Today

- Survey
- Course Overview

Assignments

- Exercise 1
Syllabus Overview

Communication:

• Website: cs.gonzaga.edu/bowers/courses/cpsc223
• Piazza (via invite)
• GitHub classroom (requires GitHub account)
• Blackboard (for posting grades)

Getting Help:

• Piazza: Preferred for (basic) questions, clarifications, etc.
• Office hours: Tu/Th 2–3; 11-12 & 1–2 Wed; or by appt

Grading: out of a total of 1,000 points

• Programming Assignments: 600 points (10 at 60 points each)
• Quizzes: 100 points (10 at 10 points each)
• Midterm Exams: 100 points (2 at 50 points each)
• Final Exam: 125 points (exam at 100 points, report at 25 points)
• Participation: 75 points (30 class meetings at 2.5 points per meeting)

Software: More later ...

• Dev tools: cmake, make, google test (gtest), g++, gdb, valgrind, git, gnuplot
• Either: VS Code remote development on ada server ... “supported”
• Or: Setup tools on your own machine (Mac or WSL) ... not “supported”
The material in this class is necessary for professional software development...

- technical interviews often ask you about your knowledge of data structures
- interviewers will often even ask you to live code what we discuss or extensions
- “Bad programmers worry about the code. Good programmers worry about data structures and their relationships.” – Linus Torvalds

Expectations:

- Stay engaged and participate in class
- Do your own work
- Ask when you have questions
- Start assignments early and give yourself enough time to succeed
- Assume you have everything you need already (and ask when in doubt)

(Some) collaboration is encouraged...

- can discuss and talk about assignments
- can work together (as long as you are doing your own work)
- plagiarism not allowed: no splitting up work, no code sharing, etc.

Things to avoid:

- Falling behind (assignments build on each other)
- Not studying the material (e.g., for quizzes)
- Not reading and following instructions
- Using YouTube, Google, StackOverflow, and other sites to “learn” the material
- Not coming to office hours and not using (or checking) piazza
Homework late and resubmission policy

- 12 point late penalty (20%) within one week after due date
- 15 point late penalty (25%) after one week past due date
- if score < 45 points (75%), can fix and resubmit (for max score of 45)
- but note that homework assignments build on each other

Online access to lectures (see syllabus for details)

- Only for those that have a COVID-19 illness or related issue
- Must notify me prior to lecture (I’ll need some lead time)
- Must send me your notes from lecture for participation credit

Homework warning:

- longer, more complex programming than you may have seen so far
- moves quicker than CPSC 121 and 122
- assignments like intense “exercises” (versus “real” programs)

Note on “Exercises”:

- take home exercises to help (mostly) with non-programming content
- will count towards “participation” grade
Basic Terminology

A **Data Type** defines
- a set of **values** and **operations** over the values

A **Base** Data Type
- is “built into” (provided by) the language
- e.g.: `int`, `double`, `bool`, `char`, sometimes `string`
- base-type values are "unstructured" (can’t be decomposed)

A **Composite** Data Type
- defines values that are structured (decomposable)
- e.g.: a record (struct) of base types, an array of integers

An **Abstract** Data Type (**ADT**)  
- is a data type without a specified implementation (usually many possible)
- usually for collections of data (e.g., instances can store 0 to \(n\) data items)
- e.g.: stack, queue, set, dictionary, tree, graph

A **Data Structure**
- is a concrete implementation of a data collection
- prescribes a particular structure / organization for the collection
- e.g.: resizable array, linked list, hash table
- useful for implementing different ADTs with different performance properties
Survey and Course Topics

C++ Programming ... (first few weeks)

• but not a learn-to-program C++ class!
• we’ll cover a couple of new (minor) topics
• primarily more practice with classes, pointers, recursion, etc.

Different Data Structures ... (majority of class)

• Brief review of linked lists
• Resizable arrays
• Hash tables
• Binary search trees
• Plus more as time allows (e.g., skip lists, heaps)

Data structures in the context of a few specific ADTs

• Sequences (first few assignments)
• Dictionary (most of semester) ... key-value pair collection
• Priority Queues (as time allows)

Algorithms and Analysis ...

• Adding, removing, and searching for data (main focus)
• Sorting data (“classic” sorting algorithms) ... useful for searching
• Comparisons of algorithmic performance (complexity vs experimental)