Group Project: Search

Data Structures CSCI 2320

Project Objective

- Enhance your understanding of search ADTs.
- Understand the implication of algorithm complexity (Big O) on search performance.
- Enhance your understanding of software development on a team
- Practice using git to manage your source code in a team.
- Practice designing and creating an experiment.
- Practice organizing and presenting technical information.

Project Overview

As a software developer for **Bear Flix**, you have been asked to construct an experiment to determine the best search ADT to use for storing, searching, and deleting movie data. You and your team must present your findings and recommendations to the lead architect and provide a recommendation for which data structure to choose.

Getting started

1. The starter repository for the project is available in GitHub Classroom. **One** person on the team follows the URL and creates the team repository. After that, all other team members follow the URL and join the established team. Your source code grade will be based upon your last sync to GitHub before the deadline.
2. You will need to create a shared Google folder for team documents. Remember to share with me when you create it (ptalley2@uca.edu). Your project report and presentation slides should be in this folder at the time of submission.

Project Details

Your team will implement five search ADTs. **You may use source from the Internet (including AI) for your ADTs ONLY, but you must cite the source in your code and your final paper, as well as verify that it works properly. You may not use any algorithms built directly into C++ unless explicitly stated.**

The four required search algorithms are:

- Linear Search using **std::list**
- Binary Search Tree (BST)
- AVL
- Hash Table

One additional search ADT we have not talked about in class:

- 2-3-4 trees
- Red-Black Trees
- B Trees

- B+ Trees
- Others you find on the Internet

Your team will design an experiment to compare the five different ADT's performance on three operations (insert, search, and delete) using three different data set sizes (ten thousand items, one million items and ten million items). **You may NOT use AI to generate your experiment driver (and I will know if you do).**

For each data set, your driver program will need to perform the following for each ADT.

- insert: Read in the random values, convert to integer and insert into the ADT. Report the total time for the inserts, the number of values successfully inserted, and the number of values that were not inserted.
- search: Read in the search values, convert to integer and search for the values in the ADT (created in the insert step). Report the total time for the searches, the number of values successfully found, and the number of values that were not found.
- delete: Read in the delete values, convert to integer and delete from the ADT (created in the insert step). Report the total time for the deletes, the number of values successfully deleted, and the number of values that were not deleted.

You may use your creativity to implement the driver code. I recommend using chrono for capturing the execution times.

The datasets will be provided to you on Blackboard.

Project Report

You will write one project report to explain your experiment and your results. Your report should provide a description of the experiment (including figures) and the results of your experiment. It should include the following sections:

- Introduction: describe the goal of the experiment.
- Methods: explain how you conducted the experiment.
- New Algorithm Description: describe the algorithm, including a simple example to help explain how it works (including data values).
- Results: provide the results of your experiment including at least one chart or graph comparing results of three methods in the five algorithms using three different data set sizes.
- Analysis/Discussion: provide a discussion of your findings by comparing your algorithms and recommending one.
- References: cite any references here. You do not have to use a particular reference style.
- Time Log: create a chart showing team members, tasks, and approximate time for each task. Include all task for the project such as coding, running experiments, and writing documents. (See example below).

Team Member	Task	Approximate Time
Michelle Talley	Implemented linear search and BST	1 hour
Noah Kahn	Researched new ADT	30 minutes

Project Presentation

You will give a 5–10-minute oral group presentation using presentation slides. Your presentation should focus on the following:

1. The results of your experiment (comparison graph/chart should be included).
2. A description of your new algorithm.

You may also choose to discuss challenges you faced during the project, driver implementation details, and anything else you want to share. You can reuse material from your experiment report and your new algorithm description report.

Everyone on the team must speak about the project. You will be graded individually on how well you present your portion of the project.

Team Tips

You can do most of your work asynchronously. I recommend meeting as a team as soon as possible to determine your communication channel, assign tasks, and establish milestones and deadlines. Example milestones might include the following:

- Complete all ADTs.
- Complete driver skeleton with file handling and timing. **DO NOT use AI.**
- Integrate ADTs and driver.
- Run experiments.
- Create graph/chart of results.
- Create the project report.
- Create presentation slides.
- Practice presentation.

Teamwork is 20% of your grade for this project, so each person needs to contribute equally to the team. Each team member should use git to push changes to the repository and pull changes from the repository as you are working on the project.

You may distribute the work how you want, but I am providing you this list to give you some ideas. Please adjust as needed.

- Team member 1: Implement the four required ADTs. Carefully test your algorithms on your own BEFORE integrating it into the experiment.
- Team member 2: Research and implement the new ADT. Carefully test your algorithm on your own BEFORE integrating it into the experiment.
- Team member 3: Create the experiment driver skeleton program that reads in the three files and is ready to call the ADTs when ready. Implement timing code for each operation for each ADT. DO NOT leave this team member with all of the work of integration. All team members should participate.
- All team members. Integrate all code to work together (call ADTs for the provided data sets with run times). Run experiment and collect data.
- All team members: Create the project report and presentation slides.

Rubrics

Your grade is based on the team's final product, so every team member is responsible for evaluating the final product against the requirements and the rubrics. You will also be graded individually as well. This project is 10% of your grade for the course and worth 100 points. The points are allocated as follows:

Artifact	Percent of Grade
Source code	30%
Project Report	30%
Team work (Team Report, Team Evaluation, GitHub)	20%
Presentation	20%

Source Code Rubric

	Exceptional	Good	Basic	Unsatisfactory
	(100%~90%)	(89%~70%)	(70%~50%)	(49%~0%)
Readability (20 pts)	The code is exceptionally well organized and very easy to follow.	The code is fairly easy to read.	The code is readable only by someone who knows the intent of the program.	The code is poorly organized and very difficult to read.
Documentation (20 pts)	The documentation is well written and clearly explains what the code is accomplishing and how.	The documentation consists of embedded comment and some simple header documentation that is somewhat useful in understanding the code.	The documentation is simply comments embedded in the code with some simple header comments separating routines.	The documentation is simply comments embedded in the code and does not help the reader understand the code.

Programs that do not compile and execute will get 0 points.

Project Report Rubric

	Exceptional	Good	Basic	Unsatisfactory
	(100%~90%)	(89%~70%)	(70%~50%)	(49%~0%)
Writing (50 pts)	Writing is clear and focused	Ideas are well organized and help the reader move along	Some of the ideas are presented well	Poorly written

	Exceptional	**Good**	**Basic**	**Unsatisfactory**
	Relevant, quality details give the reader important information	Key points are presented	Offers plausible conclusion(s)	Sequencing and pace of topics seems random
	Offers compelling conclusions and helps the reader develop insight into the topic	Coherent conclusions are offered, but does not demonstrate in-depth understanding		Doesn't lead to clear conclusions
Analysis (50 pts)	Algorithms were appropriately selected, described, and classified by complexity.	Algorithms were appropriately selected, described, and classified by complexity.	Most algorithms were appropriately selected, described, and classified by complexity.	Some algorithms were appropriately selected, described, and classified by complexity.
	Data presentation (charts, tables, graphs) is easily understood and interpreted.	Data presentation (charts, tables, graphs) is usually easily understood and interpreted.	Data presentation (charts, tables, graphs) is somewhat easy to understand.	Data presentation (charts, tables, graphs) is unclear.

Teamwork Rubric

	Exceptional	**Good**	**Basic**	**Unsatisfactory**
	(100%~90%)	(89%~70%)	(70%~60%)	(59%~0%)
Teamwork (100 pts)	Contributed to the team by accomplishing significant tasks and investing significant time.	Contributed to the team at an acceptable level by accomplishing sufficient tasks and investing adequate time.	Contributed to the team at a minimal level but could have contributed more by accomplishing more tasks and by investing more time.	Did not contribute to the team in any significant way.

Presentation Rubric

Oral Presentation Rubric

	4—Excellent	3—Good	2—Fair	1—Needs Improvement
Delivery	• Holds attention of entire audience with the use of direct eye contact, seldom looking at notes • Speaks with fluctuation in volume and inflection to maintain audience interest and emphasize key points	• Consistent use of direct eye contact with audience, but still returns to notes • Speaks with satisfactory variation of volume and inflection	• Displays minimal eye contact with audience, while reading mostly from the notes • Speaks in uneven volume with little or no inflection	• Holds no eye contact with audience, as entire report is read from notes • Speaks in low volume and/or monotonous tone, which causes audience to disengage
Content/ Organization	• Demonstrates full knowledge by answering all class questions with explanations and elaboration • Provides clear purpose and subject; pertinent examples, facts, and/or statistics; supports conclusions/ideas with evidence	• Is at ease with expected answers to all questions, without elaboration • Has somewhat clear purpose and subject; some examples, facts, and/or statistics that support the subject; includes some data or evidence that supports conclusions	• Is uncomfortable with information and is able to answer only rudimentary questions • Attempts to define purpose and subject; provides weak examples, facts, and/or statistics, which do not adequately support the subject; includes very thin data or evidence	• Does not have grasp of information and cannot answer questions about subject • Does not clearly define subject and purpose; provides weak or no support of subject; gives insufficient support for ideas or conclusions
Enthusiasm/ Audience Awareness	• Demonstrates strong enthusiasm about topic during entire presentation • Significantly increases audience understanding and knowledge of topic; convinces an audience to recognize the validity and importance of the subject	• Shows some enthusiastic feelings about topic • Raises audience understanding and awareness of most points	• Shows little or mixed feelings about the topic being presented • Raises audience understanding and knowledge of some points	• Shows no interest in topic presented • Fails to increase audience understanding of knowledge of topic

Due Dates and Academic Integrity

The due date is specified on Blackboard. This project is due at the beginning of the class on the due date for all groups regardless of presentation date. Everyone must participate in the presentations and attend all group presentations.

You may copy ADTs source code from other sources, but NOT the experiment driver. Also, provide references for your ADTs in the source code and the project report.

Project Artifacts

The following should be completed by the due date/time specified on Blackboard.

- GitHub: Check in all source code changes to your GitHub repository. Please check your URL using a web browser to verify that your changes have been synced.
- Google Folder: Ensure the project report and presentation slides are in the shared Google folder.
- Blackboard: Source code URL and shared Google Folder URL (contains project report and presentation slides).
- Google Forms: Team member evaluation for yourself and each team member. Evaluations for team presentations and group name. (Link available on Blackboard).