Contents

1 Course Information

1.1 Website

https://www.swamiiyer.net/cs110/

1.2 Catalog Description

An introduction to computer programming – the concepts involved in using a high-level language and the program development process. The goal of this course is proficiency in the design and implementation of programs of significant size and complexity. This course is quite demanding because of the length of the programming exercises assigned. This is the first course in the computer science major sequence.

Prerequisites: Math 140 credits or placement; or Math 130 with a B or higher in the previous semester; or permission of the instructor.

Students who successfully complete this course will be able to tackle computational challenges that they might encounter later in their careers. Students interested in computer science will be well-prepared to delve deeper into the field and students in science and engineering will be able to incorporate computation into their studies.

1.3 Course Staff

Swami Iyer will be the primary instructor for the course. He will be assisted by graduate teaching assistants (TAs), undergraduate/graduate course assistants (CAs), and an undergraduate supplemental instruction (SI) leader.

1.4 Class

In each class, the instructor will present the material for that class for an hour, and conduct an online quiz on that material for the remaining fifteen minutes. You are strongly encouraged to review the material before each class, and review it again thoroughly soon after.

1.5 Discussion

Starting from the second week, there will be a discussion every week. The focus of the discussion for a particular week will be the current assignment. The teaching assistant (TA) will walk you through the assignment problems systematically. The TA will also answer any specific questions you may have about the assignment or the course material in general. You may also seek help from the course assistant (CA) who will be assisting the TA during the discussions. The discussions will be worthwhile only if you go to the sessions having read the assignment writeup thoroughly and have at least a moderate understanding of the problems involved. The TA will assume that you have done the reading in advance.

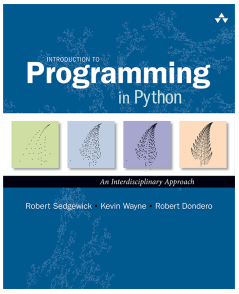
1.6 Supplemental Instruction

As part of the College of Science and Mathematics Freshman Success Program, supplemental instruction (SI) is available to all CS110 students free of charge. The SI sessions will also start from the second week. The focus of the sessions for a particular week will be the material covered in class during the previous week. The SI leader will walk you through the relevant lecture notes and solve problems. In addition, the SI leader will answer any specific questions you may have about the current assignment, or the course material in general. The SI sessions are optional, but highly recommended, especially if you feel like you are falling behind in the course. You may attend as few or as many sessions as you like. You will receive extra points for attending the sessions (see the Grading Scheme section below).

1.7 Tutoring

One-on-one tutoring for this course is available through the Tutoring Programs.

1.8 Recommended Text



Introduction to Programming in Python: An Interdisciplinary Approach by Robert Sedgewick, Kevin Wayne, and Robert Dondero

This text offers an excellent introduction to computing principles, motivating each principle by examining its impact on specific applications drawn from fields ranging from materials science to genomics to astrophysics to internet commerce.

1.9 Grading Scheme

1.9.1 Assessments

Item	% of Final Grade
Programming Assignments (best 5 out of 6)	40
Exams (1 and 2)	50
Participation	10

- The goal of the programming assignments is to make sure that you can apply the concepts learned in class to solve somewhat challenging computational problems.
- The exams will test your understanding of the material covered in class as well as concepts from the programming assignments. Each of the two exams will have a written and a programming component. The written part, worth 75 points, will take place during a class period. The exam will be closed-book and will contain 30 multiple-choice/short-answer questions, each worth 3 points. The programming part, worth 25 points, will take place during a discussion session; you will use your laptop to work on the problems and submit your solutions onto Gradescope.
- Your participation score will be based on weekly quizzes (8%) and discussion attendance (2%). Each weekly quiz, conducted at the end of a class, will test your understanding of the material covered in that week. Each question in a quiz is worth 2 points (1 point for correct response and 1 point for responding). Each quiz score will be normalized to 100 points. The lowest 2 (4 if you have accommodations) quiz scores will be dropped.
- You can earn up to 2% extra points for attending the SI sessions. Your SI score will be calculated as $\frac{a(e_1+e_2)}{100n}$, where a is the number of unique sessions you attended (attending multiple sessions in a week just counts as one), n is the number of unique sessions held, and e_1 and e_2 are your Exam 1 and Exam 2 scores, respectively. For example, if $a = 10, n = 12, e_1 = 75$, and $e_2 = 85$, the SI score is 1.33%.
- You can earn 0.01x% extra points if x% of the class completes the end-of-semester course evaluation.
- If your overall score falls within half a percent of a higher grade, your score will be elevated to that grade.

1.9.2 % Score to Letter Grade

`[93, 100]: A, [90, 93): A-, [87, 90): B+, [83, 87): B, [80, 83): B-, [77, 80): C+, [73, 77): C, [70, 73): C-, [67, 70): D+, [63, 67): D, [60, 63): D-, [0, 60): F`

1.10 Software Needed

1.10.1 iClicker

We will use iClicker to conduct in-class quizzes, for which you will need an iClicker Student App Subscription.

1.10.2 Piazza

We will use Piazza as the Q&A platform for the course. If you have any general questions about the assignments, exams, or the lecture material, the most effective way to get them answered is by posting them on Piazza. You can expect your questions to be answered by the course staff or your peers.

1.10.3 Gradescope

We will use Gradescope to grade your programming assignments and exams.

1.10.4 Programming Environment

To write and execute Java programs in this course, you will need a laptop (Linux, Mac, or Windows) properly configured with the necessary software. Click here for setup instructions.

1.10.5 Zoom

We will use Zoom to hold remote office hours and SI sessions.

1.11 CS Account

In order to use the computing resources of the department, and in particular, those in the UNIX/PC Lab (`M-3-0731`), you need to setup a CS account. With your CS account credentials, you can connect to our designated server (`users.cs.umb.edu`) remotely using SSH. With the same credentials, you can also sign into the Linux systems in the CS Lab. In addition, you can sign into the Windows systems in the lab with the same username and an initial password `abcd_1234`, which you must change the first time you sign in.

Visit CS Labs Portal to register for a portal/CS account and confirm via email. If you already have a CS account, use the same username. The next step is to sign into the portal and select your courses for the term. You will be notified via your UMB email once the course directories and your account are created.

1.12 Policies

1.12.1 Classroom

Come to class/discussion on time and stay for the entire session. If you have to leave early, let the instructor/TA know in advance. Have your mobile phone silenced or turned off during the entire session. Use of earphone/headphone during the session is not permitted. Use of a laptop during the session is permitted only for class purposes. Do not talk to each other during the session. If you have any questions, bring them up to the instructor/TA.

1.12.2 Piazza

If you have a question, first make sure that it has not already been asked/answered. Clearer questions get better answers, so re-read your question before you post it. Ask your questions early. Posts are categorized using channels, so pick an appropriate tag for your post. Use the platform only for questions that can be asked in a general way, without sharing code or other assignment-related work. However, if you are stuck on a problem despite your valiant efforts to solve it, you may seek help from the course staff by posting your code privately, as properly formatted text (not images). Any post that is inappropriate or violates the academic honesty code will be deleted by the course staff.

1.12.3 Excused Absence from Discussion and Makeup Exam

You must provide appropriate documentation if:

- You could/will not attend a discussion and want your absence to be excused.
- You were/are unable to take an exam on the scheduled date and want to arrange a makeup exam.

The documentation must be a letter from the Dean of Students if the type of your absence is among those listed on their website. For other types of absences, the supporting documentation must be emailed to the instructor directly.

Note: There will be no makeup for missed quizzes.

1.12.4 Late Days

Assignment deadlines are firm and non negotiable, but you have up to 5 (10 if you have accommodations) late days that you may use to submit one or more assignments a bit late. You may request an extension for any assignment only once. To do so, you must send the instructor a request via email at least 48 hours prior to the assignment deadline, clearly stating the number of late days needed, or else your request will not be granted.

Unused late days will be converted into $x\%$ extra points, which will be added to your final grade. x will be calculated as $x = \frac{d}{n}$, where n is the number of late days allotted and d is the number of late days unused. For example, if $d = 3$ and $n = 5$, then $x = 0.6\%$.

1.12.5 Regrade Request

If you have any concerns about the grading of a particular assignment or exam, you may submit a regrade request via Gradescope. You must submit the request within a week from the date the assignment or exam grades are published, or else your request will be turned down.

1.12.6 Collaboration

Click here for the collaboration policy and the penalties for infractions of the policy.

1.12.7 Accomodations for Students with Disabilities

Section 504 of the Americans with Disabilities Act of 1990 offers guidelines for curriculum modifications and adaptations for students with documented disabilities. If applicable, students may obtain adaptation recommendations from the Ross Center for Disability Services. The student must present these recommendations and discuss them with the instructor within a reasonable period, preferably by the end of Add/Drop period.

1.12.8 Campus Closure

In the event of a campus closure, all class-related activities will be conducted remotely, via Zoom. If there is an exam scheduled to take place on that day, the exam will be postponed to the next suitable date.

2 Topics Covered

- Course Mechanics *[Lecture 1]*
- Programming Environment *[Lecture 1]*
- Chapter 1: Building a Computer
  - Representing Information *[Lecture 2]*
  - Logic Circuits *[Lecture 3]*
  - Von Neumann Architecture *[Lecture 3]*
- Chapter 2: Imperative Programming
  - Your First Programs *[Lecture 4]*
  - Basic Data Types *[Lecture 5]*
  - Control Flow *[Lecture 6 and 7]*

  - Collection Data Types *[Lecture 8 and 9]*
  - Input and Output *[Lecture 10 and 11]*
- Chapter 3: Procedural Programming
  - Defining Functions *[Lecture 12 and 13]*
  - Libraries and Applications *[Lecture 14 and 15]*
  - Recursion *[Lecture 16]*
- Chapter 4: Object-oriented Programming
  - Using Data Types *[Lecture 17 and 18]*
  - Defining Data Types *[Lecture 19 and 20]*
  - Design Principles *[Lecture 21 and 22]*
- Chapter 5: Algorithms and Data Structures
  - Analysis of Algorithms *[Lecture 23]*
  - Searching and Sorting *[Lecture 24 and 25]*
  - Basic Data Structures *[Lecture 26 and 27]*

3 Assignments

3.1 The List

There are 6 programming assignments in all. These are due at midnight (11:59 PM to be precise) on the dates indicated on the Calendar page of the course website.

#	Title	Goal
1	Straight-line Programs	Implement programs *without* branches and loops.
2	Control-flow Programs	Implement programs *with* branches and loops.
3	Mozart Waltz Generator	Implement Mozart's waltz game by writing a program to generate a two-part waltz and another program to play the waltz.
4	RSA Cryptosystem	Implement the RSA public-key cryptosystem.
5	Atomic Nature of Matter	Re-affirm the atomic nature of matter by tracking the motion of particles undergoing Brownian motion, fitting this data to Einstein's model, and estimating Avogadro's constant.
6	Markov Model	Use a Markov chain to create a statistical model of a piece of English text and use the model to generate stylized pseudo-random text and decode noisy messages.

3.2 Submitting Your Work

You will use Gradescope to submit your Python programs (ie, `.py` files) and the `notes.txt` file. Make sure that you only submit files listed under the "Files to Submit" section of the assignment writeup.

You may submit your files as many times as you like, up until the assignment deadline. The most recent submission is considered active by default and your score on the active submission is your official score for the assignment as well. You have the option of making any of your previous submissions active.

Note: If your active submission is partial, your assignment score will also be partial, so in order to be eligible for full credit, make sure you have an active submission containing all the required files for the assignment.

3.3 How the Assignments will be Scored

3.3.1 Correctness

Your solution to each assignment problem will be evaluated for correctness by an autograder. Each test that is used for this purpose is worth some number of points; your solution will receive all the points from a test that passes and 0 points from a test that does not pass. Your overall correctness score will be normalized to 80 points.

3.3.2 Clarity and Efficiency

Your solution to each assignment problem will additionally be checked by a TA for clarity and efficiency. Your code will receive 10 points if it meets our expectations (must include adequate comments, must follow good programming principles, and must meet any problem-specific requirements), and will be marked down otherwise.

3.3.3 Notes File

The given `notes.txt` file for an assignment must be uploaded with the three sections (#1 mandatory, #2 if applicable, and #3 optional) filled in as appropriate. In section #1, for each problem, you must include in nore more than 100 words: a short, high-level description of the problem; your approach to solve it; and any issues you encountered and if/how you managed to solve them. Your notes file will receive 10 points if it meets our expectations and will be marked down otherwise.