

CS2028C Lab 8

Nicholas McClorey Daniel Wood, Kyle Van Blaricom

February 25, 2019

Overview

The purpose of the this lab was to create a linked list. Linked lists demonstrate how pointers can be used to create a list that performs more efficiently than other types of lists. Since all of the data isn't stored in sequential memory, items can be added, removed and inserted without shifting other items or reallocating memory for the entire list. This is ideal for lists that change size unpredictably. It also shows the ability of using data types that contain a pointer to another instance of the class. In our case, the "Node" class contained a pointer to another Node class. This could allow us to create all sorts of data structures that are connected by pointers.

Compiling

This program was compiled using gcc version 6.3.0. The tdm compiler can also be used to compile this program. The command to compile this program is "g++ -std=c++11 main.cpp Part.cpp". This program was compiled using Windows 10.

Task 3

```

Choose:
1 for AddItem: Adds an item to the list
2 for GetItem: Removes a given item from the list and returns it
3 for IsInList: Checks if a given item is in the list
4 for IsEmpty: Checks if the list is empty
5 for Size: Returns the number of elements in the list
6 for SeeNext: Checks and returns the next value of a stored pointer
7 for SeeAt: Checks the value at a given index and stores the location in the pointer
8 for Reset: Resets the pointer for the SeeNext and SeeAt functions to point the the first item in the list
9 for printContents: Prints the contents
10 to exit
1
Enter part number: 89
Enter description: shoes
Enter price: 60
Enter unit of measure: pair
Enter quantity: 13
Choose:
1 for AddItem: Adds an item to the list
2 for GetItem: Removes a given item from the list and returns it
3 for IsInList: Checks if a given item is in the list
4 for IsEmpty: Checks if the list is empty
5 for Size: Returns the number of elements in the list
6 for SeeNext: Checks and returns the next value of a stored pointer
7 for SeeAt: Checks the value at a given index and stores the location in the pointer
8 for Reset: Resets the pointer for the SeeNext and SeeAt functions to point the the first item in the list
9 for printContents: Prints the contents
10 to exit
2
Enter the part number: 89
89 : shoes
Item was removed
Choose:
1 for AddItem: Adds an item to the list
2 for GetItem: Removes a given item from the list and returns it
3 for IsInList: Checks if a given item is in the list
4 for IsEmpty: Checks if the list is empty
5 for Size: Returns the number of elements in the list
6 for SeeNext: Checks and returns the next value of a stored pointer
7 for SeeAt: Checks the value at a given index and stores the location in the pointer
8 for Reset: Resets the pointer for the SeeNext and SeeAt functions to point the the first item in the list
9 for printContents: Prints the contents
10 to exit
|

```

the Quantity On Hand is > 0.
 2. `IsInList()` accepts a date. This returns true if `QuantityOnHand` is > 0. It calculates if the desired date is > Lead Time (added to current date) and the results of that comparison.
 3. Overloads of the `<`, `>`, and `=` operators that compare the `PartNumber` to two `Part` instances.

Complete this before moving on to task 3.

Task 3: Create a test program that has a menu allowing you to test each of the functions of the class (Task 1). This should present the user with a choice of public member functions of your class and ask which the user would like to try. When the user selects a member function, the program will prompt the user for it. For example, the `GetItem` function only requires an `Part`. The `SeeAt` function requires the user to enter a `Part` description, price, unit and optionally a Quantity On Hand.

Test your program. Include a screen shot of some of this testing in your lab report.

Complete this before moving on to task 4.

Task 4: Create a visualization of your list using `AST` (Task 1). Modify your `part` class to include a display method that will output to the screen contents of the class.

- a. This method should write directly to the screen.
- b. This method does not need to display all members of the item, just key members. Your choice of key members in your lab report.

Include a screen shot of the class as much as possible. Include a screen shot of the `main` method to display the list.

- a. This method should write directly to the screen.
- b. This method should walk through the list and display every item in the list. The format of the output is up to you.

Task 4

When printing to the screen we chose to only print the part number and the description. We did this because these two properties exist to identify the part. The other properties are details that don't say anything about the item itself. We also did this because it was concise and minimal.

```
Enter part number: 428
Enter description: baseball
Enter price: 5
Enter unit of measure: balls
Enter quantity: 87
Choose:
1 for AddItem: Adds an item to the list
2 for GetItem: Removes a given item from the list and returns it
3 for IsInList: Checks if a given item is in the list
4 for IsEmpty: Checks if the list is empty
5 for Size: Returns the number of elements in the list
6 for SeeNext: Checks and returns the next value of a stored pointer
7 for SeeAt: Checks the value at a given index and stores the location in the pointer
8 for Reset: Resets the pointer for the SeeNext and SeeAt functions to point the the first item in the list
9 for printContents: Prints the contents
10 to exit
9
{ 45:widget, 78:hairclip, 125:phone, 231:headphones, 428:baseball }
```