# Next ?

MCDB 170

## Where are we now with Python?

- Variables
- Conditionals
- Loops
- Functions
- Data containers
- Classes
- Modules
- Packages

- Python: General programming language, NOT domain specific
  - Web development
  - Game programming
  - Desktop application
  - System programming
  - Hardware control (Robotics, DIY projects, Embedded system, etc)
  - Scientific computing
  - Etc…
  - Ton of library
  - Most scientific fields are moving to Python/R

- Matlab: Domain specific
  - Matrix computation → Easier for scientific computation, but lacks libraries

**Some differences between Python and Matlab**

| | Python | Matlab |
|---|---|---|
| Variable | References, except for the native types (int, float, complex, string) | Values |
| Loop | for<br>while, while-else | for-end<br>while-end |
| Conditional | if, if-else, if-elif-else | if-end, if-elseif-end, if-elseif-else-end |
| Package management | import | Everything is accessible all the time |
| Code block | indentation | ‘end’ and parantheses |
| Indexing | 3:7 does not include 7 | 3:7 includes 7 |
| Array | You must learn numpy. Manipulation is a bit cumbersome | Everything is matrix and operations are intuitive |
| Broadcasting | There is certain rules | Just works |
| Code blocking | Indentation | Needs “end” or parentheses |
| Other syntaxes | Sometimes quirky for scientific computation | Relatively simple, but not very powerful |

| | Python | Matlab |
|---|---|---|
| OOP | Fully supported | Add on (clunky) |
| Libraries | Maybe the 2nd best for any scientific field | Lacking a lot |
| Difficulty | Very easy to learn | Easier for numerical computing. Not worth the effort as a general programming language |
| Fast? | OK | A bit faster for numerical analysis |
| Cool new features (e.g., AI) | Very good | OK |
| Should I learn? | Yes for everyone | Depends |

# Courses at MCDB

**MCDB 108c**

Dr. Matthieu Louis

Computational & Systems Biology

→ with Matlab

**Description**

Models of Biochemical and Cellular Systems. Introductory systems-biology approach to model the design and the function of biological systems. Students will develop an intuition about physical concepts that are fundamental to discuss how biological organisms acquire and process information from the environment. Those concepts and tools will cover probabilities and basic dynamical systems theory. Students will build models of processes of increasing complexity, ranging from viral dynamics, bacterial resistance to drugs, the maintenance of homeostatic equilibrium (trp operon), biological oscillators (mitotic clock) and genetic switches underlying cellular decisions (bacteriophage lambda and lac operon).

**MCDB 172**

Dr. Max Wilson

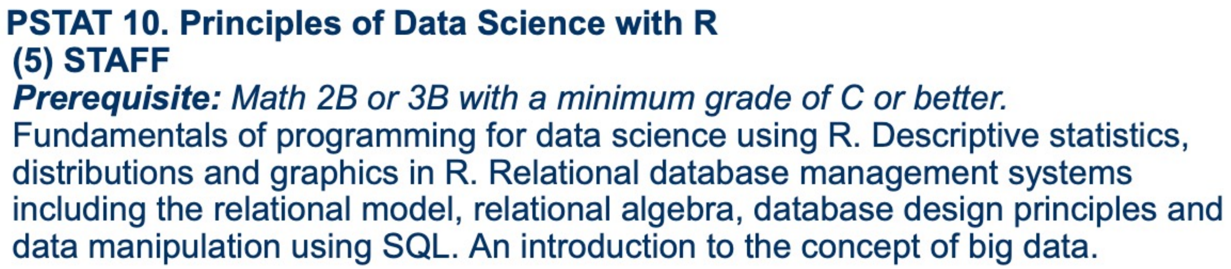
Computational & Systems Biology

→ with Python

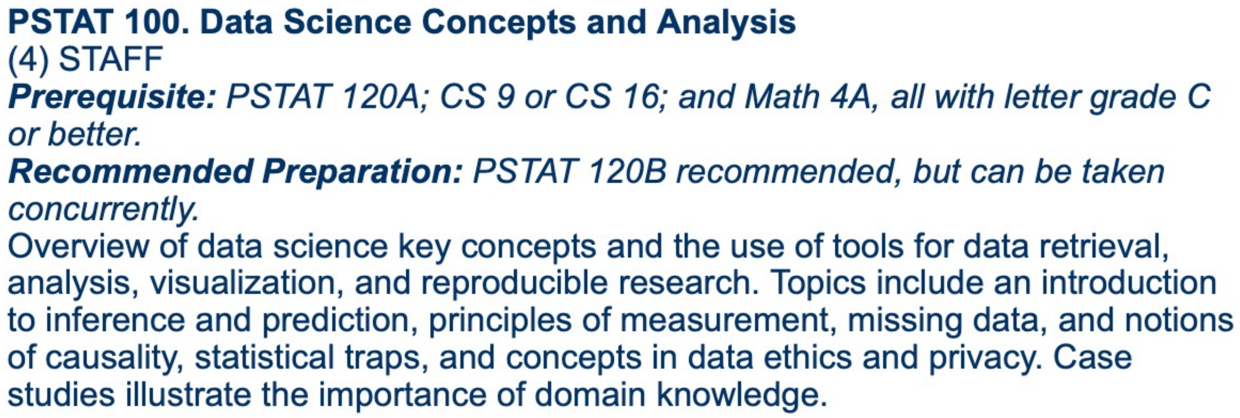
**Description**

An introduction to mathematical models and computer simulations used to describe and understand time varying biological systems. Learning Objectives: Survey mathematical methods for describing the dependence on time of biological phenomena. Illustrate how to construct mathematical models to gain insights into complex biological systems. Develop working knowledge of a python code base that enables future evaluation of common classes of models applied to the study of biological dynamics.

## Courses at Statistics → with R

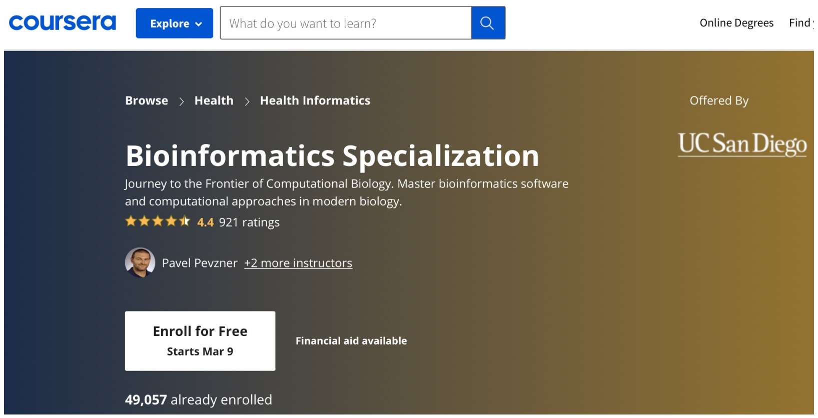


**PSTAT 10. Principles of Data Science with R**
**(5) STAFF**
***Prerequisite:*** *Math 2B or 3B with a minimum grade of C or better.*
Fundamentals of programming for data science using R. Descriptive statistics, distributions and graphics in R. Relational database management systems including the relational model, relational algebra, database design principles and data manipulation using SQL. An introduction to the concept of big data.



**PSTAT 100. Data Science Concepts and Analysis**
(4) STAFF
***Prerequisite:*** *PSTAT 120A; CS 9 or CS 16; and Math 4A, all with letter grade C or better.*
***Recommended Preparation:*** *PSTAT 120B recommended, but can be taken concurrently.*
Overview of data science key concepts and the use of tools for data retrieval, analysis, visualization, and reproducible research. Topics include an introduction to inference and prediction, principles of measurement, missing data, and notions of causality, statistical traps, and concepts in data ethics and privacy. Case studies illustrate the importance of domain knowledge.

# Online Courses



Browse › Health › Health Informatics

Offered By UC San Diego

## Bioinformatics Specialization

Journey to the Frontier of Computational Biology. Master bioinformatics software and computational approaches in modern biology.

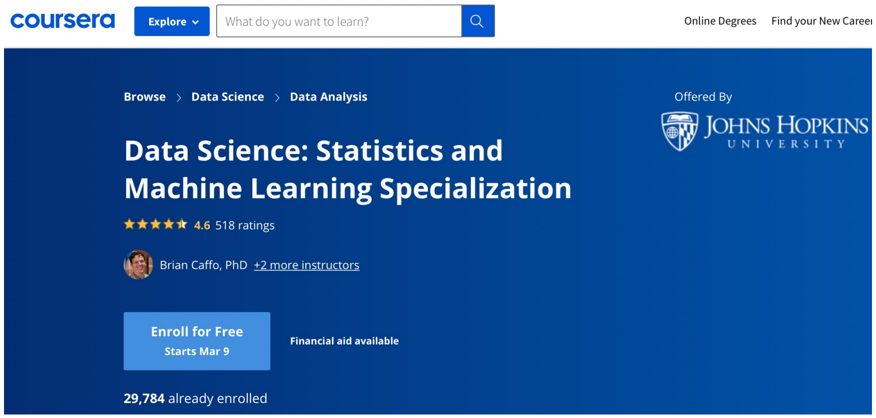
★★★★★ 4.4 921 ratings

Pavel Pevzner +2 more instructors

Financial aid available

49,057 already enrolled

No specific language



Browse › Data Science › Data Analysis

Offered By Johns Hopkins University

## Data Science: Statistics and Machine Learning Specialization

★★★★★ 4.6 518 ratings

Brian Caffo, PhD +2 more instructors

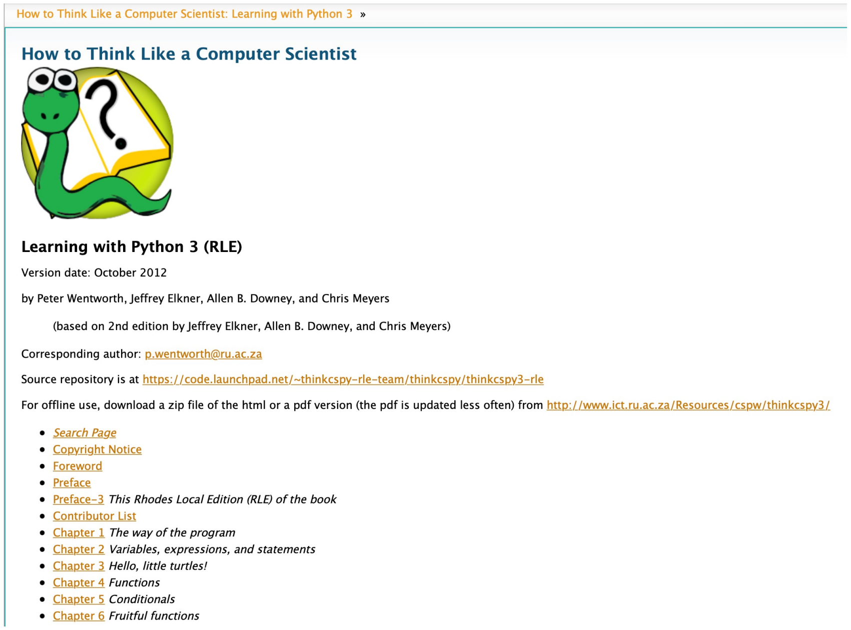
Financial aid available

29,784 already enrolled

with R

# About Python

https://openbookproject.net/thinkcs/python/english3e/index.html



How to Think Like a Computer Scientist: Learning with Python 3 »

## How to Think Like a Computer Scientist

### Learning with Python 3 (RLE)

Version date: October 2012

by Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers

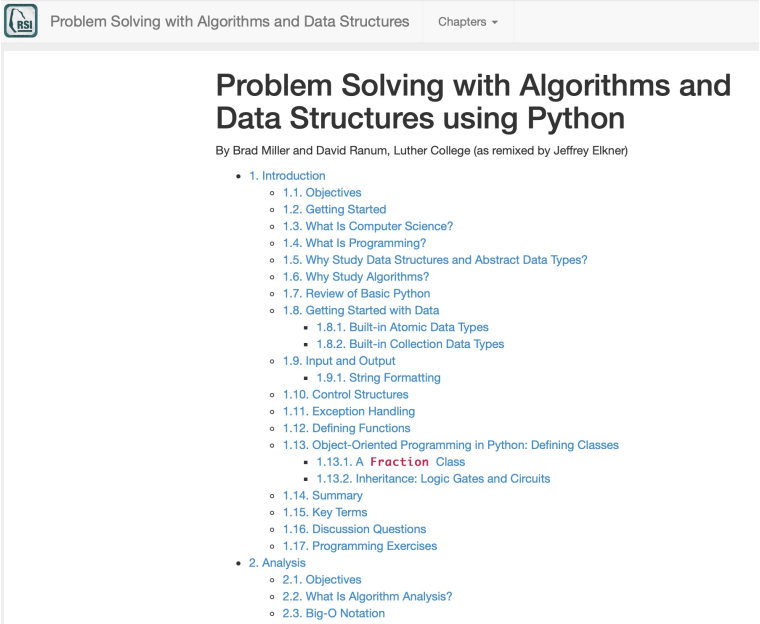
(based on 2nd edition by Jeffrey Elkner, Allen B. Downey, and Chris Meyers)

Corresponding author: p.wentworth@ru.ac.za

Source repository is at https://code.launchpad.net/~thinkcspy-rle-team/thinkcspy/thinkcspy3-rle

For offline use, download a zip file of the html or a pdf version (the pdf is updated less often) from http://www.ict.ru.ac.za/Resources/cspw/thinkcspy3/

http://www.openbookproject.net/books/pythonds/index.html



## Problem Solving with Algorithms and Data Structures using Python

By Brad Miller and David Ranum, Luther College (as remixed by Jeffrey Elkner)

# What should be your NEXT language?

- Web development → HTML5, Java script, Ruby, Java
- Game programming → C#, C++
- Desktop application → Java, C++, C#
- System programming → Go, C/C++
- Hardware control (Robotics, DIY projects, Embedded system, etc) → C/C++, Rust
- Scientific computing → **R**, **Python**, *Matlab*, Julia, C/C++