Idaho State University | Computer Science

Data Structures and Algorithms

CS 2235 – Fall 2019

Professor: Isaac Griffith

Office: BA 315
Phone: (208) 282-4876
Email: grifisaa@isu.edu
URL: https://www2.cose.isu.edu/~grifisaa/

Office Hours:

- **TR**: 1430 – 1530
- By appointment scheduled at: https://isaac-griffith.youcanbook.me/

Course Information

Meeting Time: TR: 09:30 – 10:45
Room: Pocatello: LIB 16, Idaho Falls: CHE 208

Final Exam Dates

Section: 01 – Pocatello
Date: Thursday Dec 12, 2019
Time: 0730 – 0930
Room: BA 506

Section: 02 – Idaho Falls
Date: Thursday Dec 12, 2019
Time: 0730 – 0930
Room: Idaho Falls Testing Center

Prerequisites:

- CS 1181

Textbooks:

- **Data Structures and Algorithms in Java, 6th Edition** – Goodrich, Tamassia, and Goldwasser.

ISBN: 978-1118771334

Course Description

Implementation, usage, and design concerns of important data structures and their operations. Implementation and discussion of basic search and sorting algorithms. Discussion will include both O(n log n) and linear sorting algorithms. Incorporates aspects of time complexity and asymptotic analysis of algorithms. Students will be required to develop small to medium sized programs.

Technology

This is a Computer Science course and is heavily focused on programming related activities. As a student you are expected to have at your disposal a computer system capable of running a recent operating system such as MacOS X, Windows 10, or a recent Linux. This course is focused on the Java™ language and thus requires you to have installed at a minimum the following tools:

- Java Development Kit version 1.8 or higher.
- A Professional Grade Java IDE such as JetBrains IntelliJ IDEA, Eclipse, or Netbeans with the associated Gradle plugins.
- Gradle Java Dependency Management and Build system.

Learning Outcomes

Object-Oriented Programming

- Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language. [U]

Basic Type System

- Give an example program that does not type-check in a particular language and yet would have no error if run. [F]
- Discuss the differences among generics, subtyping, and overloading. [F]
- Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software. [F]
- For multiple programming languages, identify program properties checked statically and program properties checked dynamically. [U]
- Use types and type-error messages to write and debug programs. [U]
- Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections. [U]

Basic Analysis

- Explain what is meant by “best”, “expected”, and “worst” case behavior of an algorithm. [F]
- In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors. [A]
- State the formal definition of big O. [F]
- List and contrast standard complexity classes. [F]
- Give examples that illustrate time-space trade-offs of algorithms. [F]
- Determine informally the time and space complexity of simple algorithms. [U]
- Perform empirical studies to validate hypotheses about runtime stemming from mathematical analysis. Run algorithms on input of various sizes and compare performance. [A]

Fundamental Data Structures

- Describe common applications for each of the following data structures: stack, queue, priority queue, set, and map. [F]
- Compare and contrast the costs and benefits of dynamic and static data structure implementations. [A]
- Write programs that use each of the following data structures: arrays, records/structs, strings, linked lists, stacks, queues, sets, and maps. [U]
- Compare alternative implementations of data structures with respect to performance. [A]

Fundamental Data Structures and Algorithms

- Implement simple search algorithms and explain the differences in their time complexities. [A]
- Be able to implement common quadratic and O(N log N) sorting algorithms. [U]
- Solve problems using fundamental graph algorithms, including depth-first and breadth-first search. [U]
- Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm. [U]
- Trace and/or implement a string-matching algorithm. [U]
- Describe the implementation of hash tables, including collision avoidance and resolution. [F]
- Discuss the runtime and memory efficiency of principal algorithms for sorting, searching, and hashing. [F]
- Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data. [F]
- Explain how tree balance affects the efficiency of various binary search tree operations. [F]
- Describe the heap property and the use of heaps as an implementation of priority queues. [F]

Algorithms and Design

- Implement a divide-and-conquer algorithm for solving a problem. [U]
- Apply the techniques of decomposition to break a program into smaller pieces. [U]
- Identify the data components and behaviors of multiple abstract data types. [U]
- Implement a coherent abstract data type, with loose coupling between components and behaviors. [U]
- Determine whether a recursive or iterative solution is most appropriate for a problem. [A]

Introduction Modeling and Simulation

- Create a simple, formal mathematical model of a real-world situation and use that model in a simulation. [U]
- Create a simple display of the results of a simulation. [U]
- Explain the concept of modeling and the use of abstraction that allows the use of a machine to solve a problem. [F]

Student Expectations

The above objectives cannot be met unless you, the student, take an active role in your education. Thus you are expected to:

- **Attend class** on a regular basis and devote your attention to the material presented
- Prepare for each and every class by **reading** the assigned material and completing both **pre- and post-lecture assignments**
- Devote the necessary **time** to preparing assignments and turning them in on time.
  - **Computer Science** is time-intensive
  - You should be prepared to give the time need for each assignment
  - As the class progresses the time required for assignments will increase, be prepared.
- **Time Management** is a requirement. Do not procrastinate, as the amount of time required for any given assignment and any given student cannot be estimated. For this reason you are encourage to begin assignments at the earliest possible date so that you will be able to complete them on time.

This is a 3 credit, as a student, you should expect to put forth on average 6 – 9 hours of additional effort outside the classroom towards this course. Given that, I expect to utilize this completely.

Valid Excuses

This course requires that you attend class and participate in classroom activities (including exams). The student handbook notes the following can be considered viable excuses for class absence, in consultation with your instructor:

- Serious Illness
- Severe Weather
- Religious Holidays
- Approved University Activities (e.g., extracurricular athletics, performance groups, student government)
- Emergency Family Issues

Invalid excuses include anything else, but I will specifically note the following:

- Minor Illness
- Typical Winter Weather Conditions
- Work
- Non-emergency Family Issues
- You or your family Purchased Plane Tickets (non-refundable)

Moodle

Course material including lectures, assignment requirements, handouts, and solutions can be viewed using your Moodle account. Announcement and Help forums will also be available on Moodle. Students are expected to access their Moodle account on a daily basis to keep apprised of course developments.

Attendance

Attendance in the course is mandatory. Missing 9 class meetings (without an acceptable excuse as defined by University Policy) during the course will result in failing the course, and a resulting X grade (IAW CoSE policy). Attendance will be monitored by in class activities, quizzes, etc.

Assignments

Homework assignments are due as assigned on Moodle. Do the homework, it helps. Late homework will be accepted up to the last day of the course, but each day after the due date will incur a linearly increasing penalty. This penalty will result in a 0 grade at midnight of the last day of class.

This course is based upon four components:

- Attendance at lectures and participation during in class exercises.
- Reading assignments
- Project
- Exams

Exams

You will be tested on your mastery of topics listed above as taken from lecture, readings, and assignments. Any information addressed by a component of this class will be considered for exams. There will be two exams, a mid-term and a final. The **final** will be a **comprehensive** exam.

Note: I do not give exams on any date other than the assigned date. The only exceptions to this are to accommodate exceptions noted above. These exceptions will only be accommodated if provided in advance and will require a minimum of **1 week advanced notice** unless an emergency.

Note: If you do not earned a **60% or higher average** on the exams in this course the highest grade you will receive in this course is a **60%** which equates to a **D–**

Grade Distribution

Grades for all assignments, exams, and the final grade will be assigned according to the following table:

Grade	–		+
A	90.00 – 92.99	93.00 – 100.0	
B	80.00 – 82.99	83.00 – 86.99	87.00 – 89.99
C	70.00 – 72.99	73.00 – 76.99	77.00 – 79.99
D	60.00 – 62.99	63.00 – 66.99	67.00 – 69.99
F		00.00 – 59.99	

The final grade calculation for this course will be allocated as follows:

Grade Event Type	Percent of Final Grade
Programming Assignments	40%
Programming Projects	20%
Midterm Exam	15%
Final Exam	25%

Learning Environment

We are all committed to maintaining an inoffensive, non-threatening learning environment for every student. Class members (including the instructor) are thus to treat each other politely–both in word and deed. Offensive humor and aggressive personal advances are specifically forbidden. If you feel uncomfortable with a personal interaction in class, see your instructor for help in solving the problem.

Content

I reserve the right to change the content as needed to fit the flow of the class and experience of the students. Changes will be reflected on Moodle.

Policies & Procedures

Academic Integrity

Academic Integrity is expected at Idaho State University and the College of Science and Engineering. All forms of academic dishonesty, including cheating and plagiarism, are strictly prohibited, the penalties for which range up to permanent expulsion from the university with "Expulsion for Academic Dishonesty" noted on the student's transcript.

Academic Integrity violations are a scourge at any University. I detest them with a passion. Anyone found to be violating the academic integrity code on any assignment or exam will be dealt with with extreme prejudice. The standard remedy, in this course, **for any Academic Integrity Violation will be FAILURE of the course**. Please note that in accordance with the Policy at Idaho State University that attempts to withdraw from the course to avoid such punishment will work to no avail.

Academic dishonesty includes, but is not limited to:

1. Cheating on Exams
2. Plagiarism
3. Collusion
4. Sharing solutions or code on programming assignments

Definitions

Cheating on an examination include:

- Copying from another's exam, any means of communication with another during an exam, giving aid to or receiving aid from another during an exam;
- Using any material during an examination that is unauthorized by the proctor;
- Taking or attempting to take an exam for another student or allowing another student to take or attempt to take an exam for oneself
- Using, obtaining, or attempting to obtain by any means the whole or any part of an administered exam.
- Talking to anyone other than the professor during an exam.

Plagiarism is the unacknowledged incorporation of another student's work into work which the student offers for credit.

- The use of the source code of another person's program, even temporarily, is considered plagiarism. This includes attempting to hide the plagiarism by changing variable names, method names, or class names.
- Copying material from another project (including open source projects) without attributing (citing) that project.

Collusion is the unauthorized collaboration of another in preparing work that a student offers for credit.

- Allowing another person to use your source code, even temporarily, is considered collusion.

Other types of academic dishonesty include:

- Using other student's content from their assignments, disk, etc.
- Performing any act designed to give unfair advantage to a student or the attempt to commit such acts

Exceptions

In this course, the specific exceptions given below are not considered scholastically dishonest acts:

- Discussion of the algorithm and general programming techniques used to solve a problem.
- Giving and receiving aid in debugging.
- Discussion and comparison of program output **(output only not code)**

Student Notification

All students are responsible for checking Moodle and their email on a regular basis, preferably daily, for notification of any class scheduling changes or assignment clarification. Often such notifications

will be posted late at night.

Instructor Availability

The instructor will be available during posted office hours, but additional efforts are made to increase accessibility to the students. If the instructor is not available at the telephone number above, the student can leave a detailed voicemail message. However, the instructor's email is checked at a minimum of twice a day and often the student will get an immediate response to questions submitted by email. Email is usually the most reliable means of contact.

Note that I am a working research scientist, thus I may need to attend conferences both with my local colleagues and with my international colleagues. Thus, I am constantly in meetings, both here and abroad. That being said, I work 7 days a week 12 months a year. If you need my help and all other means of scheduling have led to no avail, please do not hesitate to contact me. Note that I am not an emergency service, I will require that you contact me 24 hours prior to an assignment being due, for which you need help. If you do not plan ahead, I cannot help.

Email Etiquette

Email is the best possible method of reaching me outside of my office hours. Note that I have certain expectations for communicating with me via email, as follows:

- DO NOT use chat or SMS shorthand in your messages.
- Use full words, and full sentences.
- Maintain the frame of mind that your are communicating to a professional, and that a professional demeanor is required.

Failure to abide by these requirements will result in your email being deleted and forgotten.

Disability and Special Needs

The Computer Science program at Idaho State University is committed to ensuring that all students achieve their potential. If you have a disability (physical, hearing, vision, psychiatric, or learning disability) that may need a reasonable accommodation, please contact the ADA & Disabilities Resource Center located in the Rendezvous Complex, Room 125, 282-3599, as early as possible.

Closed Week Policy

Information about the ISU Closed Week Policy can be found online. Note that the policy does not prevent the presentation of new material during closed week.

CoSE X Grade Policy

In the College of Science and Engineering, a student who earns a failing grade via course work (exams, homework, etc.) and has unexcused absences that total more than 30% of class meetings will receive a grade of "X".