result, Keep Track of How It Was Produced

Rule 2: Avoid Manual Data Manipulation Steps

Rule 3: Archive the Exact Versions of All External Programs Used

Rule 4: Version Control All Custom Scripts

Rule 5: Record All Intermediate Results, When Possible in Standardized Formats

Rule 6: For Analyses That Include Randomness, Note Underlying Random Seeds

Rule 7: Always Store Raw Data behind Plots

Rule 8: Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected

Rule 9: Connect Textual Statements to Underlying Results

Rule 10: Provide Public Access to Scripts, Runs, and Results

COMP 383 Course Structure

1. First month, learn Python and the Unix command line, apply them to common biological problems (toy examples) - 28%
2. Group ***Research*** Project: You will be given freedom to work on a computational biology problem - 72%

- This could involve automating a research pipeline or building an analysis tool
- This is a “real life” research project, so data will be messy and unanticipated obstacles will occur
- It will be important to set benchmarks and stay on task in class
- Don’t procrastinate, get things done and strive to make your code better
- Learn all you can, the future of biology/medicine/data analysis needs you!

The syllabus and assigments are posted on the class website

http://hwheeler01.github.io/CompBio/

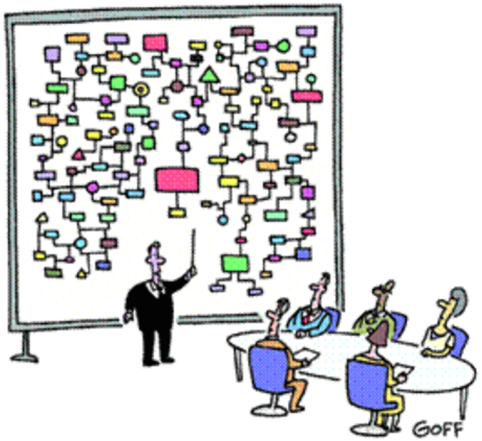
COMP 383 Computational Biology

Dr. Heather E. Wheeler, Spring 2017

Syllabus Assignments Resources

Course objectives

- Analyze primary literature in the field of bioinformatics and computational biology
- Design and implement advanced data structures for computational biology
- Design and implement advanced algorithms for computational biology
- Increase proficiency in object-oriented and scripting programming languages



"And that's why we need a computer."

Why Python?

1. Everybody’s doing it. It has a large (and growing) user base among scientists.

2. Python is a general purpose, high level, programming language – you should be able to do anything you want to do using Python, and it should be relatively easy to accomplish.

3. It’s free, well-documented, and runs on all operating systems.

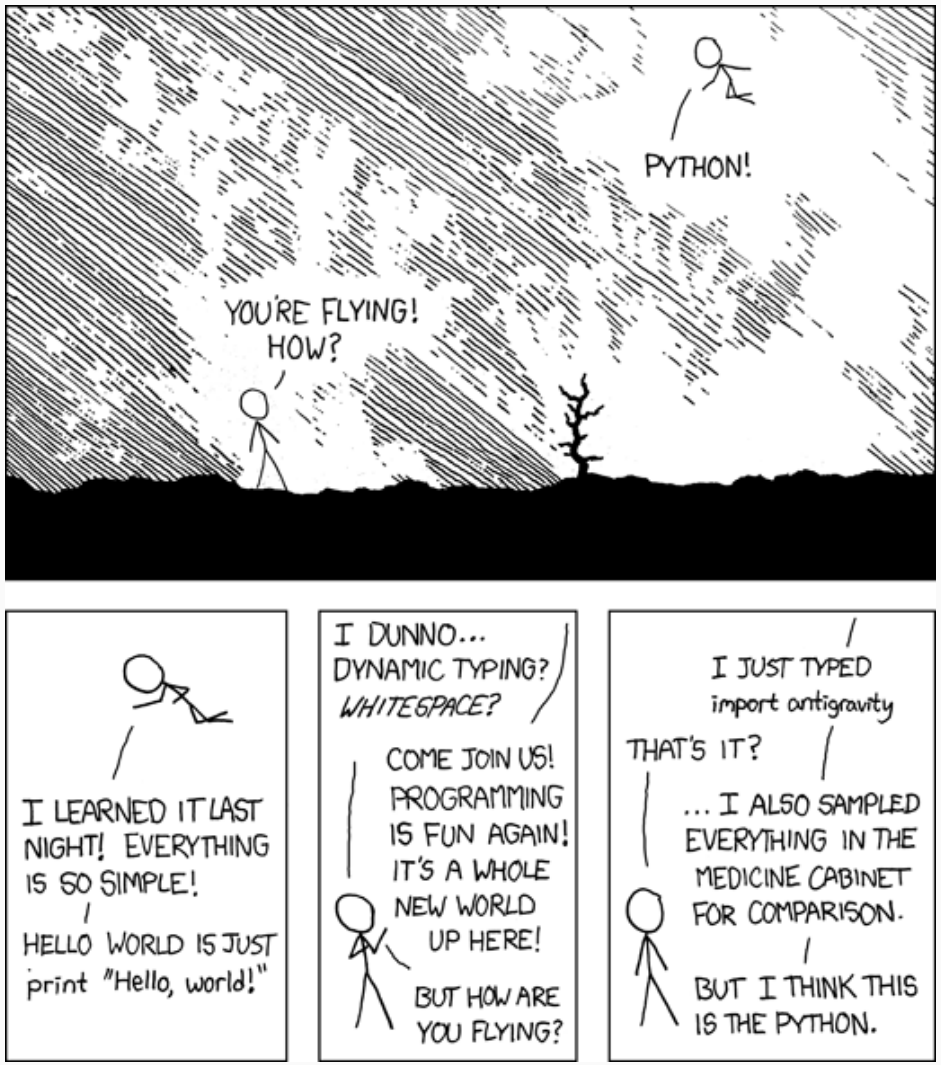
Why Python?

4. It’s easier for novices to pick up than most other languages and it’s also used by many professional programmers. This makes collaborating with both novices and experts easier.

5. A dynamic language like Python allows us to write small programs quickly and to also manage the complexity of larger ones.

6. If we want to squeeze every last ounce of performance out of our hardware, then a compiled language (e.g. C, C++) is better, but if we want to quickly answer a research question or build a pipeline around other software, Python is often easier/faster.

Why Python?


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Python 2.7

For this class, I recommend Python version 2.7 rather than Python 3.x

- ROSALIND assumes 2.7
- BioPython for Python 3.x is not as stable, but improving all the time
- If you’re used to Python 3, feel free to use it, just note some of the ROSALIND hints may require slightly different syntax

Python 2.7 Strings

```
print "Hello, World!"
a = "Hello"
b = "World"
type(a)
print a + ", " + b + "!"*3
a[0:4]

## Hello, World!
## <type 'str'>
## Hello, World!!!
## 'Hell'
```

Python 2.7 Numbers

```
a = 12
b = 2.5
c = a + b
type(a)
type(b)
print str(a) + " + " + str(b) + " = " + str(c)

## <type 'int'>
## <type 'float'>
## 12 + 2.5 = 14.5
```

Python 2.7 Division

```
18/5
18/5.0
18/float(5)
```

```
## 3
## 3.6
## 3.6
```

Or place this import at the beginning of the file:

```
from __future__ import division
18/5
```

```
## 3.6
```

Install Python and Rodeo

Anaconda distribution

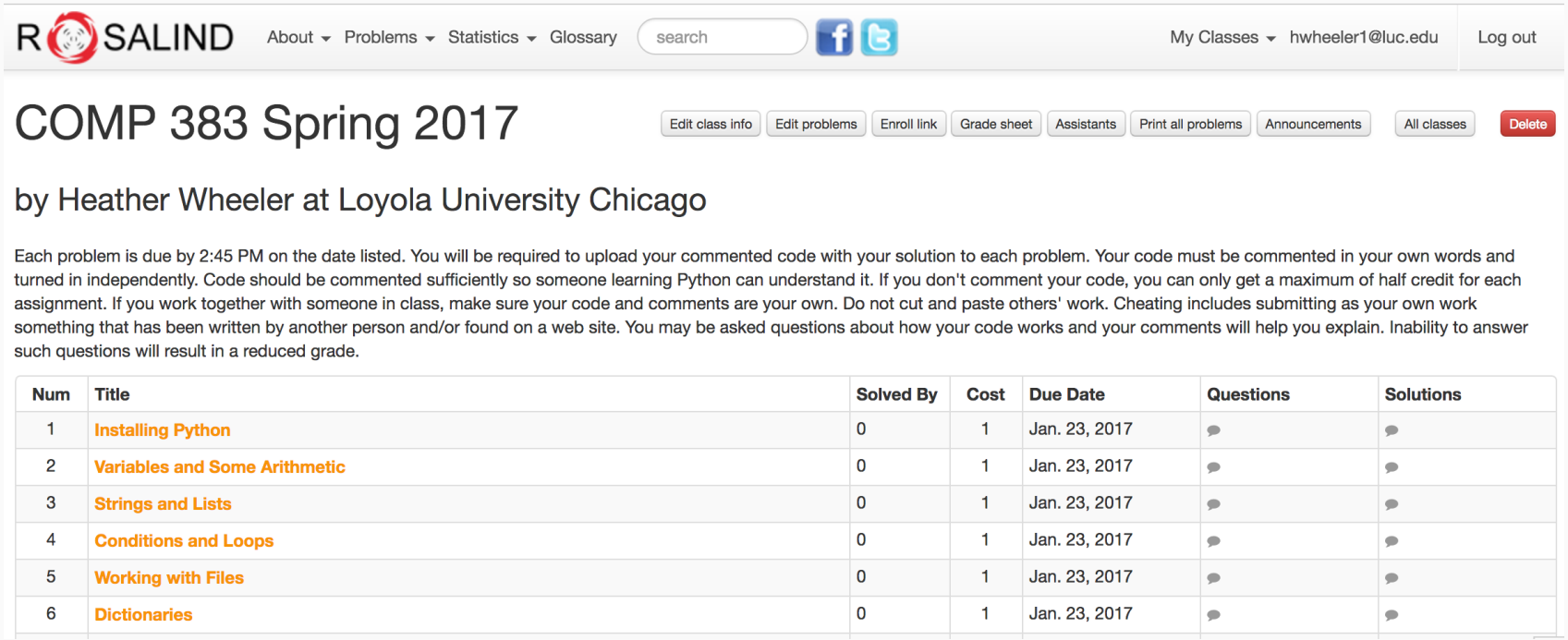
- Python with a bunch of packages useful for scientific computing
- https://www.continuum.io/downloads

Rodeo IDE

- similar to RStudio, but for Python
- https://www.yhat.com/products/rodeo

Start first assignment in ROSALIND

http://hwheeler01.github.io/CompBio/assignments/



Num	Title	Solved By	Cost	Due Date	Questions	Solutions
1	Installing Python	0	1	Jan. 23, 2017		
2	Variables and Some Arithmetic	0	1	Jan. 23, 2017		
3	Strings and Lists	0	1	Jan. 23, 2017		
4	Conditions and Loops	0	1	Jan. 23, 2017		
5	Working with Files	0	1	Jan. 23, 2017		
6	Dictionaries	0	1	Jan. 23, 2017		