

CS 2402 Data Structures

Spring 2009

1. General Information

Instructor

Olac Fuentes

ofuentes@utep.edu

www.cs.utep.edu/ofuentes

(915) 747-6956

Office hours: Tuesdays and Thursdays 11:00-12:00, or by appointment, in CSB 208 (feel free to drop by at other times if I'm there).

Teaching Assistants (TAs):

Patricia Reyes

preyes5@miners.utep.edu,

Office hours: Mondays and Wednesdays 12:30-1:30, Tuesdays and Thursdays 11:00-12:00, or by appointment, in CSB 304.

Christian Servín

christians@miners.utep.edu

Office hours: Mondays 4:00-5:00, Tuesdays 11:00-12:00, or by appointment, in CSB 304.

Peer Leaders (PLs):

- To be announced

Lectures

- MWF 10:30-11:20, Room CS 321

Labs

- MWF 11:30-12:20, Room CS 300. TA: Christian Servín
- TR 12:00-13:20, Room CS 301. TA: Patricia Reyes

Class web site:

http://www.cs.utep.edu/ofuentes/cs2402_Spring09.html

2. Objectives and Outcomes

This is the third and final course in the fundamental computer science sequence. Students will learn about fundamental data structures and analysis and design of algorithms.

Level 3 Outcomes: Synthesis and Evaluation:

Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to:

- specify data structures and operations associated with abstract data types
- define the signature and pre- and post-conditions for operations of an abstract data type
- given a scenario, describe the abstract data types that could be created
- implement binary and binary search trees, using pre-, post-, or in-order traversals as appropriate for a given situation
- judge which data model (list, tree, graph, or set) is appropriate for solving a problem
- justify the choice of a data structure to solve a problem based on issues such as time, and space, of the data structure
- judge which implementations are best suited for an application that requires a list data model: lists, circular lists, circular queue, or generalized list
- judge whether an array or linked implementation is best suited for an application that requires a data model
- judge which graph representations (adjacency list, adjacency matrix, edge list) are appropriate for solving a problem
- judge which sort algorithm (insertion, selection, mergesort, heapsort, quicksort, radix) is appropriate for solving a problem
- judge which search algorithm and data structure is appropriate for solving a problem
- implement a recursive solution to a problem

Level 2: Application and Analysis:

Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able to:

- categorize algorithms based on programming strategy, i.e., divide-and-conquer, greedy, backtracking, and dynamic programming strategies
- analyze iterative and recursive algorithms with respect to time and space
- describe the applications for a dictionary/map ADT, e.g., the application of a symbol table
- give representations for and operations on a binary tree, general tree, threaded tree, heap, binary search tree, B-tree, quadtree, and graphs
- determine the order for a B-tree based on memory issues
- apply graph algorithms for determining shortest paths (Dijkstra's and Floyd's algorithms), minimal spanning tree (Prim's and Kruskal's algorithms), transitive closure (Floyd's algorithm), and topological sort

- select an appropriate sorting algorithm for a given situation and defend the selection
- explain differences and similarities among approaches for resolving collisions in hash tables, e.g., linear probing, quadratic probing, double hashing, rehashing, chaining
- apply design methods and other problem-solving strategies. Examples might include (but are not limited to) functional decomposition, design patterns, top-down design, abstraction, CRC

Level 1: Knowledge and Comprehension

Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to:

- describe the characteristics of static, stack, and heap allocation
- explain issues related to disk read/write time
- define strategies for balancing a binary search tree
- define the algorithms for implementing B-tree operations
- define the procedure for conducting an external sort

3. Policies and Other Information

Prerequisites: Minimum "C" grade in CS2401 and MATH 2300.

Attendance Policy: Students are allowed at most five absences. Three tardies will count as one absence. A tardy will be recorded each time a student shows up ten minutes after the start of class.

Textbook: Reading and laboratory assignments will be drawn from *Data Structures outside in, with Java* by Sesh Venugopal. You are required to obtain this book for use in this course. Note that photocopied textbooks are a violation of copyright law. Any student caught with a photocopied book will be referred to the Dean of Students for discipline.

Grading: Final grades will be based on a combination of lab projects, homework assignments, in-class attendance and performance, four partial exams, and a final exam. The approximate percentages are as follows:

- 24% - Lab projects
- 10% - Written homework assignments
- 10% - In-class exercises, quizzes, and anti-quizzes
- 32% - Partial Exams (4 exams, 8% each)
- 24% - Final Comprehensive Exam

The nominal percentage-score-to-letter-grade conversion is as follows:

- 90% or higher is an A
- 80-89% is a B

- 70-79% is a C
- 60-69% is a D
- below 60% is an F

Additionally, any one of the following **will result on a final grade of F**, even if the overall average is greater than 60%.

- Obtaining an average of less than 60% on the lab projects
- Missing more than six combined hours of lectures, labs and PLTL sessions

We reserve the right to adjust these criteria downward, e.g., so that 88% or higher represents an A, based on overall class performance. The criteria will not be adjusted upward, however. You must earn a C or better to be able to register for upper division computer science courses.

Late homework submission: Homework up to a day late will receive up to 80% of full credit, and it will not be accepted after that.

Collaboration: Discussion of homework and projects among students is encouraged, but your answers and your code should be written and tested by you alone. **Do not exchange programs or let someone look at your code, even "just so they can see how you did it."** If you need help, consult the professor, your TA, or your peer-leader.

Cellular telephones are prohibited during lecture and lab sessions. Students are required to turn off their cellular telephones before entering the classroom or laboratory session. If your cell phone rings while you are at a lecture or laboratory session the TA will mark you as absent.

Disabilities: If you feel that you may have a disability that requires accommodation, contact the Disabled Student Services Office at 747-5184, go to Room 106E Union, or email dss@utep.edu

4. Lab Submission Guidelines

Lab assignments will be posted on-line. Each lab grade will be computed from the following three elements:

- Report (30% of grade)
- Source code (70% of grade)
- Demo session (pass/fail)

Report:

You must submit a printed report of every lab that includes the following items:

- Introduction – Description of the problem you are trying to solve
- Proposed solution – How did you solve (or attempt to solve) the problem? Provide an informal, high-level description
- Implementation – Description of your code (not the actual code). Explain the design choices you made, including how you broke the program into modules, your user interface, input and output, etc.
- Experimental results – Describe the experiments you performed to test your program. The experiments must be described in a way that allows anybody to replicate them using your code.
- Conclusions – Explain what you learned from the project.

Reports will be graded as follows:

- Completeness (8%)
Does your report cover all required aspects in enough detail?
- Clarity (8%)
Are those aspects clearly explained?
- Language (8%)
Is the report written with proper grammar and spelling?
- Presentation (6%)
Is the formatting appropriate?

Source Code:

Working programs must be submitted online to both your TA and instructor, using the e-mail addresses listed below. Labs not submitted this way will not be eligible for credit.

- Olac Fuentes: olacfuentes@gmail.com
- Patricia Reyes: preyes5@miners.utep.edu
- Christian Servín: christians@miners.utep.edu

Source code will be graded using the following guidelines:

- Correctness (42%)
Does the program compile?
Does the program run correctly?
- Design (7%)
Are operations broken down into methods in a reasonable way?
- Style (7%)
Is the program indented correctly and consistently?
Do methods and variables have meaningful names?
- Robustness (7%)
Does the program handle erroneous or unexpected input gracefully?
- Documentation (7%)
Do all program files begin with a comment that identifies the course, author, assignment, instructor, T.A., date of last modification, and purpose of program?
Are all methods clearly documented?
Are all non-obvious code segments clearly explained?

Demo session:

After submitting your program, you must schedule a one-on-one session with your TA in which you will explain how your code works and he/she will ask questions to test your understanding of the program being submitted. The TA will then assign a pass/fail grade for this session. A student receiving a failing grade in this session will receive a grade of zero for the whole lab, otherwise he/she will receive the grade corresponding to the combination of submitted report and source code. Demo sessions will last between five and ten minutes and will normally be scheduled either during lab hours or during the T.A.'s office hours.

Policy on late projects:

Lab project grades will be reduced by a factor of 8% for each day they are late.

Official turn-in dates:

For grading purposes, the official turn-in date for labs is when all three parts are finished. Thus, a lab will be considered to be late if ANY of the three parts is late. There will be a two working day grace period for reports and demo sessions. For example, if a lab is due on Monday, the source code must be submitted on or before Monday, and the report must be submitted and the demo shown on or before Wednesday.

Laboratory Sessions: Laboratory sessions are designed to give you guidance in getting your homework assignment started well. In a typical lab session, the Teaching Assistant will present additional material that will help you complete the assignment and answer your questions as you begin working. Attendance to these sessions is mandatory unless you have already finished the current assignment and no new assignments have been given.

Once you have completed the current assignment and you have submitted it via e-mail you may work on the next assignment or leave the laboratory, if it is not a Peer-Led Team Learning session. However, during lab sessions you should not use the computers for something other than working on your lab projects. The TA will mark as absent any student violating this policy.

5. Peer-Led Team Learning (PLTL) Sessions

PLTL sessions are designed to help you practice course concepts by engaging you in group activities. These sessions will take place once a week during your regular lab sessions and will last for 50 minutes. Attendance to these sessions is mandatory.

6. Standards of Conduct and Academic Dishonesty

You are expected to conduct yourself in a professional and courteous manner, as prescribed by the UTEP Standards of Conduct: <http://studentaffairs.utep.edu/Default.aspx?tabid=4386>

Academic dishonesty includes but is not limited to cheating, plagiarism and collusion. Cheating may involve copying from or providing information to another student, possessing unauthorized materials during a test, or falsifying data (for example program outputs) in laboratory reports. Plagiarism occurs when someone represents the work or ideas of another person as his/her own. Collusion involves collaborating with another person to commit an academically dishonest act.

Professors are required to - and will - report academic dishonesty and any other violation of the Standards of Conduct to the Dean of Students.