CMPT 225
Course Overview
Data Structures

- A course on **Data Structures**: common ways of organizing computer memory, with algorithms that manipulate this memory.
- We use the **Abstract Data Type** approach, which goes hand-in-hand with object-oriented programming.
- The computer language we will be using is C++, and there will be a lot of programming, but this is **not** a programming course in the same way as first-year courses are.
Professor and TAs

Dr. Thomas C Shermer, a.k.a. Tom
TASC-I 8021
shermer@sfu.ca
Office hours: MWF 15:30-16:20 (or by appointment) on Zoom

TAs:
Pourya Vaziri pvaziri@sfu.ca
Chao Zhang cza131@sfu.ca
**Text**

*Data Structures and Algorithms in C++*

by Goodrich, Tamassia, and Mount, 2\textsuperscript{nd} edition.

The text is required.

It’s an all-around good text. Quite clear and has good types of examples. As a theoretician and pragmatist, I’m impressed.

(As a software engineer, there are a few things I’d change with the examples, but that’s not a big concern at this point.)
Marking

- Midterm Exam 10%
- Final Exam 30%
- Homework (6) 60%

The final covers the entire course (it is cumulative).
Marking Policies

- Partial marks are given on exams:
  - If you get the wrong answer but show work that shows some understanding, you will get some marks.
  - If you get the right answer but show work that shows some misunderstanding, you will lose some marks.

- In the event of a marking dispute (you think your mark isn’t fair) first contact the marking TA to try to resolve it. If that doesn’t resolve it, then bring it to the professor.
## Important Dates

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sept 9</td>
<td>Classes Start</td>
</tr>
<tr>
<td>Sept 25</td>
<td>Homework 1 due</td>
</tr>
<tr>
<td>Oct 9</td>
<td>Homework 2 due</td>
</tr>
<tr>
<td>Oct 12</td>
<td>No class (Thanksgiving)</td>
</tr>
<tr>
<td>Oct 23</td>
<td>Homework 3 due</td>
</tr>
<tr>
<td>Oct 28</td>
<td>In-class midterm</td>
</tr>
<tr>
<td>Nov 6</td>
<td>Homework 4 due</td>
</tr>
<tr>
<td>Nov 11</td>
<td>No class (Remembrance Day)</td>
</tr>
<tr>
<td>Nov 20</td>
<td>Homework 5 due</td>
</tr>
<tr>
<td>Dec 4</td>
<td>Homework 6 due</td>
</tr>
<tr>
<td>Dec 7</td>
<td>Last day of class</td>
</tr>
<tr>
<td>TBA</td>
<td>Final exam</td>
</tr>
</tbody>
</table>

© 2019 Shermer
Laboratories

- Due to the pandemic, we can't gather in a lab and so we will **not** have lab sessions.
- Instead, I have increased the number of homeworks so that we basically have one every two weeks.
- Be sure to attend TA (or professor) office hours if you need help with homeworks.
Assignment Submission

- Assignments must be submitted by 11:59 pm on the due date.
- Assignments are to be submitted on CourSys (coursys.sfu.ca).
- **Late penalties** are -10% per day, up to 5 days. Days are calendar days—weekends and holidays count.
- Assignments submitted after 5 days late will be given a 0.
Course Syllabus

- We will follow the text. You will gain the most benefit by reading ahead of lecture.
- The approximate pace is one chapter per week.
- We will not finish the book, but I do recommend finishing it on your own.
- Chapter 1 is a C++ Primer and I assume you have this knowledge from your prerequisites. Please read Chapter 1 and ensure that you know the material, including the part on pseudo-code.
- Lecture will start with Chapter 2, Object-Oriented Design.
Course Syllabus

- Chapter 2: Object-Oriented Design
- Chapter 3: Arrays, Linked Lists, and Recursion
- Chapter 4: Analysis Tools
- Chapter 5: Stacks, Queues, and Deques
- Chapter 6: List and Iterator ADT's
- Chapter 7: Trees
- Chapter 8: Heaps and Priority Queues
- Chapter 9: Hash Tables, Maps, and Skip Lists
Course Syllabus

- Chapter 10: Search Trees
- Chapter 11: Sorting, Sets, and Selection
- Chapter 12: Strings and Dynamic Programming
- Chapter 13: Graph Algorithms
- Chapter 14: Memory Management and B-Trees
C++ and Java and ...

- We use C++ exclusively in this course.
- Each computer language is a tool with its own characteristics, strengths, and weaknesses.
- Don’t argue over whether a hammer or a screwdriver is a better tool. Or C++ or Java.
- C++ is a language designed so that correct programs compile quickly.
- Java is a language designed so that incorrect programs are easy to diagnose.
- Use whichever tool is appropriate for the problem at hand.
Software

- **g++ (GCC 10.2)**
  - available at https://gcc.gnu.org/

This is the compiler that we will use to compile your programs. You can develop your programs on whatever platform you like, provided what you submit works with GCC 10.2.

The standard runtime libraries are allowed but other libraries (e.g. libraries of data structures) are not, unless noted on the assignment.
Code Style - Comments

- Comment your code. Most student code is undercommented.
- Remove as many comments as possible from your code by making the code say what the comment says.

```c
// add today’s sales to yearly sales
ytd += sales;

yearToDateSales += dailySales;
```
void foo(int* A, int n) {
    ...
    // initialize the array A
    for( int i = 0; i < n; i++) {
        ...
    }
    ...
}

void initializeArray(int* A, int n) {
    for( int i = 0; i < n; i++) {
        ...
    }
    ...
}
Code Style - Optimization

- Premature Optimization is the root of all evil.
  - Clarity and correctness are often more desirable than speed.
  - When speed is an issue, first write the program clearly and correctly, then determine what code is slowing the program down, and only then optimize that code.
Code Style - Formatting

- **Always** format your programs consistently.
  - Indentation
  - Blank lines
- **In finished work, never** leave in commented-out or debugging code.
- **Always** include braces around a subordinate block:

  NO:
  
  ```
  for(int i=0; i<n; i++)
  sum += A[i];
  ```

  YES:
  
  ```
  for(int i=0; i<n; i++) {
  sum += A[i];
  }
  ```

© 2019 Shermer