CS5004 Lab05 Report

Rong Huang

Lab 5 part 1 : Linked List of Tasks

1. Reflection (What did you learn?)

In this project, I learned how to design a linked list using object-oriented principles. For instance, I started by defining a Node interface, followed by defining an EmptyNode interface to represent the end of the linked list. Then, I defined additional interfaces related to data storage and operations, such as TaskNode. The most crucial aspect was mastering the usage of recursion to operate on the linked list. For example, for tasks like finding specific conditions, deleting, or searching for specific conditions within the linked list, we can abstract these recursive methods within the Node interface. Then, we can implement these methods recursively within the concrete data node interfaces. Finally, by calling relevant methods in the LinkedList, this design aligns more with the Dependency Inversion Principle and Object-Oriented Design principles. This approach makes the code more modular and easier to maintain.

2. What do you think the advantages are of using a linked list in this application?

Flexibility: With linked lists, we can effortlessly add, remove, or rearrange tasks without worrying about fixed memory allocations. For instance, if we need to insert a new task or rearrange the order, linked lists allow us to do so without the overhead of resizing arrays or shifting elements.

Recursive Operations: Linked lists lend themselves naturally to recursive operations, which simplifies many tasks in our project. For example, when we need to traverse the list to perform certain actions on each task, such as updating task status or counting elements, recursive algorithms make the process more intuitive and easier to implement.

Easy Insertion and Deletion: One of the standout benefits of linked lists is their efficiency in handling insertion and deletion operations. In our project, this means that adding, removing, or updating tasks can be done with minimal overhead. Unlike arrays, where these operations may involve shifting elements, linked lists allow us to directly adjust pointers between nodes, resulting in faster and more efficient operations.

3. Do you think this could have been implemented without a linked list? Explain.

Yes, we can implement it using ArrayList in Java. However, there are also some pros and cons.
Advantages:

- Resizable: ArrayList adjusts its size automatically as we add or remove items, so we don't have to worry about managing its size.
- Easy Access: It's easy to access elements directly by their index, which is handy if we need to quickly find or update specific tasks.
- Simple to Loop Through: We can loop through all the tasks in ArrayList without much hassle, making it easy to perform actions on each task.

Disadvantages:

- Slow Insertions and Deletions: Adding or removing tasks in the middle of the list can be slow because other elements may need to be shifted around.
- Uses More Memory: ArrayList tends to use more memory compared to linked lists because it's based on an internal array structure.
- Potential Performance Issues: If we're constantly adding or removing tasks, ArrayList might slow down over time due to the overhead of resizing the array and moving elements.

4. Extensions (What extensions are you requesting?)

1) add additional functionality: getTaskByID method

This method uses recursion to find a task by its ID. If the task exists, it prints out the detailed information about that task.

2) add additional functionality: addTaskByPriority method

This method uses a loop to insert a task into the list based on its priority. Tasks with higher priority are placed closer to the beginning of the list, ensuring they are handled before lower priority tasks.

3) demonstrate other functionality in the driver

I tested all functionalities of the TaskLinkedList in my driver class including:

add: addTaskAtTop, addTaskAtBottom, addTaskByPriority

changeTaskDate

getTaskByID

list: list all tasks, tasks by priority, and expired tasks

remove: removeTaskAtIndex, removeCompletedTasks, removeTasksByPriority, removeAllTasks

5. Grading Statement (Based on the rubric, what grade do you feel you deserve? Be honest.).

Add the extensions, total 100.

6. Academic integrity statement

I understand that my learning is dependent on individual effort and struggle, and I acknowledge that this assignment is a 100% original work and that I received no other assistance other than what is listed here.

Acknowledgements and assistance received:
Professor Molly, TA Will
Google, StackOverFlow

I did not use generative AI in any form to create this content and the final content was not adapted from generative AI created content.

I did not view content from any one else's submission including submissions from previous semesters nor am I submitting someone else's previous work in part or in whole.

I am the only creator for this content. All sections are my work and no one else's with the exception being any starter content provided by the instructor. If asked to explain any part of this content, I will be able to.

By putting your name and date here you acknowledge that all of the above is true and you acknowledge that lying on this form is a violation of academic integrity and will result in no credit on this assignment and possible further repercussions as determined by the Khoury Academic Integrity Committee.

Name:Rong Huang	Date: 03/13/2024