**CS-230**

**Data Structures**

**Link List Project**

**Professor Mehri**

**Matthew Hinkle**

**3/15/2020**

**ABSTRACT**

In this lab, the students will work with C++ to create code that enters all students with their attributes, courses and with all their attributes, assign four courses to student, , and allows users to append/modify/delete data from the dynamic list. The code created by the students will have a series of statements to insert and pull information out of the link lists the students will create and use to house the data/records. In order to do this the students must do an average amount of research into each C++ statements and inheritance/polymorphism and linked list that accomplishes the student’s task. This lab report will cover the C++ code that completes the projects requirements that consist of displaying or inserting data/records.

**CONTENT**

INTRODUCTION

The purpose of the lab is to teach the students the basics of Data structures in C++. This will provide the students the skills to insert, display, modify, and delete information while using linked list and object orientation in their code so that different ways of manipulating the data is possible.

MATERIALS/ EQUIPMENT

* Computer (Hardware)
* C++ Compiler (Software) ex. DevC++/Visual Studio

RESULTS

Screenshot 1:

This is my screenshot showing that I used the faculty class to create a pure virtual class in order to implement inheritance and runtime polymorphism in my code.

A screenshot of a computer

Description automatically generated

Screenshot 2:

This screenshot shows me implementing polymorphism in my code as I create a pointer from the abstract class and point it to the derived class’s object to use this pointer to insert to the student class.

A screenshot of a cell phone

Description automatically generated

Screenshot 3:

This is the menu of my program, the first being the student options for the student linked list and the professor options to manipulate the link list reserved for those individuals. As seen my options consist of inserting data to the front or back, deleting records, modifying records, viewing the contents of either list, or exiting the program.

A screenshot of a computer

Description automatically generated

Screenshot 4/5:

These screenshots are showcasing my options in action one being inserting data from the front and then viewing the data after it was entered in the next photo. This also shows the attributes used for students, I force any student to assign themselves classes, if none then the user can just insert n/a.

A screenshot of a computer screen

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Screenshot 6:

This is the option 3 being used to delete a record based on the key it has and as seen below a Student Record was deleted.

A screenshot of a cell phone

Description automatically generated

Screenshot 7/8/9:

These screenshots are of entering data from the front first then inserting it from the back showing the functionality of option 2 works.

A screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Screenshot 10/11:

This is the screenshots of using the modification option that alters a record already within the list

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Screenshot 12/13:

These screenshots are showcasing that professor options work as well as they are using similar functions for each option students use. It also shows that I’m using rank to determine what professor they may be.

A screenshot of a cell phone

Description automatically generated

A screenshot of a computer

Description automatically generated

Screenshot 14:

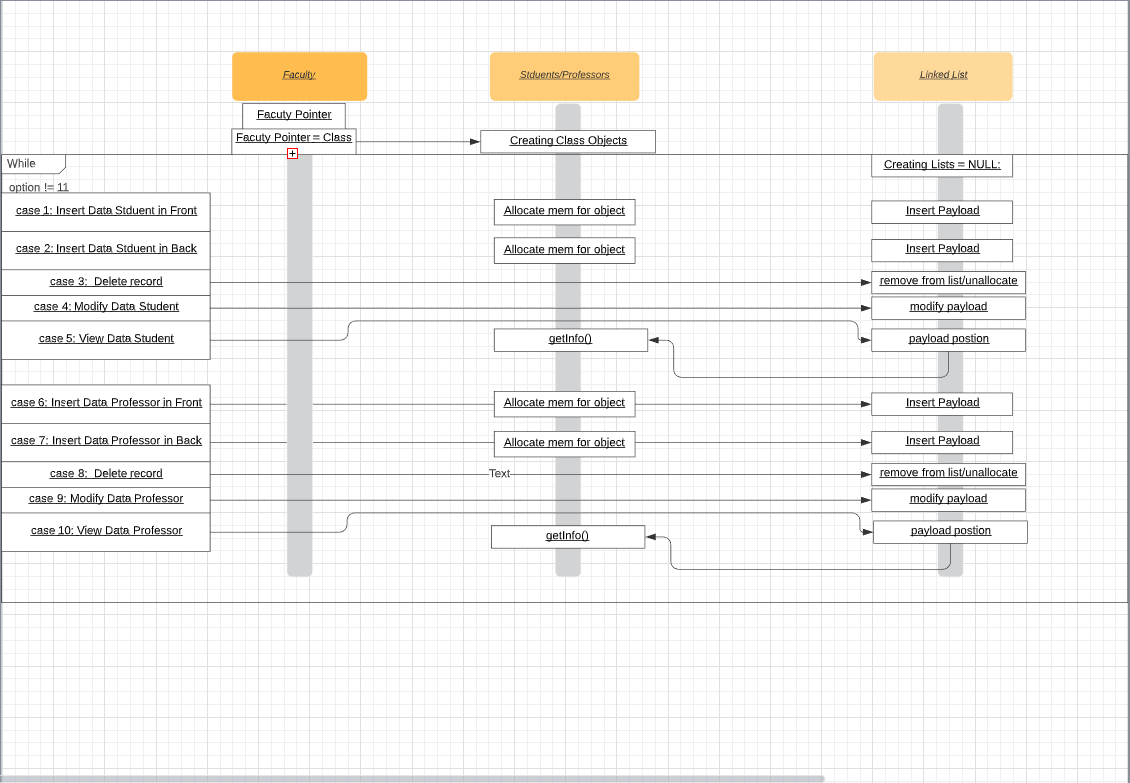
This screenshot is of using the last option of quitting the program, this is done by setting switch inside a do while which will complete the switch cases until the option 11 was entered.

A screenshot of a cell phone

Description automatically generated

Screenshot 14/15/16:

My thought process on making the linked list was that it would be easier to complete different functionality for student and professors if I made exclusive linked list just for their classes. This led me to having the below diagrams and made it easy to implement different attributes and functions without crossing different data, so in other words I normalized the linked list class. After that I simply assigned menu options to the right pointers that added/deleted/modified data.

A close up of a piece of paper

Description automatically generatedA close up of text on a white background

Description automatically generated

ANALYSIS

This lab exercise was perfect in learning the different aspects of C++ and implementing different data structures that structured the data in a specified format so the computer could use the data efficiently. I made the decision to make five classes as it completed the lab in the easiest and most understandable method for myself. Both the student and professor classes were given their own linked list class (Professor\_List Class/ Student\_List Class) in order to separate the different data from crossing over accidentally and made functionality easier. I tried to implement them both in the same list, but it was a lot of verifying code that I would have to enter, so simply copying over the class and changing the names within the class I was able to easily execute both list for professors and students. Linked List opened a lot of doors that wasn’t available without a dynamic list as arrays were a bottleneck for the program in the previous iterations. I implemented polymorphism and inheritance into my code by adding the Faculty abstract class but kept running into issues as the base class was affecting the other header files causing the program not to function anymore. I knew the error, which was that there were no definitions for the functions. I felt that they were not needed as they were being inherited and faculty would never use them itself. After watching a few videos on virtual functions, I learned abstract classes had pure virtual functions meaning that never needed a definition from a faculty.cpp and I applied this aspect of virtual functions to my code and it began to work. This way made it easy to retrieve the desired data whenever I needed them. I ran into some issues with my code where I was trying to add new functionalities to my code. It ended up breaking older functionalities for example my code use to make students insert their info, but it returned as null or empty string values “”. This was due to how I implemented courses into the code as I tried to use another constructor to add this extra information since I couldn’t add to many arguments to a function call. Besides that, after fixing the previous problems with my code I was able to add these additional functionalities without further difficulty.

CONCLUSION

This lab has been beneficial in recapping students in basic skills in C++ and forcing them to be efficient in Linked List and OOP. The skills students have gained from the lab is how to be efficient when storing data in a dynamic list/classes and in presenting the information or records/querying the data for efficient reporting. Research is necessary when working/ orchestrating with new data types, data structures, or dynamic list and plenty of time is required when simplifying code so that it is more compact and faster. Overall, I feel like after completing this lab I am more confident in working with Data structures and C++.

**REFERENCES**

Abstract Class and Pure Virtual function in C -51. (n.d.). Retrieved from https://www.youtube.com/watch?v=SvesRBYu65k

Data Structures. (n.d.). Retrieved from https://www.geeksforgeeks.org/data-structures/

Data Structures in C . (n.d.). Retrieved from https://www.youtube.com/playlist?list=PL8btQaJFYgQWxf-WzUB6WySI60H7FfaG4

Passing by pointer Vs Passing by Reference in C . (2017, May 29). Retrieved from https://www.geeksforgeeks.org/passing-by-pointer-vs-passing-by-reference-in-c/

Polymorphism in C . (2019, October 16). Retrieved from https://www.geeksforgeeks.org/polymorphism-in-c/