Syllabus

CSci 127: Introduction to Computer Science
Hunter College, City University of New York
Spring 2020

General Information

Description: *3 hours, 3 credits:* This course presents an overview of computer science (CS) with an emphasis on problem-solving and computational thinking through 'coding': computer programming for beginners. Other topics include: organization of hardware, software, and how information is structured on contemporary computing devices. This course is pre-requisite to several introductory core courses in the CS Major. The course is also required for the CS minor. MATH 12500 or higher is strongly recommended as a co-req for intended Majors.

Grading Policy

Expectations: Completing homework is an essential part of the learning experience. Students are expected to learn both the material covered in class and the material in the online Labs, the textbook and other assigned reading.

Honor Code: You are encouraged to work together on the overall design of the programs and homework. However, for specific programs and homework assignments, all work must be your own. You are responsible for knowing and following Hunter College's Academic Integrity Policy:

> *Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.*

All incidents of cheating will be reported to the Office of Student Conduct in the Vice President for Student Affairs and Dean of Students office.

Course Format: This course is taught in a **hybrid (blended)** format with a third of the in-class meetings replaced with online activities. Each week the course meets:

- 1.5 hours of lecture (Tuesdays, 9:45am-11:00pm, Assembly Hall-- 118 HN),
- 0.5 hours of in-person assessments (appointments and drop-ins between Mondays-Fridays, 11am-6:30pm, 1001E HN), and
- 1.0 hours of **Online Labs to be completed by the students independently** (links to the labs can be found in the Course Outline).

Lecture Preview: Before every lecture, there is a short on-line quiz based on the material for the week. The preview is available on Blackboard the day before lecture and can be repeated as many times as you

would like until the start of lecture. Turning in lecture previews will only help your grade. There are no make-ups for lecture previews. Instead, if you miss a lecture preview, your grade on the final exam will replace that grade.

Lecture Participation: At every lecture (Tuesdays, 9:45am-11:00pm, 118 HN), there will be a lecture slip to be submitted, either electronically or on paper. Completed lecture slips are given full credit. Turning in lecture slips will only help your grade. There are no make-ups for lecture slips. Instead, if you miss a lecture slip, your grade on the final exam will replace that grade.

Quizzes: Every week, there will be an quiz on the lecture notes, reading, submitted programs, and laboratory exercises.

- As the semester progresses, quizzes will include review questions as well as short programming exercises based on the homework.
- All quizzes must be taken by the deadline.
- There are no make-up quizzes. Instead, your score on the final exam will replace missing quiz grades (the final exam will also replace a quiz grade when you take the quiz but do better on the final exam).
- Quizzes are on paper (echoing the style of the final exam and the first pass of many programming job interviews).
- For the quizzes, you may not use any notes, books, or any computer device (laptop, phone, calculator, smart watch, etc.).
- Quizzes are offered 11am-6:30pm Mondays to Fridays, when classes are in session. You may sign up for a time slot in advance or drop in. Advance sign-up for quizzes will be available a week before classes start (see class Blackboard page for links).

Code Reviews: Every week, there will be an in-person code review on the recently submitted programs.

- For each code review, you choose one of the programs due the previous week to explain on the whiteboard.
- All code reviews must be taken by the deadline.
- There are no make-up code reviews. Instead, your score on the final exam will replace missing code reviews (the final exam will also replace a code review grade when you take the code review but do better on the final exam).
- Code reviews are on the whiteboard (echoing the style of many programming job interviews).

Laboratory Exercises: Each week, you are expected to work through the associated lab exercise. The on-line labs are available on the course webpage can be completed in the lab or at home (you will need a computer with Python and Unix shell, and as the term goes on, C++).

Homework: Programming exercises are posted on the class website, usually three weeks before the due date. They reinforce concepts covered in lecture and lab. Note that as the semester progresses, the programs will require work on design and programming outside of class to complete. To receive full credit for a program, the program must perform correctly, must include comments, be written in good style, and be submitted by **6pm deadline**. Programs are submitted via gradescope. You can miss up to 5 programming assignments without affecting your grade (if you turn in all the programming assignments, we will drop the lowest 5 scores). **No late homework is accepted.**

Final Exam: The final exam is required. It is comprehensive, covering all the material of the course. Sample and past exams are available on the course webpage. We will end most lectures with past exam questions and review to prepare for the final. **You must take and pass the final to pass the course.**

Grades: The grading for the course will be based on:

- Problem Sets: 30%.

- (Paper) Quizzes: up to 25% (if you miss a quiz, or do better on the final exam, your grade on the final will replace that paper quiz grade).
- Code Reviews: up to 5% (if you miss a code review, or do better on the final exam, your grade on the final will replace that code review grade).
- Lecture Previews: up to 5% (if you miss a lecture preview, or do better on the final exam, your grade on the final will replace that lecture preview grade).
- Participation (Lecture Slips): up to 5% (your grade on the final will replace the grades for missing or incomplete lecture slip).
- Final Exam: 30% (up to 70% if you do better on the final exam than on quizzes, code reviews, lecture previews, and lecture slips).

Materials, Resources and Accommodating Disabilities

Textbook & Readings: The following free on-line book is required for the course:

- How to Think Like a Computer Scientist (Python 3) (free on-line textbook).

Additional readings and tutorials are available on the course webpage.

Technology: This course uses multiple software tools and languages. All are available on the lab computers.

- Python: a programming language that is freely available across multiple platforms. See Lab 1 for details on obtaining it for your home computer.
- Unix: an operating system that underlies Ubuntu Linux (what the lab computers run) as well as MacIntosh OSX.
- Blackboard: the learning management system used throughout CUNY. Accounts are provided automatically to all enrolled students. In this course, in-class quizzes will be given via Blackboard.
- Gradescope: an automatic grading program from UC Berkeley. All programming assignments will be submitted to gradescope. An email with access information will be sent to your email of record on Blackboard two days before the semester starts.
- Github: a freely available site for hosting and collaborating, particularly for programming.
- C++: a programming language that is freely available. We will use it in the last third of the course. Information on downloading it for your computer will be available in Lab 12.

Computer Access: Part of this course will use university computer laboratories. These machines are for work related to this course only and a code of conduct applies to computer use in the department and on-campus. Misusing university computers could result in losing your computer access for the rest of the term, making it exceedingly difficult to complete this course.

Tutoring: The CSci 127 course has a dedicated laboratory, 1001E North, and a staff of undergraduate teaching assistants to provide tutoring and assistance with the course.

Accommodating Disabilities: In compliance with the American Disability Act of 1990 (ADA) and with Section 504 of the Rehabilitation Act of 1973, Hunter College is committed to ensuring educational parity and accommodations for all students with documented disabilities and/or medical conditions. It is recommended that all students with documented disabilities (Emotional, Medical, Physical, and/or Learning) consult the Office of AccessABILITY, located in Room E1214B, to secure necessary academic accommodations. For further information and assistance, please call: (212) 772- 4857 or (212) 650-3230.

Hunter College Policy on Sexual Misconduct: In compliance with the CUNY Policy on Sexual Misconduct, Hunter College reaffirms the prohibition of any sexual misconduct, which includes sexual

violence, sexual harassment, and gender-based harassment retaliation against students, employees, or visitors, as well as certain intimate relationships. Students who have experienced any form of sexual violence on or off campus (including CUNY-sponsored trips and events) are entitled to the rights outlined in the Bill of Rights for Hunter College.

- Sexual Violence: Students are strongly encouraged to immediately report the incident by calling 911, contacting NYPD Special Victims Division Hotline (646-610-7272) or their local police precinct, or contacting the College's Public Safety Office (212-772-4444).
- All Other Forms of Sexual Misconduct: Students are also encouraged to contact the College's Title IX Campus Coordinator, Dean John Rose (jtrose@hunter.cuny.edu or 212-650-3262) or Colleen Barry (colleen.barry@hunter.cuny.edu or 212-772-4534) and seek complimentary services through the Counseling and Wellness Services Office, Hunter East 1123.

See CUNY Policy on Sexual Misconduct Link.

Course Objectives

The successful student will be prepared with competencies and knowledge required for subsequent courses required for the Computer Science Major or Minor. At the end of the course, students should:

1. be able to design and implement a computer program in Python of realistic complexity that includes functions, list/array data structures, user and file I/O, loops and conditionals.
2. be able to design and implement a simple C++ program using command line tools in a Linux environment, including navigating the Linux file system.
3. understand the basic architecture of a digital computer to the extent that they can write a simple machine language program for a virtual architecture.
4. be fluent in hexadecimal and binary numbering schemes.
5. be able to understand boolean logic to the extent that they can design a simple binary circuit.
6. understand the relationship between the operating system, application and utility software and how they interact with main memory, disk memory and the software development cycle.
7. have been exposed to a small selection of more advanced computer science topics such as artificial intelligence, data science, networking, algorithm and data structure design, etc.

CSci 127: Introduction to Computer Science
Hunter College, City University of New York
Spring 2020

This course presents an overview of computer science (CS) with an emphasis on problem-solving and computational thinking through 'coding': computer programming for beginners. Other topics include: organization of hardware, software, and how information is structured on contemporary computing devices. This course is pre-requisite to several introductory core courses in the CS Major. The course is also required for the CS minor. MATH 12500 or higher is strongly recommended as a co-req for intended Majors.

Course Designers:

- Dr. Katherine St. John
- Dr. William Sakas
- Prof. Eric Schweitzer

Course Instructors:

- Dr. Tiziana Ligorio (office hours)
- Katherine Howitt, adjunct lecturer

IMPORTANT:

The course has **lecture on Tuesday mornings** and **required weekly quizzes & code reviews in Lab 1001E** (see lab hours below and your Blackboard account for scheduling information).
This is a HYBRID course which means that **you are responsible for independently reading the weekly Lab** found in the "Handouts" column in the Course Outline below.
Starting February 6, **there is a programming assignment due EVERY DAY**. Programming Assignments are directly related to Labs and Lectures.
Lecture notes will also be posted weekly in the "Handouts" column in Course Outline below.

Lecture: Tuesday 9:45-11:00 118 HN Assembly Hall

Lab Hours: There is a dedicated computer laboratory, North 1001E for this course:

- Staffed Hours: Monday-Friday, 11am to 6:30pm, when classes are in session.
- Reservation link available on Blackboard on the left sidebar.
- Holiday, Lab Schedule and Program Due Dates (grid format; more details in the list format below)
- Undergraduate Teaching Assistants

Useful Links:

- Syllabus
- Programming Assignments
- Quiz & Code Review Topics and Deadlines

- ASCII Table
- Book & tutorial pages:
  - Python: How to Think Like a Computer Scientist by Miller *et al*.
  - Logical Circuits: Burch's Logic & Circuits, Explain Logic Gates
  - Machine Language: U Idaho reference sheet, MIPS Wikibooks
  - C++: Cplusplus Tutorial, C++ Tutorials Point, The Rook's Guide to C++

Course Outline:

Week:		Topics:	Handouts:	Quiz & Code Review:	Reading:
#1	**Lecture:** 28 January	Syllabus & Class Policies, Introductions, Introduction to Python: definite loops, simple output, primitive data types, overview of objects & modules; What is an algorithm?	Syllabus, Programming Assignments, Hello, World, Hexagon example, Fancier hexagon, Lecture Notes		Think CS: Chapter 1 & Chapter 4
	Lab & Quiz: 28-31 January, 3 February	Getting started with Python & IDLE; Using modules and definite loops	Lab 1	Academic Integrity	
#2	**Lecture:** 4 February	Strings & Lists: looping through strings, console I/O, ASCII representation CS Survey: Prof. William Sakas (computational linguistics)	Loop Puzzle 1, Loop Puzzles 2, Caesar Cipher example, input() example, Lecture notes		Think CS: Chapter 2 & Chapter 3
	Lab, Quiz, & Code Review: 4-10 February	String methods; Problem solving and the design process (simple parsing and translating)	Lab 2	Loops & Turtles	
#3	**Lecture:** 11 February	Arithmetic; Indexing & Slicing; Colors, Hexadecimal notation; Prof. John Ranellucci, Educational Psychology	Event Timing (Arithmetic Challenge), Slicing Challenges, Color Challenges, Lecture notes		Think CS: Section 8.11 & Chapter 11, Numpy tutorial (DataCamp)
	Lab, Quiz, & Code Review: 11-14 February	Arrays and images in numpy, hexadecimal representation of colors (image processing)	Lab 3	Strings & Loops	
12 February	Lincoln's Birthday: Lab closed				
17 February	President's Day: Lab closed				

#4	**Lecture:** 18 February	More on Lists & Arrays; Images; Decisions; Airplane Design	Loop & Slice Challenges, Decision Challenges, turtleString.py, Lecture notes		Think CS: Chapter 7 & Chapter 11
	~~**Lab, Quiz, & Code Review:**~~ 18-24 February	Programming with decisions & files (flood maps)	Lab 4	Loops & Unix	
#5	**Lecture:** 25 February	Logical Expressions, Circuits, Binary Numbers; CS Survey: Bernard Desert & Elise Harris (CUNY2X@Hunter)	Types & Decisions Challenges, Logical Operators Challenges, SemesterIfAndExample, Basic Gates, Circuit Challenge1, Circuit Challenge2, Circuit Challenge3, Lecture notes		Think CS: Chapter 7, Burch's Logic & Circuits, Explain Logic Gates
	Lab, Quiz, & Code Review: 25-28 February, 2 March	More on Decisions (snow pack); Circuits & Logical Expressions	Lab 5	Decisions & Color	
#6	**Lecture:** 3 March	Accessing formatted data; CS Survey: Prof. Kelle Cruz (Astrophysics)	Arithmetic Challenges, List/String Challenges, Lecture notes		Think CS: Chapter 6, 10-minutes to Pandas Tutorial, DataCamp Pandas Tutorial, Ubuntu Terminal Reference Sheet
	Lab, Quiz, & Code Review: 3-9 March	CSV files via pandas (population change); Shell Scripts, github	Lab 6	Circuits, Truth Tables, & Logical Expressions	
#7	**Lecture:** 10 March	Functions; NYC OpenData CS Survey: Brian Campbell, Hunter College Alumnus and Software Engineer at Seamless	Motto Challenge, quarterImage.py, Hello with main(), Prep #1.2, Total & Tax Challenge, Greet Example, Happy Example, Jam Example, Month String Example, NYC OpenData Lecture notes		Think CS: Chapter 6, 10-minutes to Pandas Tutorial, DataCamp Pandas Tutorial
	Lab, Quiz, & Code Review: 10-16 March	OpenData NYC (shelter data); Using `main()` functions; Python from the command line	Lab 7	Formatted Data & Shell Commands	
12-19	Instructional Recess - No Lecture, Online help via email and Blackboard Discussion Board				

April	available				
#8	**Lecture:** 24 March	More Functions & Parameters;	Decisions & Functions Example, Dessert Exam Questions, Foo example, Koalas, Lecture notes		Think CS: Chapter 6
	Lab, Quiz, & Code Review: 24-26 March	Binning data (parking tickets); Top-down design (herd of turtles); Command line `git`	Lab 8	Functions & More Pandas	
27 March - 1 April	Recalibration Period - No Lecture, No Online Help				
2-6 April	Online Help and Code Review 8 resume				
7 April	Wednesday at Hunter - No Lecture				
8-10 April	Spring Recess - No Online Help				
#9	**Lecture:** 14 April	Programming with Functions, Top-down Design; Mapping GIS Data (Folium); Random Number Generation; Preview: Indefinite Loops	Sisters Example, numsConvert.py num2string example, Distance Check, Random Walk, Lecture notes		Think CS: Chapter 6, folium tutorial
	Lab, Quiz, & Code Review: 14-20 April	Folium/leaflet.js (mapping CUNY locations); Finding errors; Regular expressions (command line)	Lab 9	Parameters & Functions	
#10	**Lecture:** 21 April	Indefinite Loops; Simulations; Design Patterns: Max;	Nums & While, Max Num, Random Search (turtles), Lecture notes		Think CS: Chapter 8
	Lab, Quiz, & Code Review: 21-27 April	More on Indefinite loops; Writing functions; unit testing	Lab 10	More on Functions & Top-down Design	
#11	**Lecture:** 28 April	Python Recap; Simplified Machine Language; Design Patterns: Searching;	Search, WeMIPS Emulator,		U Idaho reference sheet, MIPS Wikibooks
	Lab, Quiz, & Code Review: 28 April - 4 May	Simplified machine language	Lab 11	Indefinite Loops & Simulations	
#12	**Lecture:** 5 May	Introduction to C++: program structure, data	cin/cout example, convert example,		Cplusplus Tutorial,

		representation and I/O. Final Exam Overview	loops example, growth example, nested loops,		C++ Tutorials Point, The Rook's Guide to C++
	Lab, Quiz, & Code Review: 5-11 May	Using gcc	Lab 12	Simplified Machine Language & More Unix	
#13	**Lecture:** 12 May	C++ control structures	Decision example (C++), Logical Expressions (C++), Input Checking (C++), Input Checking, II (C++), Growth Example (C++),		Cplusplus Tutorial, C++ Tutorials Point, The Rook's Guide to C++
	Lab, Quiz, & Code Review: 12-14 May	Control Structures in C++	Lab 13	Introduction to C++	
MONDAY 18 May 9am-11am		Final Exam	Final Exam Information		

(This file was last modified on 27 January 2020.)