CS310 – Advanced Data Structures and Algorithms
Syllabus

Instructor:  Nurit Haspel (nurit.haspel@umb.edu)

Course Description and Objectives

• A systematic study of the methods of structuring and manipulating data in computing.

• Application programming interfaces (APIs), data abstraction, and the encapsulation of implementations, familiarity with the Java Collection Classes, Advanced techniques for program development and organization.

• The design and analysis of algorithms, including run time analysis.

• Familiarity with advanced algorithmic techniques such as divide and conquer, dynamic programming, graph algorithms and greedy algorithms.

Prerequisites

• CS110 and CS210 (or one year of higher-level language instruction in Java, C, C++ or similar computer language, and fluency in Java).
  – Knowledge of stacks, queues, binary search trees, sorting (covered in CS210). More recently, graphs are being covered in CS210. If you are not familiar with graphs, let me know and I’ll give you some pointers.
  – Static and dynamic memory allocation, the stack and the heap.

• CS240 (or knowledge of C and Unix). If you have not worked with Unix before, please refer to a basic Unix guide here: http://www.cs.umb.edu/~ghoffman/linux/unix_cs_students.html

• CS220 (Applied Discrete Mathematics).
  – Basic proof techniques - contradiction, induction, reasoning.
  – Basic runtime analysis.
  – Mathematical formulas: Summations, counting and combinatorics.

• Math 140 (Calculus 1) indirectly, since it is also a pre-requisite of CS220.
Textbooks


Topics

- Review of basic and advanced algorithm analysis: Big O. The tyranny of growth rates.
- Review of Collection classes, continued from CS210. Maps and Sets and the search for $O(1)$. Looking for fast insertion and retrieval algorithms in various contexts. Recognizing the right collection for an application.
- Hash tables, collision resolution techniques.
- Algorithmic techniques: Divide-and-conquer, dynamic programming, greedy algorithms.
- Data compression: Huffman’s coding, BWT, LZW.
- Graph algorithms: Graph API and implementations. traversals, shortest paths, A*, strongly connected components, spanning trees, flow.
- Complexity classes and NP if time permits.

Assignments and Grading

The following grading scheme is subject to change (I will do my absolute best to not change it and if I do, I’ll let you know):

- Homework and Programming Assignments – 15% of your final grade for each type of assignment (30% total)
- Midterm exam – 30% of your final grade
- Final exam – 40% of your final grade. You must also pass the final exam (at least with a grade of 40) to pass the course.

You must have a documented reason to schedule a makeup exam. I must know that you need a makeup exam within 2 days after the exam date.

Notice: No courses required by the CS major, minor, or certificate may be taken pass/fail.

Final Grade

Your final grade will be calculated using the following table. The minimum standard for passing the course is a percentage score of 40%. You also must pass the final exam (score at least 40% in the final exam). Keeping this in mind, your grade for the course will be calculated using the following table. Assume your final percentage score for the course is P:
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$P &gt; 90$</td>
<td>A</td>
</tr>
<tr>
<td>$85 &lt; P \leq 90$</td>
<td>A-</td>
</tr>
<tr>
<td>$80 &lt; P \leq 85$</td>
<td>B+</td>
</tr>
<tr>
<td>$75 &lt; P \leq 80$</td>
<td>B</td>
</tr>
<tr>
<td>$70 &lt; P \leq 75$</td>
<td>B-</td>
</tr>
<tr>
<td>$65 &lt; P \leq 70$</td>
<td>C+</td>
</tr>
<tr>
<td>$60 &lt; P \leq 65$</td>
<td>C</td>
</tr>
<tr>
<td>$55 &lt; P \leq 60$</td>
<td>C-</td>
</tr>
<tr>
<td>$50 &lt; P \leq 55$</td>
<td>D+</td>
</tr>
<tr>
<td>$45 &lt; P \leq 50$</td>
<td>D</td>
</tr>
<tr>
<td>$40 &lt; P \leq 45$</td>
<td>D-</td>
</tr>
<tr>
<td>$P &lt; 40$</td>
<td>F</td>
</tr>
</tbody>
</table>

**Accommodations**

Section 504 of the Americans with Disabilities Act of 1990 offers guidelines for curriculum modifications and adaptations for students with documented disabilities. If applicable, students may obtain adaptation recommendations from the Ross Center for Disability Services, M-1-401, (617-287-7430). The student must present these recommendations and discuss them with each professor within a reasonable period, preferably by the end of Drop/Add period.

**Student Conduct**

Students are required to adhere to the University Policy on Academic Standards and Cheating, to the University Statement on Plagiarism and the Documentation of Written Work, and to the Code of Student Conduct as delineated in the catalog of Undergraduate Programs, pp. 44-45, and 48-52. The Code is available online at: [www.umb.edu/student_services/student_rights/code_conduct.html](http://www.umb.edu/student_services/student_rights/code_conduct.html)

**UNIX accounts, class email**

You can test your work on the Department’s network of Unix systems, or you can work on your home computer and deliver the projects to Gradescope. Either way, it is recommended to have an account at our site.

- Apply for an account as soon as possible, following the instructions posted here: [https://www.cs.umb.edu/sp/resources/other/faqs/#FAQ02](https://www.cs.umb.edu/sp/resources/other/faqs/#FAQ02)

- I will be using your umb email (the one that looks something like: first.last001@umb.edu) to communicate with the class and for Piazza and Gradescope. **Be sure to read your e-mails regularly.** If you prefer another e-mail address (say gmail), forward your UMB e-mail there.

- Look for the line ”module load ” in your UNIX .cshrc file and add java to the end of it, so you will be using a current Java distribution

**Homepage**

The course home page is [http://www.cs.umb.edu/cs310](http://www.cs.umb.edu/cs310). This directory is visible in the filesystem of our UNIX machines as /courses/cs310. All material for this course will be kept in this area, which may sometimes be called $cs310.