Syllabus

Course Description

Welcome to your first programming course! CS 220 / CS 319 (Data Science Programming I) is a gentle introduction to coding for students outside of Computer Science. Our goal is simple: to write Python code to answer questions about real datasets. CS 220 / CS 319 will require you to practice coding a lot this semester. CS 220 students will complete 13 programming projects. CS 319 students will complete 9 programming projects and work on a self-chosen graduate level project. It's a lot of work, but if you take this course seriously and invest the time, you'll walk away with an incredible new skill: the ability to make computers work on your behalf. Students are required to have access to a laptop with MAC or Windows OS, where they can install Python, in order to complete projects for this course.

CS319 meets with CS220. Common aspects between CS319 and CS220: lectures, quizzes, exams, and first 9 programming projects.

Additions To Syllabus Made During Semester

- No additions made yet.

Readings

We'll assign readings from four main sources this semester (all free). Stay on top of them!

- *Python for Everybody* by Barb Ericson, et. al.: Read Online
- *Think Python 2nd Edition* by Allen B. Downey: Read Online
- *Automate the Boring Stuff with Python* by Al Sweigart: Read Online
- Course Notes (we'll sometimes write these ourselves)

Course Instructors

- Mike Doescher (Teaching Faculty - Department of Computer Sciences) mdoescher@wisc.edu
- Gurmail Singh (Teaching Faculty - Department of Computer Sciences) gurmail.singh@wisc.edu
- Andrew Kuemmel (Teaching Faculty - Department of Computer Sciences) kuemmel@wisc.edu

Lectures (Meeting Time and Location)

- LEC001 204 EDUC SCI MWF 08:50 AM - 09:40 AM
- LEC002 2650 HUMANITIES MWF 11:00 AM - 11:50 AM
- LEC003 1310 STERLING HALL MWF 01:20 PM - 02:10 PM
- LEC004 B102 VAN VLECK MWF 03:30 PM - 04:20 PM

Lecture recordings will not be provided. In-person attendance is expected (see below for details on lecture attendance).

Instructional Modality

- LEC001 through LEC004: in-person

CS220 Grading (not applicable to CS319 students)

Your overall grade is based on the following:

- Class surveys: 1%
- Lab attendance: 4% (13 labs scheduled, lowest 3 scores dropped)
- 13 Projects (numbered P1 to P13) - overall 47%, **none dropped**
  - P1: 1%
  - P2-P13 (except P9): 4% each
  - P9: 2%
- 10 quizzes (Q1 to Q10): 2% each (lowest 2 scores droppped) - overall 16%
- 3 Exams - overall 32% grades distributed as follows:
  - E1 and E2: 10% each,
  - E3 (Final Exam): 12%.

CS319 Grading (not applicable to CS220 students)

Your overall grade is based on the following:

- Class surveys: 1%
- 9 CS220 Projects (numbered P1 to P9) - overall 31%, **none dropped**
  - P1: 1%
  - P2-P8: 4% each
  - P9: 2%
- Graduate-level project - overall 20%
  - Graduate Project part 1: proposal: 4%
  - Graduate Project part 2: data set: 4%
  - Graduate Project part 3: initial code: 4%
  - Graduate Project part 4: final code: 4%
  - Graduate Project presentation: 4%
- 10 quizzes (Q1 to Q10): 2% each (lowest 2 scores droppped) - overall 16%
- 3 Exams - overall 32% grades distributed as follows:
  - E1 and E2: 10% each,
  - E3 (Final Exam): 12%.

Graded Component Details

Class surveys is based on filling out surveys: start of the semester and end of the semester surveys. Corresponding assignements will be created on Canvas.

Labs (only applicable for CS220 students):

- You will be given a grade for each week's lab which is based on arriving on time, staying until the end of the lab, and participating in the lab activites.
- Every lab attendance is allocated 5 points:
  - 1 point for arriving within the first 5 minutes and **picking up your name tag**
  - 3 points for actively completing the lab
  - 1 point for staying at least until 5 minutes before the end of the lab and **returning your name tag**
- You must follow the process of picking up name tag at the beginning and leaving it at the end. TAs will remind you to do this every lab.
- There are 13 weekly labs.
- We will drop your three lowest lab scores, to account for any absences you may have for health, other coursework, or emergencies. Score drops happen automatically - you do not have to email anyone about missing lab. If you miss a lab, we expect you to complete the lab during your own time.

Projects:

- Primarily focus on your ability to write and debug code.
- Please see the course schedule for project due dates. Projects are due by 12:00am Central Time on the day stated.
- Students are required to have access to a laptop with MAC or Windows OS, where they can install Python.
- **Partnership:**
  - You may work alone, or with one project partner of your choosing. **We strongly encourage you to find a project partner.**
  - CS220 students can partner up with any other CS220 student, no matter which lecture they are enrolled in. We strongly recommend finding a project partner in the same lab. Students are not allowed to find a different partner just for the lab.
  - CS319 students are allowed to partner only with CS319 students.
  - Project partners **must not** work on alternating projects. Project partners **must not** split the project questions and work on separate half of the project. Both of these will be considered as cheating! The point of partners is to learn from your peers, not to do half the work.
  - Ideally, you and your partner would program in pairs (two people sitting in front of a screen at the same time). Take turns driving (i.e., writing code) and giving advice. This process is called pair programming.
- **Submission**:
  - For every project, you will be given an auto-grader tests - embedded into the project notebook file. It is your responsibility to run auto-grader tests on your local laptop, prior to submitting your project.
  - Project submission will happen on GradeScope platform. Additional instructions will be provided on the project notebook file.
  - **Only one partner should upload the project on behalf of both partners**. Uploading two projects will result in our cheating detection tool flagging your work - this is bad.
  - **IMPORTANT:** If your project did not clear auto-grader tests, it is your responsibility to attend office hours to figure out further details and get your project to clear auto-grader tests on time. If your code does not clear auto-grader tests, your project submission cannot be accepted. ***If you have been trying to get help for 24-hours without any luck, you must email your instructor, to get further help***.
- **Late Policy:**
  - Students have a bank of 12 late days for the semester.
  - For a given project, you may use 3 late days without any deduction. After that, 5% deduction per late day, for the next 4 days. Projects which are late by more than 7 days will not be accepted.

    - After the bank runs out, 5% deduction will be applied per late day.
    - If a student's code fails on the server, late day rules take effect.
    - You **may not** use late days on the last project.
    - Late days only apply to projects. They do not apply to Quizzes.
    - Late days are automatically applied and do not need to be requested.
- **Project Grading (Code Review):**
    - A TA will give you detailed comments on specific parts of your assignment. This feedback process is called a "code review". Read your code reviews carefully; even if you receive 100% on your work, we'll often give you tips to avoid losing points for future projects.
    - Project grading is **results-oriented**. That means it doesn't matter how much effort you put it; it only matters how well your code works. This means it is essential that your code runs. If we can't run your code for a project, you'll get a zero on that project, because the auto-grader tests will fail. We'll never fix the code for you, and we'll never manually give a better grade for code that "looks" almost correct. It is your responsibility to make sure your code clears auto-grader tests.
    - Grades will be based on auto-grader tests (test.py released with the project description) that we run and TA / grader evaluation of your code based on a rubric (shared along with the project notebook).
    - Grades will be posted on canvas, typically before Wednesday of every week. That is grades on canvas will be posted closer to a week from the due to date of the corresponding project.
    - You may submit multiple versions of the project prior to the deadline. The TA / Grader will grade the latest submission prior to the deadline. Once the TA / grader grades a version of your project, you **may not** request for a different version to be graded.
    - You must follow the provided directions and rubric to solve the project questions. Otherwise, you will lose points.
    - No resubmissions will be allowed for any of the projects. Once a version of a particular project submission is graded, we are unable to grade another submission for the same project.

Quizzes:

- Help you assess your understanding of course content (lecture and readings).
- Open notes, computer/Internet use allowed, Python interpreter allowed.
- **No collaboration** allowed.
- Quizzes will be on Canvas. The quizzes will be multiple-choice, multiple answer, multiple dropdown, matching or some other relevant canvas question style.
- Quizzes focus on content from recent lectures. Topics will be listed in the quiz instructions.
- Out of the 10 quizzes assigned, we will drop your lowest 2 quiz scores. This is meant to accommodate emergencies, technical difficulties, or other unforseen circumstances.
- Two attempts are provided for every quiz. The highest of the two scores will be recorded.
- Quizzes must be completed before 11:59pm on the date specified. Please see the course schedule for quiz due dates.

Exams:

- Measure your ability to read and understand code.
- All lectures will have in-person exams. Location for in-person exams will be shared via Canvas message about a week before the exam date.
- Exams are closed-book and closed-laptop. The exams will be multiple-choice scantron (use a #2 pencil). They are cumulative.
- You will be allowed one 8.5-by-11 inch note sheet (printed or written) on both sides, which you must turn in with your exam.
- Exam proctoring tools (eg: HonorLock) will be used if you need to take an online exam. Failure to use the proctoring service will result in a zero on the exam.
- Please see the course schedule for midterm and final exam dates.

Letter Grades

At the end of the semester, we will assign final grades based on these thresholds:

- 93% - 100%: **A**
- 88% - 92.99%: **AB**
- 80% - 87.99%: **B**
- 75% - 79.99%: **BC**
- 70% - 74.99%: **C**
- 60% - 69.99% **D**

Letter grade ranges include decimal points, meaning we will **NOT** be rounding off scores at the end of the semester. No extra credit is given in this course.

How to Succeed in This Course

Best way to succeed in this course would be to:

- attend in-person lectures with classroom engagement
- attend and complete all labs prior to attempting projects

- complete projects within the provided deadlines or by using a maximum of 3 late days
- take weekly quiz attempt1 prior to Friday of every week
- take weekly quiz attempt2 at least by Friday of every week
- solve prior semester practice exam questions, to get sufficient preparation for exams

Communication Tools

1. **Piazza:** Here you can ask questions and see questions written by other students. **Do not post code snippets that are more than 5 lines long.** This is considered cheating!
2. **Office Hours:** To visit Teaching Asisstant (TA) / Peer Mentor (PM) / Instructor office hours please see the schedule at: Office Hours Schedule. To access the calendar, you must be logged in with your UW-Madison credentials.
3. **Email:** Email is the ***least-preferred*** way of communication. Fast and easy way to get help in this course would be via Office hours. If you have a question about your project grading, send an email to the TA / grader who graded your project (Reviewer Contact Email). If you don't get a response within 48 hours, you must CC your instructor (see Course Instructors section for email). Please make sure to utilize Office Hours resources prior to sending any email.
4. **Class Forms:** We have various forms for you to tell us things.
5. **Canvas:** We'll periodically upload grades to Canvas. Note that LEC001 through LEC004 are merged into LEC001 for Canvas. We will also be sending personal messages via canvas to convey information such as: exam room assignment and exam scantron results.
6. **Course Email:** You'll receive announcement emails on either compsci220--s23@g-groups.wisc.edu.

Accommodations

Please notify the instructor for your lecture section within the first two weeks of classes:

- If you participate in religious observances that may conflict with course requirements.
- If you have a VISA from the McBurney Disability Resource Center.
- If you have any trouble accessing or using the technologies being used in this course (such as Piazza, Canvas, and the course website).
- If you need special accommodations for exams (or other components of the course). We'll do our best to accommodate you.

Cheating

You shouldn't cheat, but what is cheating? This may not be obvious to people taking a CS course for the first time, so everybody should read this. The most common form of academic misconduct in these classes involves copying code for programming projects. Here's an overview of what you can and cannot do:

Acceptable

- any collaboration with your project partner
- doing worksheets with non-project partners
- copying code examples from online examples that is NOT specific to your project (if project solutions are leaked online, you may not use that). If you copy code, you must cite it in your code with a comment (think of it like citing a quote in a essay -- without the citation, you're plagiarizing).

NOT Acceptable

- **any form of code sharing (including algorithm sharing) with a non-project partner**
- stealing, looking at, copy / pasting, taking picture of non-project partner's code
- project partners working separately on alternate projects
- project partners working separately on one half of the project
- you are **not helping** your friend by **giving** them your code.

Similarity Detection: of course, with 1000+ students, it's hard for a human TA to notice similar code across two submissions. Thus, we use automated tools to looks for similarities across submissions. Such similarity detection is an active area of Computer Science research, and the result is tools that detect code copying even when students methodically rename all variables and shuffle the order of their code. We take cheating detection seriously to make the course fair to students who put in the honest effort.

Citing Code: you can copy small snippets of code from stackoverflow (and other online references) if you cite them. For example, suppose I need to write some code that gets the median number from a list of numbers. I might search for "how to get the median of a list in python" and find a solution at https://stackoverflow.com/questions/24101524/finding-median-of-list-in-python.

I could (legitimately) post code from that page in my code, as long as it has a comment as follows:

```
# copied from https://stackoverflow.com/questions/24101524/finding-median-of-list-in-python
def median(lst):
  sortedLst = sorted(lst)
  lstLen = len(lst)
  index = (lstLen - 1) // 2

  if (lstLen % 2):
    return sortedLst[index]
  else:
    return (sortedLst[index] + sortedLst[index + 1])/2.0
```

In contrast, copying from a nearly complete project (that accomplishes what you're trying to do for your project) is not OK. When in doubt, ask us! The best way to stay out of trouble is to be completely transparent about what you're doing.

Learning Outcomes

By the end of the course, students should be able to:

- Write Python code. We will implement all of the concepts and ideas from the course using Python 3.
- Communicate using Computer Science terminology. Using the language of programming to talk about concepts helps ease the exchange of ideas between programmers.
- Build large projects in small steps. Creating milestones and breaking large programs down into smaller functions makes complicated projects more achievable.
- Process data from existing files without manual processing. Dealing with large datasets is much more efficient and less error-prone when the process can be automated.
- Manipulate quantitative information to create models, and/or devise solutions to problems using multi-step arguments, based on and supported by quantitative information.
- Evaluate models and arguments using quantitative information.
- Express and interpret in context models, solutions and/or arguments using verbal, numerical, graphical algorithmic, computational or symbolic techniques.

Student Grievance Procedure

Any student at UW–Madison who feels that they have been treated unfairly has the right to voice a complaint and receive a prompt hearing of the grievance. The basis for a grievance can range from something as subtle as miscommunication to the extreme of harassment. Ensuring that all students feel welcome and supported is a top priority of our department. The Department of Computer Sciences has developed procedures to handle complaints or incidents that are student-related, such as bias or mistreatment, or academic, such as a grade dispute or incident with an instructor. Information on the CS reporting procedures can be found here: https://www.cs.wisc.edu/report-an-incident/. For assistance in determining options, students can also contact the drop-in staff member within the Dean of Students Office at 608-263-5700, within Bascom Hall, Room 70, Monday–Friday, 8:30 a.m.–4 p.m.

Official Statements Required on the Syllabus

Course Credits

- CS220: 4 credits
- CS319: 3 credits

Requisites

Satisfied Quantitative Reasoning (QR) A requirement or declared in the Professional Capstone Program in Computer Sciences. Not open to students with credit for COMP SCI 301.

Course Designations and Attributes

- Level: Elementary
- Breadth: Natural Science
- L&S credit type: Counts as Liberal Arts and Science (LAS) credit (L&S)
- General education: Quantitative Reasoning Part B

Credit Hour Policy: Traditional Carnegie Definition – One hour (i.e. 50 minutes) of classroom or direct faculty/instructor instruction and a minimum of two hours of out of class student work each week over approximately 15 weeks, or an equivalent amount of engagement over a different number of weeks. This is the status quo and represents the traditional college credit format used for decades. If you have regular classroom meetings and assign homework, reading, writing, and preparation for quizzes and exams, make this choice.

Regular and Substantive Student-Instructor Interaction: Substantive interaction is engaging students in teaching, learning and assessment through at least two of the following: direct instruction, providing feedback on student work, providing information about course content, facilitating discussion of course content, or other substantive interaction. Regular interaction is: predictable and

scheduled interaction with students consistent with the course length (usually at least weekly but more often in a course of short duration). Regular and substantive student-instructor interaction, as defined by the US Department of Education (Within 34 C.F.R. §600.2), is always a requirement of UW-Madison for-credit learning activities. Find more information including examples of regular and substantive instruction.

Official Course Description: Introduction to Data Science programming using Python. No previous programming experience required. Emphasis on analyzing real datasets in a variety of forms and visual communication. Recommended for Data Science majors and other majors

Course Evaluations: UW-Madison now uses an online course evaluation survey tool, AEFIS. In most instances, you will receive an official email two weeks prior to the end of the semester when your course evaluation is available. You will receive a link to log into the course evaluation with your NetID where you can complete the evaluation and submit it, anonymously. Your participation is an integral component of this course, and your feedback is important to me. I strongly encourage you to participate in the course evaluation.

Academic Integrity: By virtue of enrollment, each student agrees to uphold the high academic standards of the University of Wisconsin-Madison; academic misconduct is behavior that negatively impacts the integrity of the institution. Cheating, fabrication, plagiarism, unauthorized collaboration, and helping others commit these previously listed acts are examples of misconduct which may result in disciplinary action. Examples of disciplinary action include, but is not limited to, failure on the assignment/course, written reprimand, disciplinary probation, suspension, or expulsion.

Accommodations for Students with Disabilities: The University of Wisconsin-Madison supports the right of all enrolled students to a full and equal educational opportunity. The Americans with Disabilities Act (ADA), Wisconsin State Statute (36.12), and UW-Madison policy (Faculty Document 1071) require that students with disabilities be reasonably accommodated in instruction and campus life. Reasonable accommodations for students with disabilities is a shared faculty and student responsibility. Students are expected to inform faculty [instructors] of their need for instructional accommodations by the end of the third week of the semester, or as soon as possible after a disability has been incurred or recognized. Faculty will work either directly with the student [you] or in coordination with the McBurney Center to identify and provide reasonable instructional accommodations. Disability information, including instructional accommodations as part of a student's educational record, is confidential and protected under FERPA.

Diversity and Inclusion: Diversity is a source of strength, creativity, and innovation for UW-Madison. We value the contributions of each person and respect the profound ways their identity, culture, background, experience, status, abilities, and opinion enrich the university community. We commit ourselves to the pursuit of excellence in teaching, research, outreach, and diversity as inextricably linked goals. The University of Wisconsin-Madison fulfills its public mission by creating a welcoming and inclusive community for people from every background – people who as students, faculty, and staff serve Wisconsin and the world.