Data structures with C++

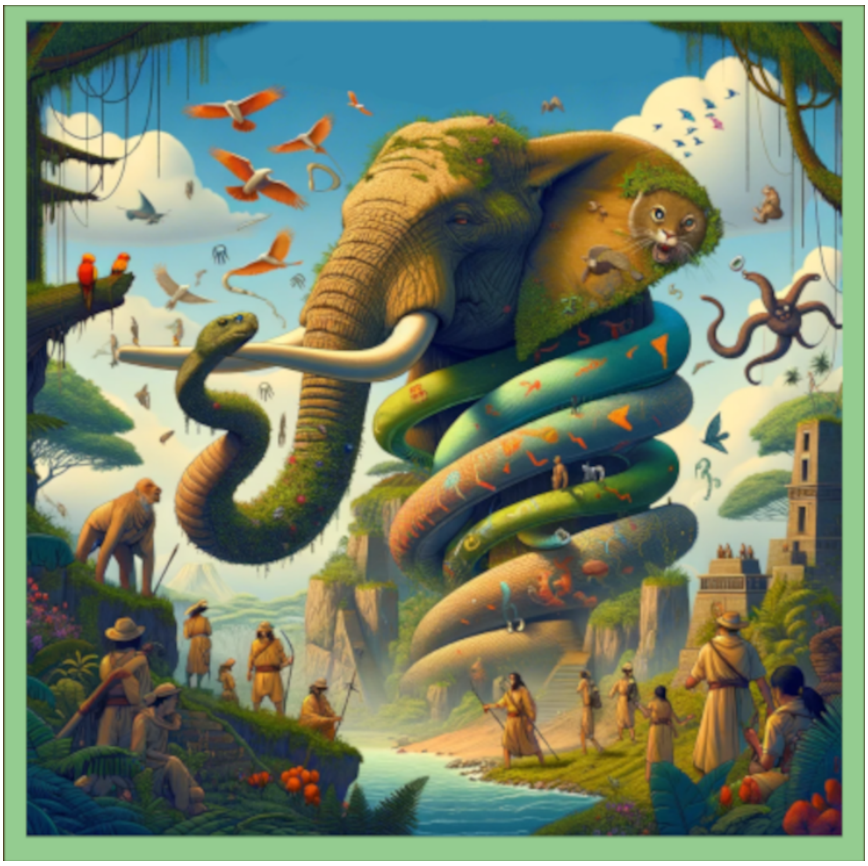
CSC 240 - Data structures with C++ - Syllabus - Fall 2024

Marcus Birkenkrahe

July 29, 2024

Contents

1 General Course Information



- Course title: Data structures with C++
- Course number and section: CSC 240.01
- Meeting Times: Tue-Thu 9:30-10.45 AM
- Meeting place: Derby Science Building computer lab room 239
- Professor: Marcus Birkenkrahe
- Professor's Office: Derby Science Building 210
- Phone: (870) 307-7254 (office) / (501) 422-4725 (private)
- Office hours: by appointment MWF 4pm, Tue/Thu 3pm, Fri 4 pm

- Textbooks (optional/recommended): Zingaro, Algorithmic Thinking (2e), NoStarch 2024. Shaw, Learn C the Hard Way, Addison-Wesley, 2015. King, C Programming (2e) W W Norton (2008), Griffiths & Griffiths, Head First, O'Reilly (2012).

2 Objectives

This course on "Data Structures with C++" offers an in-depth exploration into the theory and application of data structures using the C++ programming language. It aims to equip students with the knowledge to efficiently store, process, and retrieve data using various data structures such as arrays, linked lists, stacks, queues, trees, and graphs. The curriculum emphasizes the importance of algorithmic thinking and optimization techniques, providing a solid foundation for understanding the complexities and capabilities of modern computing systems.

3 Target audience

Ideal participants are those who have a basic understanding of programming concepts and are interested in advancing their knowledge in data structuring to improve software efficiency and performance.

4 Student Learning Outcomes

Students who complete this course will be able to:

- Demonstrate a thorough understanding of major data structures and their applications.
- Implement various data structures in C++, including arrays, linked lists, stacks, queues, trees, and graphs.
- Analyze the efficiency of data structures and algorithms in terms of time and space complexity.
- Apply algorithmic thinking to solve complex problems using appropriate data structures.
- Design and develop efficient software solutions for real-world applications.

- Critically evaluate the choice of data structures in software development projects.
- Employ advanced C++ features and object-oriented programming techniques in the context of data structures.
- Enhance problem-solving skills through the design and analysis of algorithms for data manipulation.
- Prepare to contribute effectively to technology-driven environments with a strong foundation in data structuring.

Students, who complete CSC 240 will have fulfilled the prerequisites for CSC 265 Algorithms.

5 Course requirements

Formal prerequisites: Introduction to Programming (either CSC100 or CSC115 or CSC109, and MTH101 (College Algebra).

Course requirements include a foundational knowledge of programming principles and familiarity with any programming language. Students are expected to have completed introductory courses in computer science or possess equivalent practical experience. A willingness to engage in complex problem-solving and the ability to think critically about algorithm design and data manipulation are essential for success in this course.

6 Grading

WHEN	DESCRIPTION	IMPACT
Weekly	Programming assignments	50%
Weekly	Multiple choice tests	25%
TBD	Final exam (optional)	25%

- Programming assignments are given for home completion
- Tests are open-book multiple choice exams for home
- The final exam is optional if you want to improve your grade

7 Rubric

Component	Weight	Excellent	Good	Satisfactory	Needs Improvement	Unsatisfactory
Participation and Attendance	0%	Consistently attends and actively participates in all classes.	Attends most classes and participates in discussions.	Attends classes but participation is minimal.	Frequently absent and rarely participates.	Rarely attends classes and does not participate.
Programming assignments	50%	Completes all assignments on time with high accuracy (90-100%).	Completes most assignments on time with good accuracy (80-89%).	Completes assignments but with some inaccuracies or delays (70-79%).	Frequently late or incomplete assignments with several inaccuracies (60-69%).	Rarely completes assignments and shows minimal understanding (0-59%).
Tests	25%	Demonstrates thorough understanding and application of concepts (90-100%).	Shows good understanding with minor errors (80-89%).	Displays basic understanding with some errors (70-79%).	Limited understanding with several errors (60-69%).	Minimal understanding and many errors (0-59%).
Final Exam (Optional)	25%	Demonstrates comprehensive understanding and application of course concepts (90-100%).	Shows strong understanding with minor errors (80-89%).	Displays adequate understanding with some errors (70-79%).	Limited understanding with several errors (60-69%).	Minimal understanding and many errors (0-59%).

8 Learning management system

- We use Lyon's Canvas installation for this course.
- The home page contains: assignments, grades, pages, people, syllabus, quizzes, Google Drive, Course evaluation and Zoom.
- The Zoom page includes cloud recordings of all past sessions.
- Recorded sessions will be deleted after the last class.

9 GitHub

All course materials are available in a public GitHub repository (github.com/birkenkrahe/alg1). Registration for students includes a free subscription to GitHub codespaces with the AI coding assistant Copilot. GitHub is the worldwide largest online platform for software development.

10 Lyon College Standard Policies

- tinyurl.com/LyonPolicyOnline, see also Class Attendance policy

11 Schedule

"Data structures" and "algorithms" are a little hard to separate:

- "Data structures" is concerned with storing and organizing data.
- "Algorithms" is concerned with using data to solve problems.

Both topics can be taught language-agnostic (without referencing a particular language) but your understanding will benefit greatly from examples and programming assignments.

Week	Content	Dates	Home Assignment	Tests
1	Introduction to course & dev tools	Aug 16	Emacs, GitHub	1
2	C++ Basics review	Aug 20, 22	Programs 1-2	2
3	Fundamental types and arrays	Aug 27, 29	Programs 3-4	3
4	User-defined types	Sep 3, 5	Programs 5-6	4
5	Reference types	Sep 10, 12	Programs 7-8	5
6	The object life cycle	Sep 17, 19	Programs 9-10	6
7	Linked lists	Sep 24, 26	Program 11-12	7
8	Doubly linked lists	Oct 1, 3	Program 13-14	8
9	Stacks & queues	Oct 8, 10	Program 15-16	9
10	Trees	Oct 15, 17	Program 17-18	10
11	Hash tables	Oct 22, 29	Program 19-20	11
12	Graphs	Nov 5, 7	Program 21-22	12
13	Heaps	Nov 12, 14	Program 23-24	13
14	Recursion	Nov 19, 21	Program 25-26	14
15	Hash tables, AVL trees, red-black trees	Dec 3, 5		15

12 A note on using AI to write code for you or debug your code

Short summary: For students, using AI is a waste of time at best, and a crime against your ability to learn at worst. Learning never comes without pain and (temporary) desperation. AI is like a pill but one that only works some of the time, and you'll never know when. Instead: join Lyon's Programming Student Club and experience the pain of not knowing first hand every week!

Will you be punished for using AI in my class? Not directly because nobody can tell if you used AI or not but indirectly by turning in

suboptimal results, by learning less, and by having less time for other, more productive activities.

Are there any data on this? Not much on coding as such but a recent (15 July), substantive, long (59 p) paper titled "Generative AI Can Harm Learning"), based on a very carefully conducted field experiment with a large (1000) sample of high school students concluded: "Our results suggest that students attempt to use [AI] as a "crutch" during practice problem sessions, and when successful, perform worse on their own. Thus, to maintain long-term productivity, we must be cautious when deploying generative AI to ensure humans continue to learn critical skills." (Bastani et al, 2024).

References

Bastani, Hamsa and Bastani, Osbert and Sungu, Alp and Ge, Haosen and Kabakcı, Özge and Mariman, Rei, Generative AI Can Harm Learning (July 15, 2024). Available at ssrn.com.