CSE 312: Foundations of Computing II

Summer 2020

Instructor:	Alex Tsun	**Time:**	MWF 12pm – 1pm PST
Email:	alextsun@cs.washington.edu	**Location:**	Online!

(Syllabus Last Updated: 2020-06-16, 23:01:02)

Course Resources:

1. Course Website: `https://courses.cs.washington.edu/courses/cse312/20su`
2. Gradescope, Edstem, Zoom Links, Calendar, and Lecture Videos all linked from the website above.

Announcement: You should regularly check the class web site for announcements and other information, including the most up-to-date information on problem sets and errata. The class web page will also have the schedule of topics to be covered and links to other class materials, including class handouts. If you have any personal questions, please email me directly. Any other non-sensitive questions about course content should be posted on our discussion forum.

Office Hours: See calendar on website.

Textbooks (Optional):

- Larsen and Marx, *An Introduction to Mathematical Statistics* (5th edition). Prentice-Hall.
- Dimitri P. Bertsekas and John N. Tsitsiklis, *Introduction to Probability*, First Edition, Athena Scientific, 2000. Available online here.
- Sheldon Ross, *A First Course in Probability* (10th Ed.), Pearson Prentice Hall, 2018.

Prerequisites: CSE 311 and MATH 126. Here is a quick rundown of some of the mathematical tools we'll be using in this class: calculus (integration and differentiation), linear algebra (basic operations on vectors and matrices), an understanding of the basics of set theory (subsets, complements, unions, intersections, cardinality, etc.), and familiarity with basic proof techniques (including induction).

Why is CSE 312 Important?

While the initial foundations of computer science began in the world of discrete mathematics (after all, modern computers are digital in nature), recent years have seen a surge in the use of probability as a tool for the analysis and development of new algorithms and systems. As a result, it is becoming increasingly important for budding computer scientists to understand probability theory, both to provide new perspectives on existing ideas and to help further advance the field in new ways.

Probability is used in a number of contexts, including analyzing the likelihood that various events will happen, better understanding the performance of algorithms (which are increasingly making use of randomness), or modeling the behavior of systems that exist in asynchronous environments ruled by uncertainty (such as requests being made to a web server). Probability provides a rich set of tools for modeling such phenomena and allowing for precise mathematical statements to be made about the performance of an algorithm or a system in such situations.

Furthermore, computers are increasingly often being used as data analysis tools to glean insights from the enormous amounts of data being gathered in a variety of fields; you've no doubt heard the phrase "big data" referring to this phenomenon. Probability theory is now used as one of the primary methods for designing new algorithms to model such data, allowing, for example, a computer to make predictions about new or uncertain events. In fact, many of you have already been the users of such techniques. For example, most email systems now employ automated spam detection and filtering. Methods for being able to automatically infer whether or not an email message is spam are frequently rooted in probabilistic methods. Similarly, if you have ever seen online product recommendation (e.g., "customers who bought X are also likely to buy Y"), you've seen yet another application of probability in computer science. Even more subtly, answering detailed questions like how many buckets you should have in your a hash table or how many machines you should deploy in a data center (server farm) for an online application make use of probabilistic techniques to give precise formulations based on testable assumptions.

Our goal in this course is to build foundational skills and give you experience in the following areas:

1. **Understanding the combinatorial nature of problems**: Many real problems are based on understanding the multitude of possible outcomes that may occur, and determining which of those outcomes satisfy some criteria we care about. Such understanding is important both for determining how likely an outcome is, but also for understanding what factors may affect the outcome (and which of those may be in our control).

2. **Working knowledge of probability theory**: Having a solid knowledge of probability theory is essential for computer scientists today. Such knowledge includes theoretical fundamentals as well as an appreciation for how that theory can be successfully applied in practice. We hope to impart both these concepts in this class.

3. **Appreciation for probabilistic statements**: In the world around us, probabilistic statements are often made, but are easily misunderstood. For example, when a candidate in an election is said to have a 53% likelihood of winning does this mean that the candidate is likely to get 53% of the vote, or that that if 100 elections were held today, the candidate would win 53% of them? Understanding the difference between these statements requires an understanding of the model in the underlying probabilistic analysis.

4. **Applications**: We are not studying probability theory simply for the joy of drawing summation symbols (okay, maybe some people are, but that's not what we're really targeting in this class), but rather because there are a wide variety of applications where probability allows us to solve problems that might otherwise be out of reach (or would be solved more poorly without the tools that probability can bring to bear). We'll look at examples of such applications throughout the class.

5. **An introduction to machine learning and theoretical computer science**: Machine learning is a quickly growing subfield of artificial intelligence which has grown to impact many applications in computing. It focuses on analyzing large quantities of data to build models that can then be harnessed in real problems, such as filtering email, improving web search, understanding computer system performance, predicting financial markets, or analyzing DNA. The use of randomized algorithms and probabilistic data structures has also become prevalent, as they are usually have more elegant implementations than their deterministic counterparts, and have more efficient time and/or space complexity.

Goals for Summer 2020:
We fully recognize that this experience cannot replace what we normally have on campus, and that many of you have personal situations that may change throughout the quarter. That being said, we are determined to reach the following course goals to the best of our ability: (1) To maintain the intellectual rigor of the CSE 312 curriculum while providing flexible ways for you to learn, and (2) To foster and maintain human

connections and a sense of community throughout this online course. We have adjusted the typical course structure to meet these goals.

Many of these changes that we are making are designed to help foster intellectual nourishment, social connection, and personal accommodation—through accessible, asynchronous content for diverse access, time zones, and contexts, and optional, synchronous discussion to learn together and combat isolation. Please bear in mind that all of us are experiencing remote, online classrooms at this scale for the first time, and we may have to adapt throughout the quarter. Everyone needs support and understanding in this unprecedented moment, and we are here to listen to you. Thanks and welcome to CSE 312. (Credit for this wording goes to Brandon Bayne from UNC - Chapel Hill.)

Tentative Course Outline:

1. Combinatorial Theory
2. Discrete Probability
3. Discrete Random Variables
4. Continuous Random Variables
5. Multiple Random Variables
6. Concentration Inequalities
7. Statistical Estimation
8. Statistical Inference

Lectures:
We will be holding live lectures remotely via Zoom on Mondays, Wednesdays, and Fridays, 12:00PM – 1:00PM Pacific Time. Lecture content will be covered in two parts:

1. **Asynchronous recorded videos pre-lecture**. We will be pre-recording about 5-10 minutes worth of content in short videos to cover introductory material each lecture. You should watch and be familiar with this material prior to attending live lecture. To help prepare you for effective participation in live lecture, we will have "concept checks," which are required self-tests on the pre-recorded content that will count towards your grade. To encourage you to stay on track, 15% of your course grade will come from completing concept checks. Read more about concept checks below.

2. **Synchronous, live lecture**. The 60-minute live lecture is dedicated to interactive problem solving and advanced concept learning. We'd love for you to ask questions and work together in small groups. We realize that many of you have various constraints on your schedule due to timezone differences, other academic commitments, or personal circumstances, and so all live lectures will be recorded and posted shortly after lecture (there may be a delay of up to 24 hours). To encourage discussion even after lecture has ended, we will be using our EdStem discussion forum (not Zoom chat) to manage questions during lecture.

Grading Policy:

Problem Sets (5)	55%
Concept Checks	15%
Final Project	30%

Problem Sets

- There will be 5 problem sets, equally weighted. There will be a written part and a coding part involved.
- The written parts must be typed using LaTeX, and submitted to Gradescope. A tutorial will be provided and recorded, and there will demonstrated use of these tools in lecture. If you take further math-based classes (such as CSE 446) or plan to write research papers, you will need to start typesetting anyway.

- You **must** show your work; at a minimum of 1-2 sentences per question, but ideally as much as you would need to explain to a fellow classmate who hadn't solved the problem before. Be concise. A correct answer with no work is worth nothing, less than a wrong answer with some work.
- You **must** tag the question parts of your homework correctly on Gradescope. Failure to do so will **result in a 0** on **every** untagged question. Please check your submission by clicking each question, and making sure your solution appears there.
- The coding parts will be written in Python3, with no exceptions. This because the coding parts will be autograded. There are no hidden tests, and you'll have unlimited attempts. Whatever you see last on Gradescope for that section will be your grade.
- Regrade requests are due on Gradescope within **one week** of grades being published.
- It is okay to collaborate in coming up with solutions, but you must list all your collaborators at the top of each homework.
- For some particularly challenging psets, we will allow groups of up to 2. The coding portions of each pset are always to be done individually. The exact psets for group and individual work are yet to be determined. You cannot have the same partner more than once. We will occasionally offer social sessions and surveys for finding partners.

Concept Checks

- Before each lecture, check the calendar to see which video(s) to watch beforehand.
- Watch the video(s).
- Concept checks are due at 11:45am PST on Gradescope on the day of lecture.
- You can submit your answers as many times as you want; we will only grade the final submission. Correct answers will reveal the answer explanation; all other answers will not, so you can keep trying until you see the answer explanation.
- After the initial deadline, you can submit your concept check up to a week later for partial credit (capped at 50%).

Final Project

- As you'll have plenty of practice during our flipped classroom lectures, sections, and psets to practice problem-solving, there will be no exam.
- You will work in groups of 1-3 to write up a project report (in $\LaTeX$), where you apply concepts from CSE 312 to real problems.
- You will investigate a real-world problem, and one or more solutions to that problem (depending on complexity). Ideas include: randomized algorithms, machine learning, stochastic processes, and hypothesis testing/bandits.
- A proposal, milestone, and final report will be collected, and no late days are allowed. See the calendar for the project timeline. More details will be provided later.

Late Policy:

- Problem Sets 1-4

1. On-time problem sets will be given 5% extra credit.
2. Each problem set will be accepted up to two days late without penalty.

- Problem Set 5 will **not** be accepted late.
- Concept Checks can be submitted up to one week late for up to half-credit.
- No project-based submissions (proposal, milestone, and final report) are accepted late.

Attendance Policy: Regular attendance to lecture and section is essential and expected. This class is fast-paced, but we will spend the majority of lecture time solving problems and diving deeper into concepts. This is not a part of your grade; but the problem sets will be challenging since lecture is spent solving a lot of problems. To incentivize attendance, some problem set questions will be explicitly solved **during lectures or sections**. Furthermore, the sections will be devoted to heavily guiding you in the coding portions of the psets.

Academic Honesty: Lack of knowledge of the academic honesty policy is not a reasonable explanation for a violation. Each student is expected to do their own work on the problem sets in CSE 312. Students may discuss problem sets with each other as well as the course staff. Any discussion of problem set questions with others should be noted on a student's final write-up of the problem set answers. Each student must turn in their own write-up of the problem set solutions. Excessive collaboration (i.e., beyond discussing problem set questions) can result in honor code violations. Questions regarding acceptable collaboration should be directed to the class instructor prior to the collaboration. It is a violation of the honor code to copy problem set solutions from others, or to copy or derive them from solutions found online or in textbooks, previous instances of this course, or other courses covering the same topics (e.g., STAT 394/5 or probability courses at other schools). Copying of solutions from students who previously took this or a similar course is also a violation of the honor code. Finally, a good point to keep in mind is that you must be able to explain and/or re-derive anything that you submit.

Violations of the above or any other issue of academic integrity are taken very seriously, and may be referred to the University. Please refer to the Allen School's Academic Misconduct webpage for a detailed description of what is allowable and what is not.

Accommodations:

- **Disability Accomodation Policy**: See here for the current policy.
- **Religious Accommodation Policy**: See here for the current policy.

Acknowledgements: Syllabus wording largely influenced by Lisa Yan, Chris Piech, and Mehran Sahami from Stanford University's CS 109.