CS 2420 Algorithms and Data Structures

Instructor: Vicki Allan

Office: Main 429

Communications: Phone: 797-2022, E-mail: Vicki.Allan@usu.edu. This course is being video-taped for distance students. Class lectures are available at tcs.uen.net/tcs. I find I need to download the lecture before playing it. Email will normally be answered within twelve hours, excluding weekends.

Office Hours: TH 10:30-1. Additional office hours are available by appointment.

Prerequisite: CS 1410 or equivalent knowledge. If you feel that you are a poor C++ programmer, your chances of succeeding in this class are slim. I would strongly recommend that you take CS1410 instead of this class. I expect that you know the following material: structs, unions, file operations, classes, friends, overloading operators, inheritance, polymorphism, virtual, exceptions, templates, linked lists, stacks, queues, recursion, and binary trees. There is so much new material to be learned, you do not have time to learn material that you should have mastered in earlier classes. Sometimes students like to gamble on success. The laws of gambling state that the smaller the probability of succeeding, the greater the reward. Many times the gamble in taking a class (given the time you are willing to spend and the skills you bring to it) has a very low probability of success and a small reward (you may barely pass, but the hours required may cause you to do poorly in other classes).

Every semester I see many students throwing away time and tuition money by taking classes they can't possibly pass with their skills and/or their commitment. They go through the motions of taking the course, but don't learn what they need to. Some students say, "The instructor is so nice. If I come every day and try my best, surely she/he will give me a passing grade." It won't happen. Giving you a passing grade when you haven't earned it is not only unfair to others, but it is no favor to you. You would go on to struggle in other classes.

Tutors: Tutors for data structures are available in Main 425. Please use them whenever possible. If a tutor is on duty, I expect that you will consult with her/him before you come to see me. The hours for fall semester are:
   Monday through Friday: 10:30-9:00pm
   Saturday: 12:00-4:30pm

If there are tutors that are especially good or bad, please let me know about it. We want the quality of the tutors to be high.

Course Page: All assignments and other information dealing with the course are posted on the course page www.cs.usu.edu/~allany/cs2420/cs2420F11.htm


Objectives: The main objectives of this course are: a mastery of data structures, a mastery of refining programming skills, a mastery of developing strategies for the design and evaluation of algorithms, a familiarity with algorithm analysis, a mastery of recursion, a familiarity of sorting algorithms, a familiarity of graphs, a familiarity of trees, a mastery of binary search trees, a mastery of hash tables, a mastery of priority queues, a mastery of splay trees, a mastery of merging priority queues, a familiarity of disjoint set classes, an exposure to dynamic programming, an exposure to greedy algorithms. Memorization or copying is not learning and will not be encouraged. Class discussion utilizes discovery learning and will be very different in nature from the step by step, cookbook approach of most texts. Since you will experience both the text's presentation and the derivation of the ideas in class, you will have the benefit of both teaching techniques.
Requirements:

Programming Assignments: There are seven programming assignments during the semester. The point values of the programming assignments are not commensurate with the time involved to complete them. Programming assignments should be viewed as essential preparation for exams, rather than work that is adequately rewarded. All programs are to be written using C++. You may use any computer system you desire. Each program must be your own work (this includes not allowing a tutor to write your programs). You are not allowed to use the STL (standard template library) unless the assignment explicitly states you can.

All programming assignments are to be turned in using the eagle system. Go to the eagle system and enroll in the class. All communication about the class and/or grading should be directed to me (Vicki.Allan@usu.edu). Assignments turned in after 11:59:59 p.m. on the date due are late. Students are responsible for turning in their programs on time.

I rarely alter the due date of an assignment, and will not do so unless all students can be informed of the change at least two days before the original due date. Computer failures and file loses are a part of dealing with computers and will not be considered an excuse except in extreme circumstances.

Use the Style Guidelines available from the class web page. Your programs are graded based on these guidelines; make sure you understand them. You will lose points for violating the standard. Unless otherwise specified, programming assignments are graded as follows:

- 40% Program contains no functional errors and produces correct output.
- 30% Efficient, well designed, extendible code.
- 15% Readability, good variable names, readability.
- 10% Comments. Well commented source code is often a necessity for others who will read your code. This includes explanation of variable names, functions, and descriptions of chunks of code. Note, comments should not be on every line.
- 5% Format of output is pleasing and easy to understand. A person should be able to tell what the program does my just looking at the output. Put enough information in the output so this is true.

Programming is an important part of this class. You cannot receive higher than a D+ if you are missing any programming assignments or you have less than 60% of the programming points. This is true regardless of the total points earned.

Written Assignments: We will have eight written homework. Written homework takes the following form. Specific questions are posted on the course web page. You may work in groups of one, two, or three. Groups may change throughout the semester. Feel free to visit with me about possible answers to homework exercises. If more than one person is involved, list all the names on ONE set of answers. Answers should not be compared with others not in your group. Thus, it is cheating if you visit with others about answers, but turn in the homework with only your name on it. The written assignment must be in one of the following formats: .doc, .docx, .pdf, or .odt.

You will learn much more by working in a group than you will learn working by yourself. Educationally, it is a superior experience. You have to defend your answers. You get to take turns explaining and being taught. There are more of you to seek help from me, should you need it. When you do seek help, you are more confident that you have an important question as there are three of you with the same question. Thus, you don't feel "It's just me." Instead of just skipping a question you don't understand, you are able to iterate through several choices. You come to class having really worked on every question. For good students, it will increase your understanding. For poor students, it may be the only way that you survive. I realize there are some of you who absolutely have no time to work with others, but these situations should be rare.

Exams: There are two midterm exams (100 points each), given on October 6th and November 15th, and the final (150 points) given on Thursday, December 15th, at 9:30am. Exams cover material presented in class, in the book, and on the assignments. I do not give makeup exams. The final is comprehensive, is not optional, and is not given early. Please verify that you are able to take all the exams on the dates specified.

Grading:
<table>
<thead>
<tr>
<th>Assignment Type</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming Assignments</td>
<td>140</td>
</tr>
<tr>
<td>Written Assignments</td>
<td>80</td>
</tr>
<tr>
<td>Exams</td>
<td>350</td>
</tr>
<tr>
<td>Attendance and Preparation</td>
<td>0-50</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>620</strong></td>
</tr>
</tbody>
</table>

You may be given an F if your overall average is below 50%. Remember, you cannot receive higher than a D+ if you haven't turned in all the programming assignments or don't have over 60% of the possible points on the programming assignments.

Current scores can be checked on the eagle system. Special announcements about the class are e-mailed to you, so it is suggested that you check your e-mail daily.

Grades typically fit the following pattern:

<table>
<thead>
<tr>
<th>% of points</th>
<th>Grade</th>
<th>% of points</th>
<th>Grade</th>
<th>% of points</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>94-100</td>
<td>A</td>
<td>90-93</td>
<td>A-</td>
<td>87-89</td>
<td>B+</td>
</tr>
<tr>
<td>83-86</td>
<td>B</td>
<td>80-82</td>
<td>B-</td>
<td>76-79</td>
<td>C+</td>
</tr>
<tr>
<td>70-75</td>
<td>C</td>
<td>65-69</td>
<td>C-</td>
<td>60-64</td>
<td>D+</td>
</tr>
<tr>
<td>50-59</td>
<td>D</td>
<td>&lt;50</td>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Course Outline:**

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Algorithm Analysis - sections 2.1, 2.2, 2.3, 2.4</td>
</tr>
<tr>
<td>4</td>
<td>Trees - sections 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7</td>
</tr>
<tr>
<td>5</td>
<td>Hashing - sections 5.1, 5.2, 5.3, 5.4, 5.5</td>
</tr>
<tr>
<td>6</td>
<td>Priority Queues (Heaps) - sections 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8</td>
</tr>
<tr>
<td>7</td>
<td>Sorting - sections 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7</td>
</tr>
<tr>
<td>8</td>
<td>The Disjoint Set Class - sections 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7</td>
</tr>
<tr>
<td>9</td>
<td>Graph Algorithms - sections 9.1, 9.2, 9.3, 9.4, 9.5, 9.6</td>
</tr>
</tbody>
</table>

**Regrading:** If you feel that a program, written assignment, or exam has been graded incorrectly, submit a concise written summary of your concerns to me. These requests should be submitted within a week of the return of the assignment or exam.

**MSDNAA Elms Account:** The Computer Science Department is a member of the Microsoft Academic Alliance (MSDNAA), and specifically what is now called the Developer Academic Alliance. Through this program, students in CS courses can obtain and use a large number of Microsoft's operating systems and software packages. If you are interested in downloading any of this software for your use, please follow the directions found at [http://www.cs.usu.edu/elms.html](http://www.cs.usu.edu/elms.html).

**Etiquette:** My time at the university is spent in two major chunks: time teaching and time doing research. As a university professor I am expected to excel in both areas. In order for our relationship to be a happy one, you need to understand where I'm coming from.

Tuesdays and Thursdays the majority of my time is spent with office hours, teaching, and class preparation. All my office hours are scheduled for these days.

During office hours, I try not to let one student take too much time if others are waiting. If, during my office hours, I see you have been waiting, I may ask if you have a quick question or want to listen to the explanation or may even ask the person I have been helping to do something independently while I help you. But the decision to interrupt what
I'm doing to help you is my decision. For more interesting reading about etiquette, see the web page http://www.cs.usu.edu/~allan/Advising/ Etiquette.html.

**Tardiness:** When you come to class late, every person in the room is distracted by your entrance (including your professor). You miss important material. You are saying to your professor, "My time and my schedule are more important than what you have to teach me." On the job, if you don't come to work, you will be fired. If you come late, you will be reprimanded. Come to class and come on time! It is good practice for the real world.

**Cell Phones:** Please turn off cell phones before coming to class. When a cell phone goes off, or you are texting, or reading a text message during class, it disturbs your teacher and everyone else in the class. The penalty for any of the above is that you will bring treats for the whole class the next period!

**Time:** This class takes time. Be aware that a programming class usually demands a greater time commitment than other classes. You should realize that much of the time spent comes because students are either poorly prepared for the course or do not debug wisely. Thus, you control much of the time yourself.

I would ask the following. Before you complain about the amount of time you are spending on the course, ask yourself the following questions:
- Do I start assignments late so class explanation are wasted because I'm not ready for the answers given?
- Do I refuse to learn to use the debugger?
- Do I hack at the leaves instead of getting to the root of the problem?
- Do I refuse to figure out why something is happening, but try to debug totally from observing what happens when I make random changes?"

It is extremely frustrating for students to complain, but not be willing to take the steps necessary to eliminate the problem. If you don't do everything in your power to make this course reasonable, I can guarantee you will be miserable! The result isn't intended, but it is inevitable.

**Standards:** All work in this class is graded based on the best solution rather than merely solving the problem. Sometimes students feel it is unfair for a teacher to expect the best when they are just learning, but it is important for a student to strive for the real goal. On the job, a program that "just works" is almost never adequate. How can one expect to improve or even recognized quality if it has never been demanded?

**Late Work:** The most common problem in this class is failure to complete the programs on time. Students are typically optimistic about the amount of time it takes to write a program, and tend to budget their time for the best possible case instead of for the average or worst case. In addition, when problems do arise, a person tends to think the she/he is the only one with such unforeseen problems and anticipates exceptions will certainly be made. Once a person gets behind with one program, it is very common to be behind on many programs either because a late finish on one dictates a late start on the next or because the penalty was not sufficient to avoid similar pitfalls.

Late work creates difficulties in grading. Unless a very strict policy is enforced, chaos reigns. It is not that I am insensitive to your personal problems, but rather that I must insist that you rise above them. When the instructor grants an extension to one student, it is unfair to the other students who would have benefited from such special treatment.

All programming assignments are due at 11:59:59 p.m. on the date specified. However, each student is entitled to one personal emergency. Thus, you are allowed to turn in one program up to one weel late without penalty (and without explanation). When the personal emergency has been used, late work is accepted with a penalty of 10% per day.

**Code Reuse:** For this course, it is almost never appropriate to copy code from the book or another source. When you graduate, you will often pull code from another source, but at this stage in your development, you need to write it! I try to make the assignments different from what you can find in books or on the web, but sometimes that makes the assignments harder than I would like. When I ask you to write code that has been written by thousands of others before you, you still need to write it so you appreciate it, so you learn the lessons contained therein. You learn next to nothing
by copying it from elsewhere. Using the standard template library is also not allowed (without specific permission from your instructor).

In English, if you are to write a paper, you are not allowed to find a good one on the web and turn it in. In CS, it is a little different, because you are encouraged not to reinvent the wheel. However, in this class, I want you to "invent the wheel." If you don't know how the wheel was build, you can't improve upon it. Learning must be done in layers. I can't teach you how to code exotic programs unless you have done the simpler things. Writing the simpler things yourself - independently of the book - is necessary to form the correct foundation for what you need for later algorithms.

In any course, you should never use someone else's product without clearly stating where it came from. To use someone else's creation without giving them credit is cheating. Here's the approach I would try. Study the book. When an assignment is given, try to do it without looking at anything. If you can't, study the book again - but before you start coding, shut the book. Then, you know that you have retained the most critical parts of the design.

The penalty for cheating is intended to have a punitive effect. If the penalty is so light, students can look at the penalty they receive and the probability of being caught and maximize their expected utility. So the thinking could be, “If I cheat, I will only get caught 5% of the time. When I’m caught it may lower my grade from a C+ to a C. However, the 95% of the time I don’t get caught, I win. I save 10 hours a week in effort and get a better grade than I would have otherwise.”

Fill in your own values, but if the probability of being caught is low, unless the penalty for cheating is high, people come out ahead by cheating. I do not intend to let that happen. If I don’t punish cheating, I disadvantage those who are honest. I create a climate where cheating appears to be the better option. Otherwise honest people feel they have to cheat to be competitive.

**Preparation:** Preparation is necessary for learning. For this class, preparation includes attending class regularly (90% of the time), coming on time, remaining focused until class is dismissed, asking timely questions, trying problems at your seats when directed to do so, answering questions when called upon, completing homework questions, paying attention during lecture, and reading appropriate material before coming to class.

Because you learn more if you are involved in class discussion, I often ask for class response to a question. However, do not feel that you need to answer every question. I would like to hear from everyone in the class not the same two or three people every time. To facilitate hearing from everyone, the "three strikes rule" is implemented. After you have verbalized an answer three times in a class period, you are not allowed to answer any more questions that period. If your answer is so wonderful that you will die if it isn't expressed, tell your neighbor and let her/him share it. Along the same line, make sure the questions you ask are appropriate for the entire class. If you had a bizarre occurrence on your home computer, are wondering what you will miss when you travel to Alaska on Wednesday, or want to know how to use some advanced feature of the language, ask me after class. I don't want to spend class time on questions that are of interest to only a few or intimidate others by answering questions they are not ready for.

**Department Code of Conduct:** Although you may collaborate on problem solving, each person must write, debug, and test his/her own work. Some students feel that if they worked with the other person for a long period of time (as opposed to just copying another's work without any personal effort), they haven't cheated. That is not true. Note, you don't have to feel guilty for what you have done to have it be cheating. There are degrees of cheating. Copying another person's code (with or without their knowledge) is cheating. However, if another person helps you so much that the result isn't your work, then it is still cheating, regardless of how many hours you spent in the process. Some think that because the course is demanding, it gives them license to cheat. Nope! Cheating is cheating. There is no set of circumstances that justifies it.

Submitting projects which include part or all of code/text that is obtained from sources such as a text book or a site on the Internet, without properly acknowledging the source(s) of the code/text in your submission is considered cheating. In most cases, such "borrowing" is not permitted, with or without proper acknowledgement. In case of doubt, talk to me about your intention to include such code before using it.
I do not recommend designing the code with others because it is difficult to pull yourself away from the group and do your own work. For every problem you encounter, you want to know how others in your group solved that problem. Copying code from the text or other sources (including tutors and consultants) is not allowed unless specifically indicated. Do not allow someone else to write the code for you no matter how helpful it would be. Cheating will be dealt with severely. The penalty given is worse than not having done the assignment, written the program, or taken the exam (i.e., the least penalty possible is negative points). When cheating occurs, a letter to that effect is placed in your file and a copy is sent to the dean's office. Some excuse their cheating by saying "I was under so much pressure and just ran out of time." This is like saying, "I am honest until it is inconvenient." Cheating is not tolerated.

Flagrant cheating involves turning in another's work as your own (even if that work was done by a tutor). However, there are many other forms for dishonesty that are also considered cheating. Allowing others to copy from your work is considered cheating, and both of you will be penalized. Do not put your friends in an awkward position by asking them to help you cheat.

If there are any questions, please refer to see the department Code of Conduct.

**Incompletes:** According to university policy, incompletes are not to be given for poor performance. There will be no incompletes given except for conditions beyond the student's control. Such conditions have to have written documentation. The term "conditions beyond the student's control" includes (1) incapacitating illnesses that prevent a student from attending classes for a period of at least two weeks; (2) a death in the immediate family; (3) financial responsibilities requiring a student to alter course schedule to secure employment; (4) change in work schedule as required by an employer; or (5) other emergencies of this nature. When an incomplete is given, it is anticipated that the remaining work will be finished within two or three weeks. If the course must be retaken to make up the work, an incomplete is not appropriate. There are provisions in case of emergency to permit a student to withdraw (grade of W) from a course after the regular drop period when it is not feasible to give an I.

**ADA Statement:** Students with ADA-documented physical, sensory, emotional or medical impairments may be eligible for reasonable accommodations. Veterans may also be eligible for services. All accommodations are coordinated through the Disability Resource Center (DRC) in Room 101 of the University Inn, (435)797-2444 voice, (435)797-0740 TTY, or toll free at 1-800-259-2966. Please contact the DRC as early in the semester as possible. Alternate format materials (Braille, large print or digital) are available with advance notice.

**Class Fees:** Associated with this class is a fee of $25. The monies from this fee are used to maintain lab facilities for the class, purchase software and licenses, and supervise the lab. In some cases, students may have their own computing equipment, and thus feel that they do not need to use the lab. However, the lab must be maintained regardless of and individual's use of it, and thus the fee is charged to all registered for the class. If you have questions or concerns about the fee, please see the department head.

**Late Adds:** The last day to add this class is September 19th. Attending this class beyond that date without being officially registered will not be approved by the Dean's Office. Students must be officially registered for this class. No assignments or tests of any kind will be graded for students whose names do not appear on the class list.

**Drop Dates:** The last day to drop classes is
- September 19th - without a notation on transcript.
- October 28th - with a "W" notation on transcript.