ECE36800 Data structures

Course introduction

Lu Su

School of Electrical and Computer Engineering
Purdue University

Slides Courtesy: Prof. Cheng-Kok Koh

Instructor: Lu Su

- **Associate Professor, School of ECE, Purdue University**
- **Ph.D in CS, M.S. in Statistics, University of Illinois at Urbana-Champaign**
- **Research areas: Internet of Things, Cyber-Physical Systems, Wireless, Mobile and Crowd Sensing Systems**

Office	EE 331B
Hours	TBA
Phone	(765) 496-3475
Email	lusu@purdue.edu
Zoom	https://purdue-edu.zoom.us/my/lusuroom

Teaching assistants

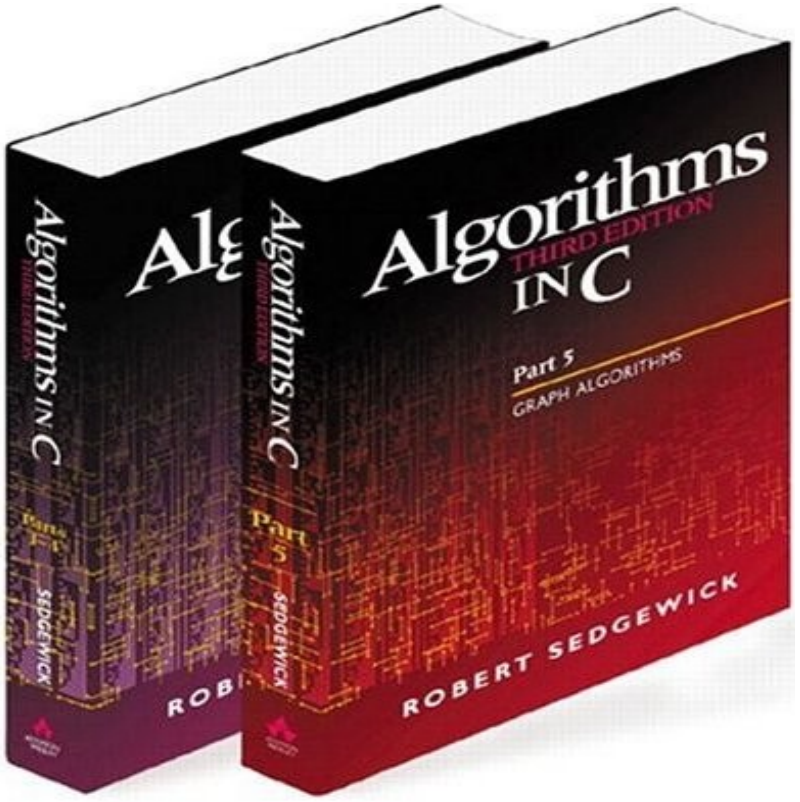
- Please check Brightspace course page for the full list of teaching assistants (graduate and undergraduate)
- Their office hours will be listed on Brightspace course page

Instructor/TA office hours

- The instructors and TAs will maintain a queue (queuplive.com), room key to be determined and announced on Brightspace
- During the office hour, join the queue and enter your Purdue email id
- When the instructor/TA is ready, he/she would invite you to join his/her virtual meeting room
- Please look at the slides about ssh and mounting drive so that you can easily do your work on your personal computer/laptop while storing your work on ECE servers

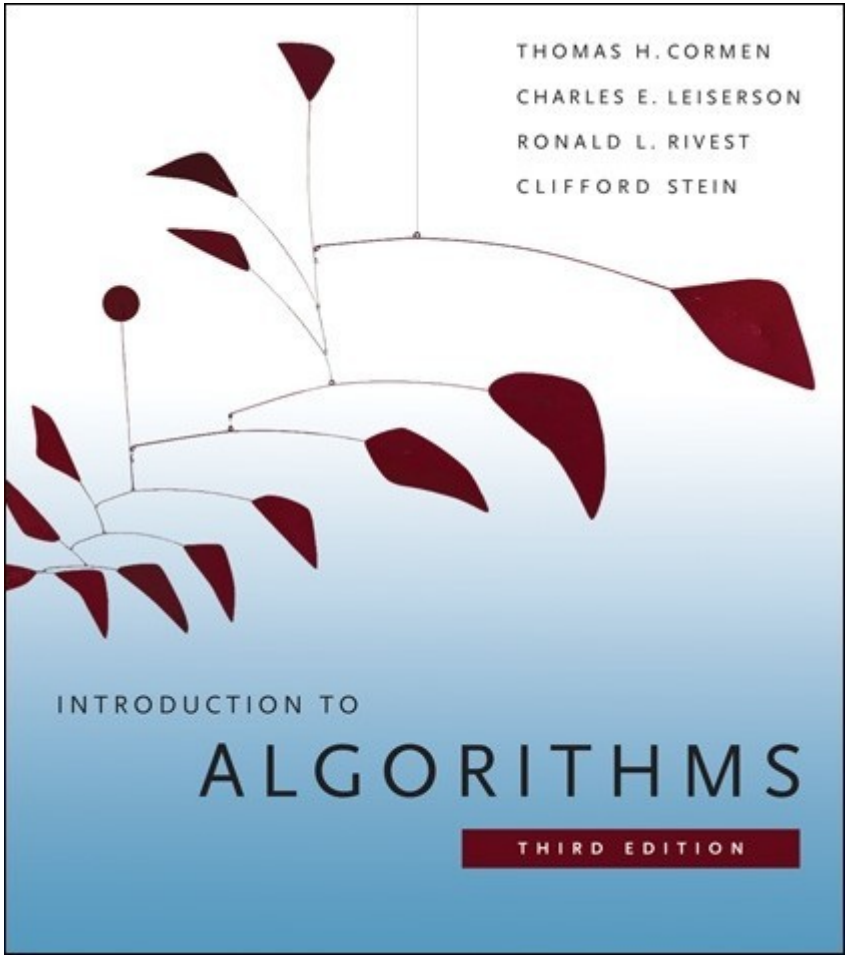
Course Textbook

- Algorithms in C, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching, Third edition, Robert Sedgewick, Addison Wesley, 1999, ISBN-10: 0201314525, ISBN-13: 9780201314526
- Algorithms in C, Part 5: Graph Algorithms, Third edition, Robert Sedgewick, Addison Wesley, 2002, ISBN-10: 0201316633, ISBN-13: 9780201316636



Recommended Reference

- Introduction to Algorithms, Third Edition
- By Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein



Why this course is important

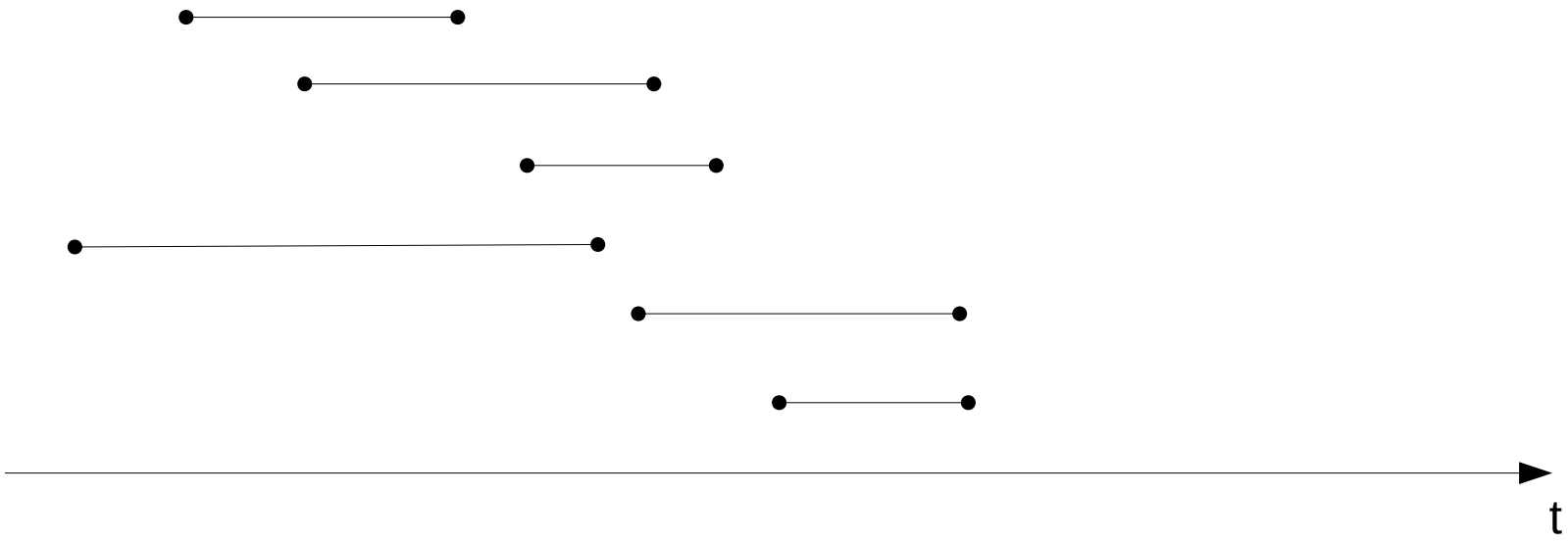
- You should learn stuff that will make you USEFUL to people who might hire you
- You should also learn stuff that will allow you to hire the right people
- Data structures and algorithms are a huge portion of what programmers need to know
  - EVERYTHING a computer does is based on data structures and algorithms
- We'll go through some of the most useful algorithms, and with luck, you'll learn enough so that you can figure out everything else when you need it

A consulting job (adapted from CACM)

- Contracted to help independent cab and limo operators in Tokyo, Japan to schedule all customers who wanted to ride with them during 2021 Summer Olympics
- Requests from customers: “I want a cab and driver from such a start time to such an end time”
  - The duration of a request starts on exactly start time s and ends before end time e
  - We represent such a request by [s, e), with s < e

A consulting job

- Scenario 1: Drivers paid a flat rate per ride
  - Your program should provide a driver the largest possible subset of requests that did not overlap in time

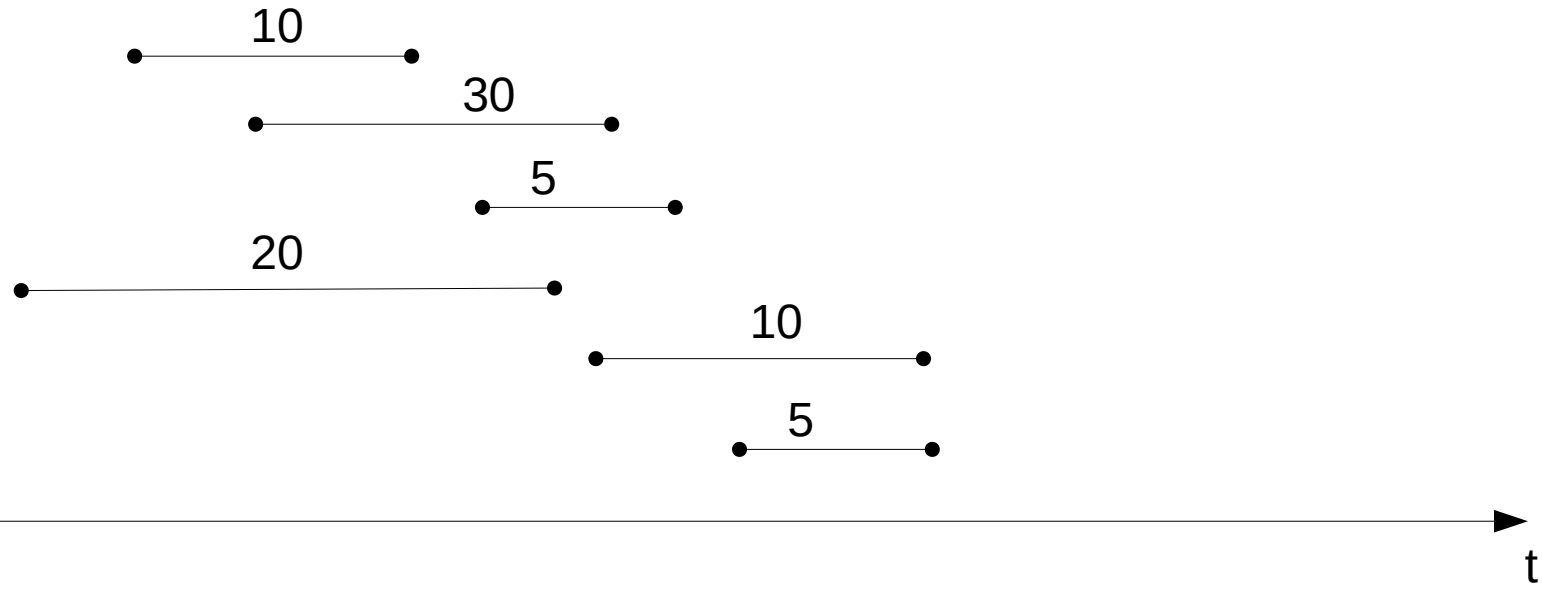


Assumptions/simplifications

- A request should have included a start location for picking up the customer and a destination to drop off the customer
- A cab, after dropping off a customer at destination D, can magically appear at start location S to pick up the next customer as long as the finish time at D ≤ the start time at S
- We do not allow ride sharing, a cab cannot pick up another customer along the way while serving a customer
- Two requests [s, e) and [s', e') can be serviced by a cab driver if [s, e) and [s', e') are non-overlapping
  - e ≤ s' or e' ≤ s

A consulting job

- Scenario 2: Customers bid for how much they were willing to pay for the requested period; opening ceremonies and swimming are more popular than (name your least favorite sport)
  - Your program should schedule the set of non-overlapping requests to maximize the amount of money the driver using the program would earn



A consulting job

- Scenario 3: Some customers might have wanted the same driver for a set of time-period requests
  - Your program should pick the sets of requests to maximize the amount of money the driver would receive without overlapping time
- The first two problems are easy, but the last is intractable

What are data structures?

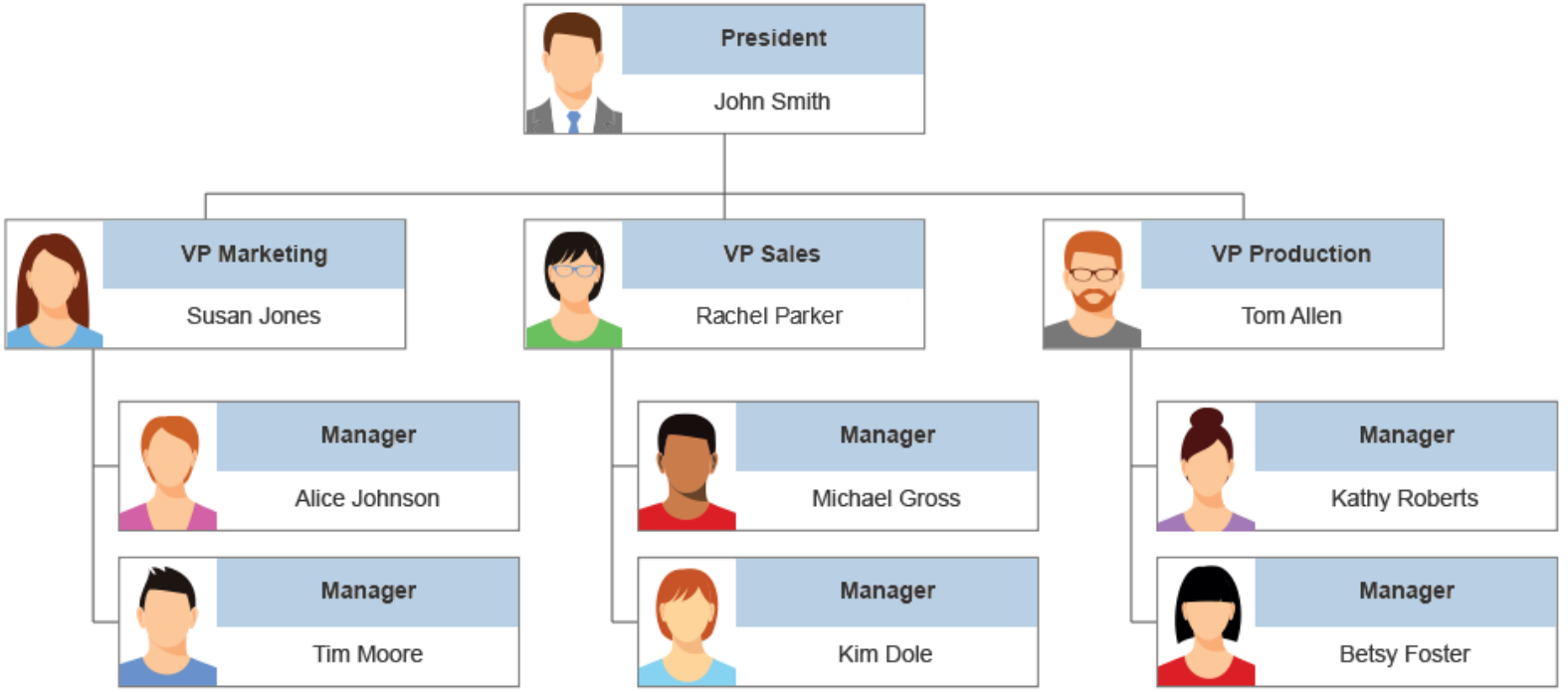
- Organization of information (objects) for ease of manipulation
- Examples:
  - Dictionary
  - Check-out line
  - Spring-loaded plate dis
  - Organizational chart

- Associated with methods (algorithms) for manipulating objects

What are data structures?

- Organization of information (objects) for ease of manipulation
- Examples:
  - Dictionary
  - Check-out line
  - Spring-loaded plate dispenser
  - Organizational chart

- Associated with methods (algorithms) for manipulating objects



  - Organizational chart
- Associated with methods (algorithms) for manipulating objects

What are algorithms?

- A method of doing something
- Examples
  - Algorithm of multiplying two numbers
  - Algorithm of making peanut butter and jelly French toast
  - Algorithm of getting dressed in the morning
- We're interested in
  - Does it do the job correctly?
  - Is it as efficient as it can be, in terms of time and/or (memory) space?

What you would learn

- Advanced programming ideas, in practice and in theory
- Data structures and their abstractions
  - Stacks, lists, trees, and graphs
- Fundamentals of algorithms and their complexities
  - Sorting, searching, hashing, and graph algorithms
- Problem solving

At the end of semester ...

- All of you can find an efficient solution to the first problem and the second problem
- Some of you will go into grad school to find out why it is unlikely to find an efficient solution to the last problem
- Or you find an efficient solution to the NP-hard problem and be rich and famous ☺ (see http://www.claymath.org/millennium-problems/p-vs-np-problem
  for a related problem, see also http://www.claymath.org/sites/default/files/minesweeper.pdf
  about the connection between minesweeper and P=NP?)

You should know how to … (prerequisite: ECE 26400)

- Write and compile a C program in UNIX
- Dynamically manage memory
- Manipulate arrays, pointers, and linked-lists
- Build a tree and traverse it
- Create complex data types
- Access various forms of I/O

Learning objectives

A student who successfully fulfills the course requirements will have demonstrated:

1) An understanding of basic data structures, including stacks, queues, and trees
2) An ability to analyze time complexity and space complexity of algorithms
3) An ability to apply appropriate sorting and searching algorithms for a given application
4) An ability to apply graph theoretic techniques, data structures, and algorithms for problem solving
5) An ability to design and implement appropriate data structures and algorithms for engineering applications

Learning objective assessment

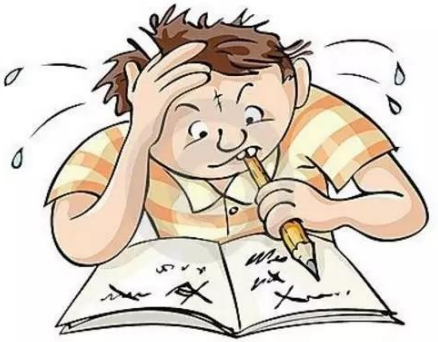
- Students must meet ALL learning objectives to receive a passing grade; “F” otherwise
- Meeting all objectives is a necessary (but not sufficient) condition to receive a passing grade
- At least TWO opportunities will be provided:
  - Non-programming homework exercises (HW)
  - Programming assignments (PA)
  - Midterm and final exams (EX)
- You meet a learning objective if you score 50% or more on ONE of the evaluation instruments

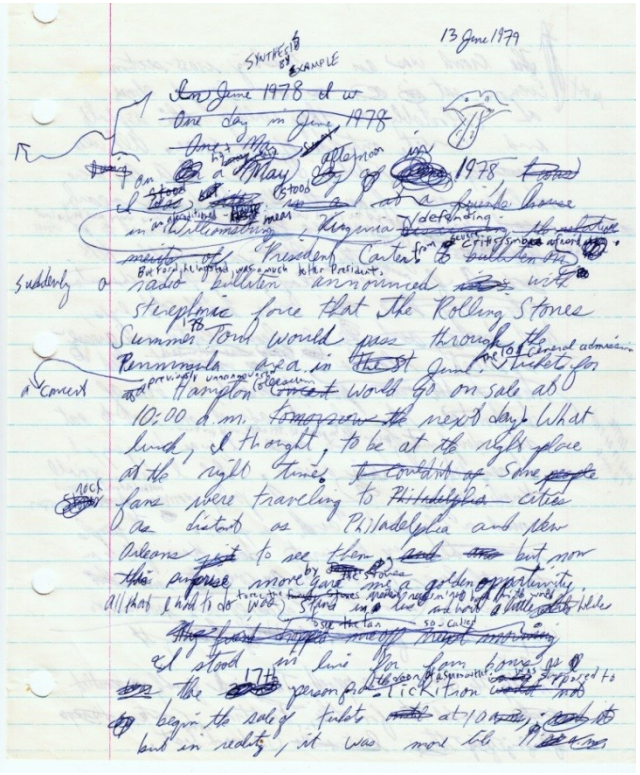
Non-programming homework exercises (HW)

- 10 (±2) non-programming homework exercises
- Homework exercises will be released on Brightspace and you turn in your homework through Gradescope by 11:59pm on the due date
- No late submission is accepted
- All non-programming homework exercises carry the same weight

Handwritten work and legibility

- Written assignments (and exams!) should be neat and legible.
- Instructor and TAs reserve the right to assign a zero grade to illegible handwritten homeworks/exams.





Programming assignments (PA)

- 3 programming assignments
- Work on your own
- Programming component to be developed on Linux and submitted electronically (through Brightspace) by 11:59pm on the due date
- Do not wait until the last minute
- No late submission is accepted
- All programming assignments carry the same weight

Code evaluation

- Code will be evaluated on a Linux platform (ecegrid)
- Important criteria: logic correctness, coding efficiency
- If your code leaks memory or has memory errors, we will deduct 50%
- The code should be organized, clear, and well commented
  - If we find your code to be disorganized, unclear, or not well documented, we may deduct up to 50%

ssh and mounting drive

- Use ssh (secure shell) to access ecegrid directly
  - You can use ecegrid-thin1.ecn.purdue.edu (also ecegrid.ecn.purdue.edu), ecegrid-thin2.ecn.purdue.edu, ... , ecegrid-thin7.ecn.purdue.edu
  - You can use the command “ssh userid@ecegrid-thin7.ecn.purdue.edu“, where userid is your login name
- Access your files on the ECE server directly through your computer
  - Use ssh to log in to any of the server and then type "pwd" to find out the path of your home directory. It is typically of the form "/home/shay/a/userid" where "shay" is the server hosting your directory, "a" is the disk partition (I think), and "userid" is your user id.
  - Follow the appropriate instructions in the following links to map the network drive. For linux, use the "connect to" feature of the "file" app to map the drive: "sftp://ecegrid.ecn.purdue.edu/home/server/a/userid".
  - For windows computers, follow the following instructions:
    - https://engineering.purdue.edu/ECN/Support/KB/Docs/MappingECNDriveWin8
    - https://engineering.purdue.edu/ECN/Support/KB/Docs/MappingECNDriveWin7
    - https://engineering.purdue.edu/ECN/Support/KB/Docs/MappingECNDriveWinXP
  - For mac users:
    - https://engineering.purdue.edu/ECN/Support/KB/Docs/MacOSXConnectingToSMB

Mid-term and final exams (Tentative)

- Open-book, open-notes
- Means to meet learning objectives
- Exam 1, 6th week, 1 hour (either in class or evening)
- Exam 2, 12th week, 1 hour (either in class or evening)
- Final exam (Exam 3), 2 hours, TBA
- No make-up exams (If conflict, notify me as soon as possible)

Academic honesty

- We expect every member of the Purdue community to practice honorable and ethical behavior both inside and outside the classroom
- Any actions that might unfairly improve your grade will be considered cheating
  - If you received help for your exercise or assignment, acknowledge it – but do your own work!
  - All programming assignments are subject to computer-based comparison and analysis
- Cheating on an exercise/assignment/exam will result in a zero for the exercise/assignment/exam
  - It will be reported to the Dean of Students Office
  - It could result in a failing grade for the course, at the discretion of instructor
- The instructors own most of the course materials (lecture slides, homework exercises, programming assignment, exams).
  - You are not allowed to upload or share any of these materials with commercial websites such as Course Hero, Chegg, or Quizlet.

Grade determination (tentative)

Non-programming homework exercises (HW)	15%
Programming assignments (PA)	45%
Mid-term exams (EX)	2 X 10% = 20%
Final exam (EX)	20%

Success in ECE 36800 (and ECE)

- While attendance is not taken in this class, it usually is essential to your success.
- It is your responsibility to make sure you obtain any material, handouts, announcements, information, etc. that you missed
- Put in consistent effort
  - Keep up with the lecture materials everyday, clear your doubts before the next lecture
  - Read the homework and assignments as soon as they are released
  - Do small chunk of work on homework and assignments everyday
- If you come with a willingness to work hard and learn, I will work with you to help you achieve your goals

Tentative outline of the semester

- Asymptotic complexity
- Simple sorting algorithms
- Stacks, Queues, Linked Lists
- Trees and Binary Search Trees
- Advanced Sorting algorithms
- Graphs and Associated Algorithms
- Multiway Search Trees
- Hash tables

COVID-19 Policy

- Effective Monday, Aug. 2, face masks will again be required in all indoor spaces for everyone on the Purdue University campus, including students, employees and visitors, regardless of vaccination status.
- Students arriving from international locations should reference Purdue’s Summer and Fall 2021 Arriving from Outside the U.S. protocols.
- Complete the Health and Safety training, update your address while enrolled, and provide Purdue with a primary student phone number you can be reached for contact tracing purposes.

COVID-19 Policy

- Reinforcing the need for all Purdue students to take personal responsibility to protect themselves, others and the community, the key to upholding the Protect Purdue Pledge going forward will be choosing to be vaccinated and submitting valid proof by August 13 OR taking part in routine surveillance testing, which will begin on August 23 and could be as frequent as weekly.
- Visit website https://protect.purdue.edu/ for more details on COVID-19 policy

In case of campus emergency

- Course requirements, deadlines and grading percentages are subject to changes that may be necessitated by a revised semester calendar or other circumstances
- Any course changes will be posted on Brightspace
- Visit website https://www.purdue.edu/ehps/emergency_preparedness/ for more details on how to respond in an emergency