Introduction to Computer Science, CSC 280, Fall 2015

Instructor: Prof. Bei Xiao, American University.

Email: bxiao@american.edu, Office: SCAN 110.

Time: Mon/Wed/Thu 11:45-1:00
Anderson B14

Office hours: Tentatively:
Mondays: 4-5pm
Thursdays 4-6pm

Textbooks and references:
We will use the textbook mainly as a reference, but we will not strictly follow any textbook.

1. How to think like a computer scientist, learning with Python. Allen Downey. The online version of the book is here: http://openbookproject.net/thinkcs/python/english2e/

2. Introduction to Computation and Programming Using Python. John Guttag. Spring 2013 edition. MIT Press. (Recommended, not required).

3. The official Python tutorial. https://docs.python.org/2/tutorial/

4. Learning Python the Hardway (very good exercises and tutorial):

Online discussions:
Sign up for Pizza here to join online discussions about lectures and homework:
piazza.com/american/fall2015/csc280001

Computer and software:
Even though computers are provided in the classroom, you are extremely encouraged to bring your own laptop.

Perquisite: None. This course is aimed at students with little or no programming experiences. Interests in solving simple math and high school algebra would be useful.

Course Description:
This course teaches programming mainly using the Python language. We will cover programming fundamentals (variables, conditionals, iteration, functions), object-oriented programming, and graphical programming. In this course we will also look

beyond programming to computer science as a discipline, exploring diverse topics such as artificial intelligence, web development, natural language processing, numerical methods, and how computers work. If time allows, we will also introduce a little bit of web programing, including languages such as HTML, CSS and JavaScript.

Classes: Programming is very much a skill learned by practice and examples. We will be spending many of the lectures working through example programs. We will dedicate at least one session each week to solving problems either as individual or in teams.

Attendance: Because the class is highly active in teamwork and possible student presentations and in-class quiz, attendance is extremely important and only viewing slides are not sufficient to acquire the necessary skills to be successful in the class. If you miss 3 classes without official proof of evidence (doctor's notes, etc), you will receive 0% in your attendance. Of course, if you miss a class, you are likely to miss the in-class quizzes.

Expected Learning outcomes:

1. Be fluent in the use of procedural statements — assignments, conditional statements, loops, method calls — and arrays.
2. Understand basic data types such as strings, lists file and be fluent in use of functional programming.
3. Understand the concepts of object-oriented programming as used in Python: classes, subclasses, properties, inheritance, and overriding.
4. Have knowledge of basic searching and sorting algorithms. Have knowledge of the basics of vector computation.

Grading: 50% Homework Assignments (6 projects), 10% Labs, 15% in-class Mid-term exam (short programming exercises), 15% Final project/exam, 5% in-class quizzes (randomly timed), 5% attendance.

Programming projects will be typically graded as follows:

10% - *Did you make a reasonable effort and submit something?*
40% - *Does it compile without errors?*
10% - *Does your code have sufficient and meaningful comments?*
20% - *Did you follow the appropriate structure?*
20% - *Does your program work correctly?*

A final letter grade will be converted from the percentage you receive through out the course.

Grading Scale listed below:

94-100	A
90-93	A-
85-89	B+

81-84 B
76-80 B-
70-75 C+
65-69 C
60-64 C- cut off for CS major in order to receive credits for major requirement.
50-59 D
0-50 F

Homework Policy:

1. Homework is all about programming and you will submit to blackboard. In the first week, I will pass out a handout code template of how to comment, name, and structure your code for homework assignment. Please follow the instructions strictly. If you name your code "homework question1" without your name, it will not be graded. You must test your code on your computer. We will use libraries extensively. You should also name your folder as lastname_firstname when submit to blackboard.
2. Teamwork (less than 3) is encouraged if specified in the homework. Otherwise, only high-level discussion is allowed.
3. No plagiarism (see Academic integrity below): only high-level discussions are allowed (i.e., not relating to a single line of code) across individuals or teams.
4. Please use **Python 2.7.** as most course demos will be done in Python2.7 so is your textbook. If you use Python 3, please note that it is not backward compatible with Python 2.x.
5. Please restrict homework related questions to office hours. If you are confused of what the homework is about, ask me last least **2 days** before the due date. Last-minuet emails regarding homework (less than 24 hours) might not be promptly answered.

Late Policy:
Home works and Projects must be submitted to blackboard **by 11:59 pm** on the due date to receive credit**. Late projects will generally not be accepted**, unless permission is sought before the project is due.

Missed exams may not be made up. If a true emergency forces you to miss an exam, permission to be excused must be sought in writing from me in advance, if possible. If granted, the final exam score will replace the score of the missed exam (or the midterm score, if the final is missed).

Exam Policy:
Mid-term exam will be announced at least one week ahead of time. If you have special needs, you need to notify me at least **5 days** before to arrange the test be performed off-class in the exam center. Missed exams cannot be made up.

Academic Integrity:

Plagiarism and academic misconduct are defined in the University Academic Integrity Code. You should be familiar with what constitutes academic dishonesty. In particular, you should observe the following rules: Collaboration on projects is restricted (if you have high-level discussion with another person, please write down the name of the person). Any information taken from the Internet, books, or anywhere else for use on your assignments must be cited. Your code must be entirely your own work. All exams will be close-book, close-note, no Internet, no smart phones. Instances of plagiarism may be reported and could result in disciplinary action.

Feedback:

I am eager to receive suggestions on how I can improve this course. If you have any comments, please feel free to tell me, either in person, by email, or anonymously in my mailbox in SCAN 108.

Approximate Course Outline (subject to modifications)

Date	Subject	Readings
Week 1:	Introduction to Programing	Chapter 1
Week 2:	Thinking "computationally"	
	Starting Python: "Hello World"	
Week 3:	Variables	
	Loops	
	Data types	
Week 4:	String and text	
	String applications	
Week 5:	Lists	
	Files	
Week 6:	Graphics	
	Mouse	
Week 7:	Functions	
	Function return	
Week 8:	If	
	Midterm around this time	
Week 9:	While	
	Booleans	
Week 10:	Objects	
	Classes	
Week 11:	More classes	
	Class applications	
Week 12:	Recursion	
Weeks 13:	Numeric Python (Numpy, Scipy)	
Week 14:15	Extra Topics	

Extra topics might include:

- Sorting Algorithms
- Search Algorithms
- Image processing
- Hacking
- Web programming
- Plotting Scientific Data