CSE 312: Foundations of Computing II

Winter 2024

Instructor:	Anna R. Karlin	**Time:**	MWF 1:30-2:20pm PST
Email:	karlin@cs.washington.edu	**Location:**	Guggenheim 220

Course Resources:

1. Course Website: https://courses.cs.washington.edu/courses/cse312/24wi
2. Gradescope, Edstem, Calendar, and other materials 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 accompanying lecture notes, slides (after lecture) etc. If you have any personal questions, please email me directly. Any other non-sensitive questions about course content should be posted on Edstem (our discussion forum).

TAs and Office Hours: See website.

Textbooks and resources

We will be using three primary resources this quarter:

- Probability and Statistics with Applications to Computing by Alex Tsun
- Lecture notes from CS 70 at Berkeley
- Introduction to Probability for Computing, by Mor Harchol-Balter

The above resources are all freely available and should be all you need.

Nonetheless, you may find following optional textbooks useful.

- 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 and statistics as a tool for the analysis and development of new algorithms and systems, and as the workhorse of modern machine learning As a result, it is becoming increasingly important for budding computer scientists to understand probability theory and statistics, 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 and statistics are the foundational methods used 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 an 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 and some of the key results in statistics**: Having a solid knowledge of probability theory and statistics 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 in machine learning and theoretical computer science**: 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 and statistics allow us to solve problems that might otherwise be out of reach (or would be solved more poorly without the tools that probability and statistics can bring to bear). We'll look at examples of such applications throughout the class. For example, 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 generating natural language, filtering email, improving web search, understanding computer system performance, predicting financial markets, or analyzing DNA. You've also probably heard of large language models (LLMs) such as chat GPT. Probability and statistics form the foundation of all of these systems. Another example application area is the use of randomized algorithms and probabilistic data structures. These usually have simpler

and more elegant implementations than their deterministic counterparts, and have more efficient time and/or space complexity. We will be learning about some of these applications and you will have the opportunity to implement some of these algorithms.

Goals for CSE 312:

As always, 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 course.

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
9. Applications in machine learning and theoretical computer science

Lectures:

The lectures will be recorded on Panopto, and you will be able to access those recordings within a few hours after class on Canvas.

In addition, linked from the website are video recordings that Alex Tsun made that accompany his textbook. You may find those helpful. They cover the same material that we cover in class, though sometimes in class we will use different examples to illustrate the same concept.

Approximate Grading Breakdown: I may end up shifting any of these by up to 5%.

Problem Sets (8)	35%
Concept Checks	15%
Midterm	15%
Final	35%

I will end up grading on a curve, setting the median at around 3.5, but in order to get a grade of 3.9 or above, you will need to have an overall high percentage, above around 93% (exact percentage to be determined).

Problem Sets

- There will be 8 problem sets. All of them will involve a written part and many of them will involve a coding problem or two as well. You will be submitting your homeworks on Gradescope.
- The coding you do on the problem sets will be done in Python. The implementation you do will provide you with a deeper understanding of how the theory we learn in this class is used in practice and should be a lot of fun. Note that we do not expect you to have any experience or knowledge of Python – we will provide you with tutorials and other kinds of help to get you started. A huge bonus of this class will be that you will come away with basic, working knowledge of Python (which you will undoubtedly use in the future, and definitely if you take CSE 446, the machine learning class).
- We strongly encourage you to type the written parts up using LaTeX. There are links to resources for learning LaTeX on the website. If you take other classes that involve a fair amount of math (such as the machine learning class CSE 446) or plan to write research papers, you will need to typeset in LaTeX anyway. It is a very useful skill, so you may as well start now.

- You **must** show your work; at a minimum 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. Use the section solutions we provide as a guide for the level of detail we are seeking.
- 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. We recommend starting each problem on a new page to keep this simple.
- The coding parts of the homeworks 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. You will be able to write and test them on the edstem platform, which means that essentially no setup is required.
- Regrade requests are due on Gradescope within **one week** of grades being published. No late regrade requests will be accepted.
- For some homeworks, we will allow groups of up to 2 to work together and submit a single homework. Regardless, it is okay to brainstorm and collaborate with others in coming up with solutions, but you must list all your collaborators at the top of each homework. On solo homeworks, you should write your solutions up *entirely on your own* and on homeworks done in a pair, the pair should write up their solutions *together*.
- Although homework is "only" worth 35% of your grade, and we will usually allow you to do it in pairs, it will be **the bulk** of the work in this class. **Importantly, if you do not spend quality time on it and if you do not understand the solutions, you will not have learned what you need to learn in this class. In addition, you will have almost no chance of doing well on the tests if you do not spend quality time on the homework.** You really need to take it seriously if you want to understand the material and do well in the class and we will be there to help you every step of the way.

Section Attendance

- Sections are designed to help reinforce the material and give you practice solving problems similar to those on the homework.
- You are welcome to attend a section other than your own if needed, but to keep things balanced, it is preferable if you attend your assigned section.

Concept Checks

- Associated with almost all lectures, there will be a "concept check" for you to take on Gradescope. This will consist of 4-8 questions that test your basic understanding of the concepts covered in lecture. Occasionally, the concept check will review something you should have learned in a previous class. The questions are intended to be very straightforward to answer; each concept check should not require more than about 20-30 minutes.
- Each concept check will be available within 40 minutes of the end of class and is due 30 minutes before the next 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.

- Concept Checks can **not** be submitted late, but earning 85% in this category leads to 100% in the grade book.
- **Very important:** Please submit each concept check, regardless of how many problems you solve (even if you do none of the problems). If you don't submit them, you will not be able to look at the solutions afterwards.

Late Policy:

- Problem Sets: You have 6 late days during the quarter, but can only use 2 late days on any one problem set. Since homeworks will be due on Wednesdays at 11:59pm, this means that you can use late days to submit any particular homework up until Friday at 11:59pm. Please plan ahead as we will not be willing to add any additional no-penalty late days, except in absolute, verifiable emergencies.
- If you run out of late days, you may still turn in an assignment late at a penalty of 33% per day.
- Concept Checks can not be submitted late, but earning 85% in this category leads to 100% in the grade book.
- If you have extenuating circumstances that interfere with any of the above, please get in touch with the course staff as soon as possible.

Attendance Policy: Regular attendance to lecture and section is strongly recommended. Moreover, I encourage you in the **strongest possible terms** to ask questions during lecture and sections (as well as in office hours and on the discussion board). That will make the class more fun for all and you will definitely learn more and have an easier time with the homework!! Anna genuinely enjoys being interrupted with questions during lectures, as it leads to better learning by everyone! There will also be an edstem thread for each lecture that you can ask follow-up questions on.

Keep in mind that this class is fast-paced, and the problem sets will be challenging. Most of the sections will be devoted to giving you practice on problems similar to those on the psets.

Academic Integrity: 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, with the following caveats:

- Do not take away any notes or screenshots from your discussions with others.
- After discussing with others, take a 30 minute break before writing up your solutions.
- Cite the names of all your collaborators somewhere on your homework.
- On a solo homework, write up your solutions entirely on your own. On a homework done in pairs, the pair should be writing up their solution together, but without any consultation with other pairs.

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, **be sure that you are able to explain and/or re-derive anything that you submit.** If we have doubts about whether you did the work on your own, we may ask you to come in and explain your solution to us verbally.

Violations of the above or any other issue of academic integrity are taken very seriously, and may be referred to the University Disciplinary Board. 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, and Alex Tsun's offering of CSE 312.