Lab 8 Report

Group 25: Randy Hucker and Sam Graler

Instructions:

.cpp and .h files for all Tasks are located in the same visual studio project. You can open and run the visual studio project as normal. The main functions for tasks 3 and 4 were combined into one .cpp file, since the only addition was an extra choice on the menu (the classes have the rest of the code).

<u>Write a lab report including the following information:</u>

a. **A description of the objects/concepts explored in this assignment including why you think they are important to this course and a career in CS and/or Engineering.**

The main concept that we explored in this lab was the double linked list data structure. This is an important concept because linked lists allow quicker insertion and deletion for a sorted list when compared to an array implementation. Additionally, overflow will never occur because the memory is dynamically allocated and pointed to using pointers. Doubly linked lists provide the extra advantage of being able to move both forwards and backwards through the list, because of the addition of the extra previous node pointer. We got more practice using templated classes, as well as exception handling using try catch blocks and custom exceptions.

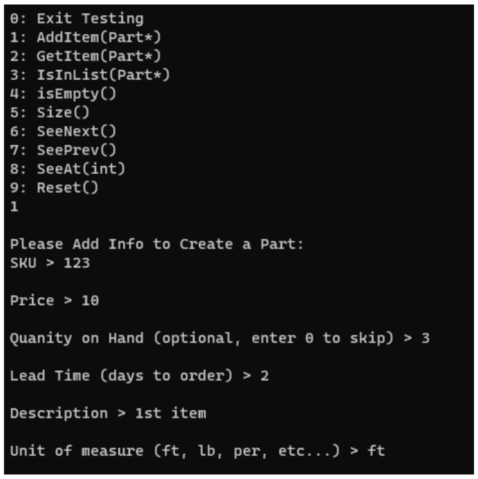
These concepts are important to a career in CS because they are fundamental coding principles, and developing good practices for templates, exception handling, and basic data structures is critical. Additionally, the problem solving skills encouraged by exception handling and templated classes are useful for engineering because they force you to think through the entire problem before rushing into a solution. Giving yourself room to expand within a project saves a lot of work later when new functionality needs to be added.

b. **The sections from each task indicated to be included in the lab report.**

<u>Task 3: Include a screenshot of some of this testing in your lab report.</u>

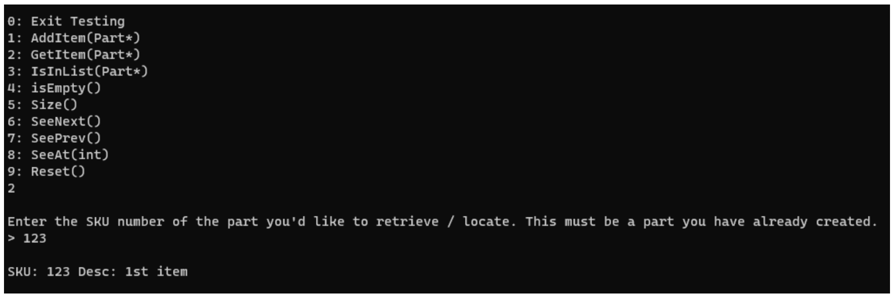
All screenshots for this task were the result of testing on a list with two elements. The first element had an SKU of 123 and description: “1st item”, and the second element had an SKU of 456 and a description of “2nd item” (other data for the elements is arbitrary since the basic print function only shows SKU and desc).

AddItem



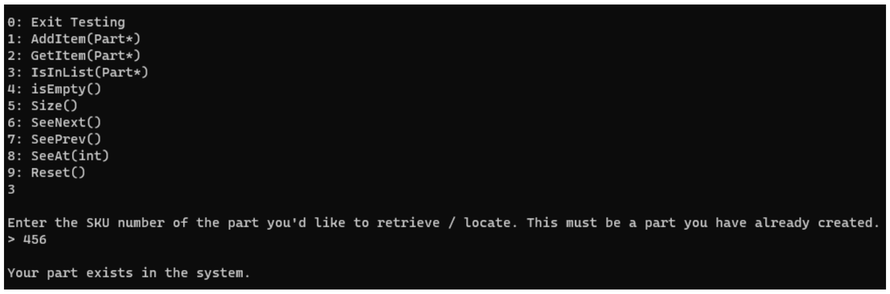
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
1

Please Add Info to Create a Part:
SKU > 123

Price > 10

Quanity on Hand (optional, enter 0 to skip) > 3

Lead Time (days to order) > 2

Description > 1st item

Unit of measure (ft, lb, per, etc...) > ft
```

GetItem: (After this is called, a new list was created to match the conditions described above because the 1st item got removed)



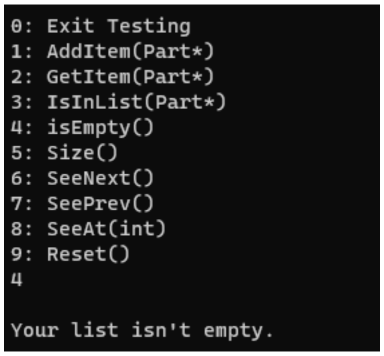
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
2

Enter the SKU number of the part you'd like to retrieve / locate. This must be a part you have already created.
> 123

SKU: 123 Desc: 1st item
```

IsInList



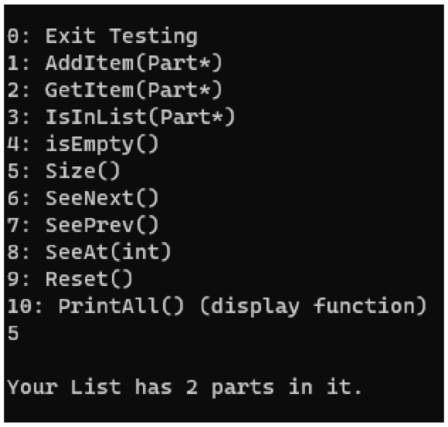
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
3

Enter the SKU number of the part you'd like to retrieve / locate. This must be a part you have already created.
> 456

Your part exists in the system.
```

isEmpty



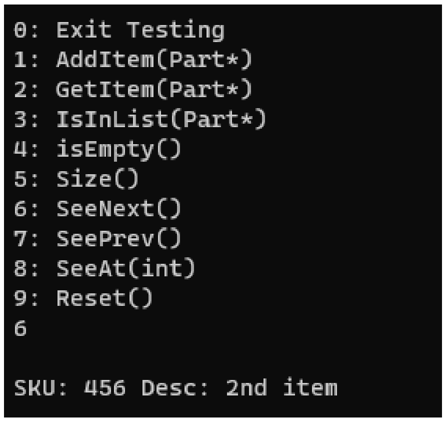
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
4

Your list isn't empty.
```

Size



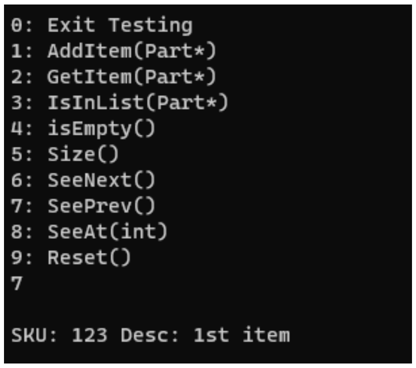
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
10: PrintAll() (display function)
5

Your List has 2 parts in it.
```

SeeNext



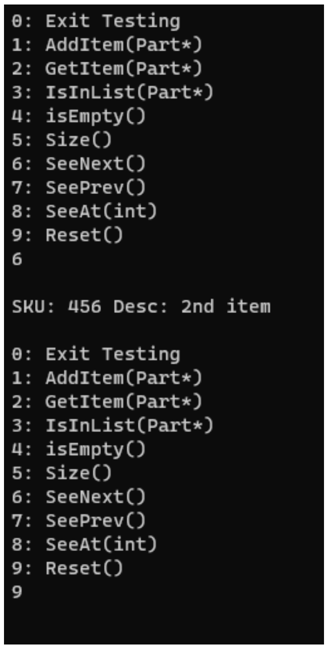
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
6

SKU: 456 Desc: 2nd item
```

SeePrev

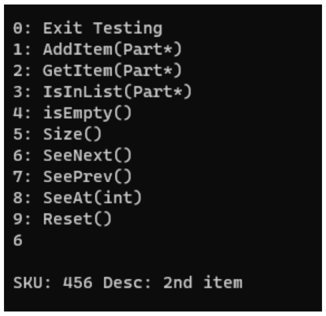


```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
7

SKU: 123 Desc: 1st item
```

Reset: (SeeNext was called to move current from the 1st item to the 2nd item, then reset was called. SeeNext was called again to prove the current pointer moved back to the first item.)



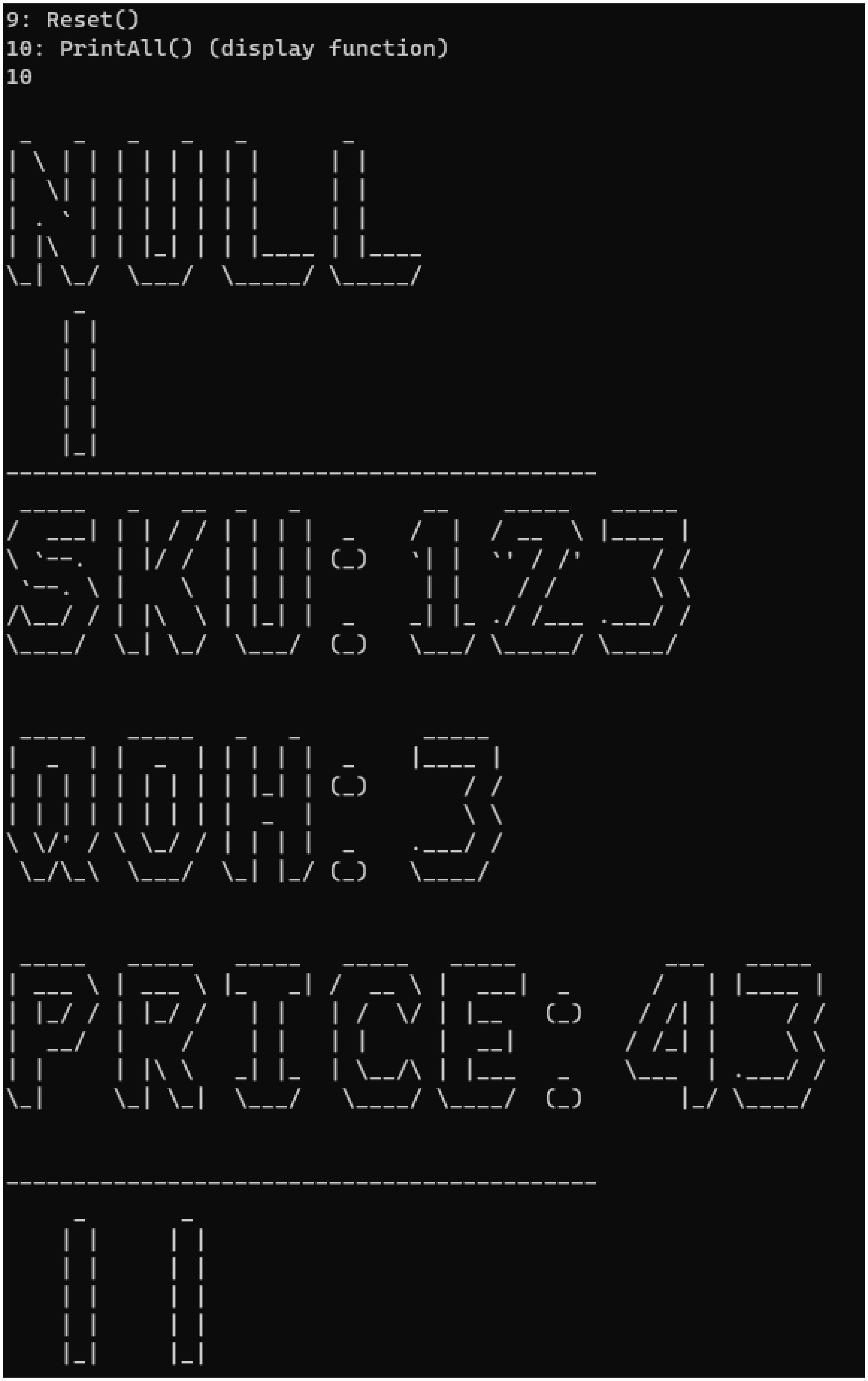
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
6

SKU: 456 Desc: 2nd item

0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
9
```



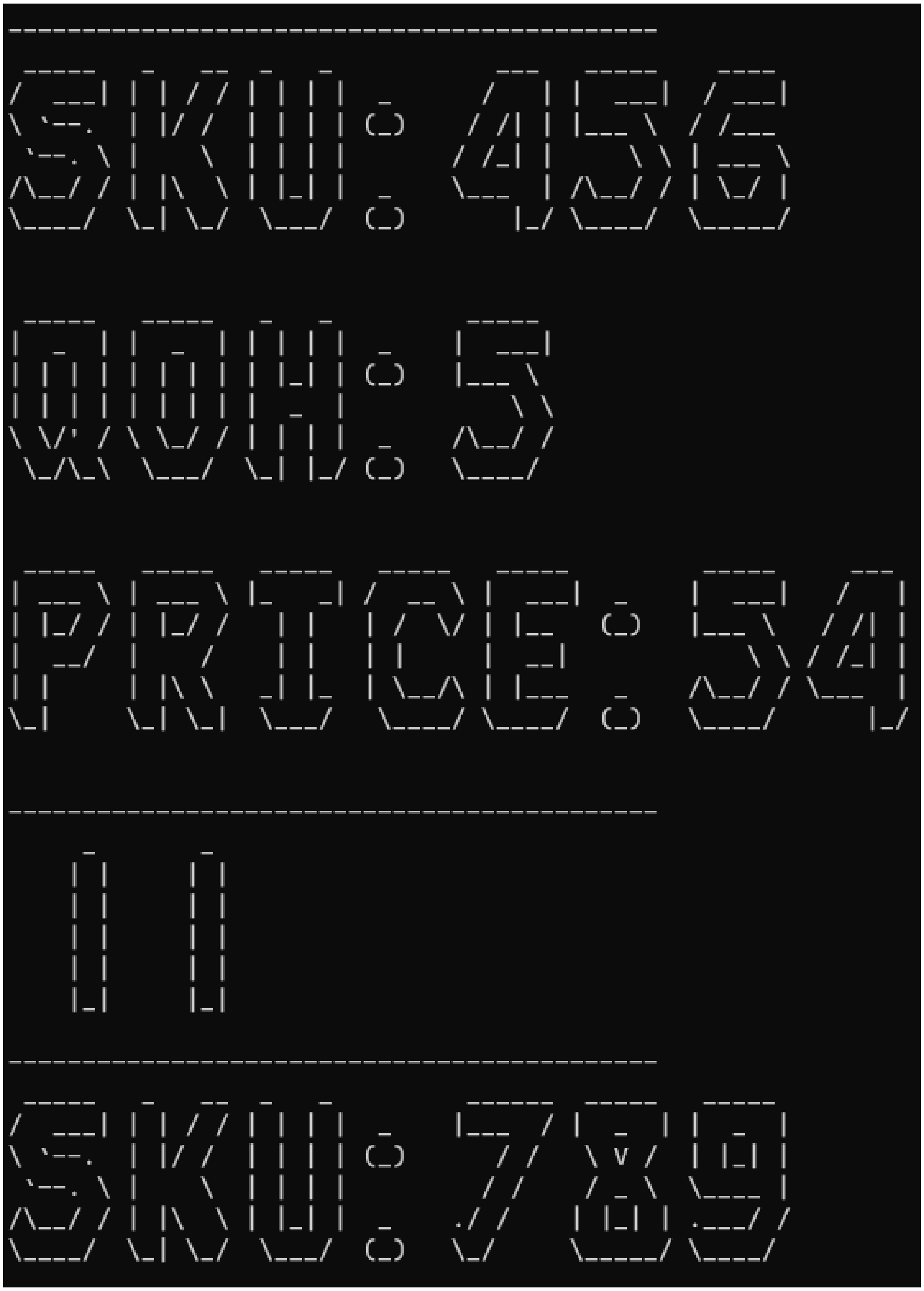
```
0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
6

SKU: 456 Desc: 2nd item
```

Task 4:Include a screenshot showing the results of this new method with at least 4 items in your list.

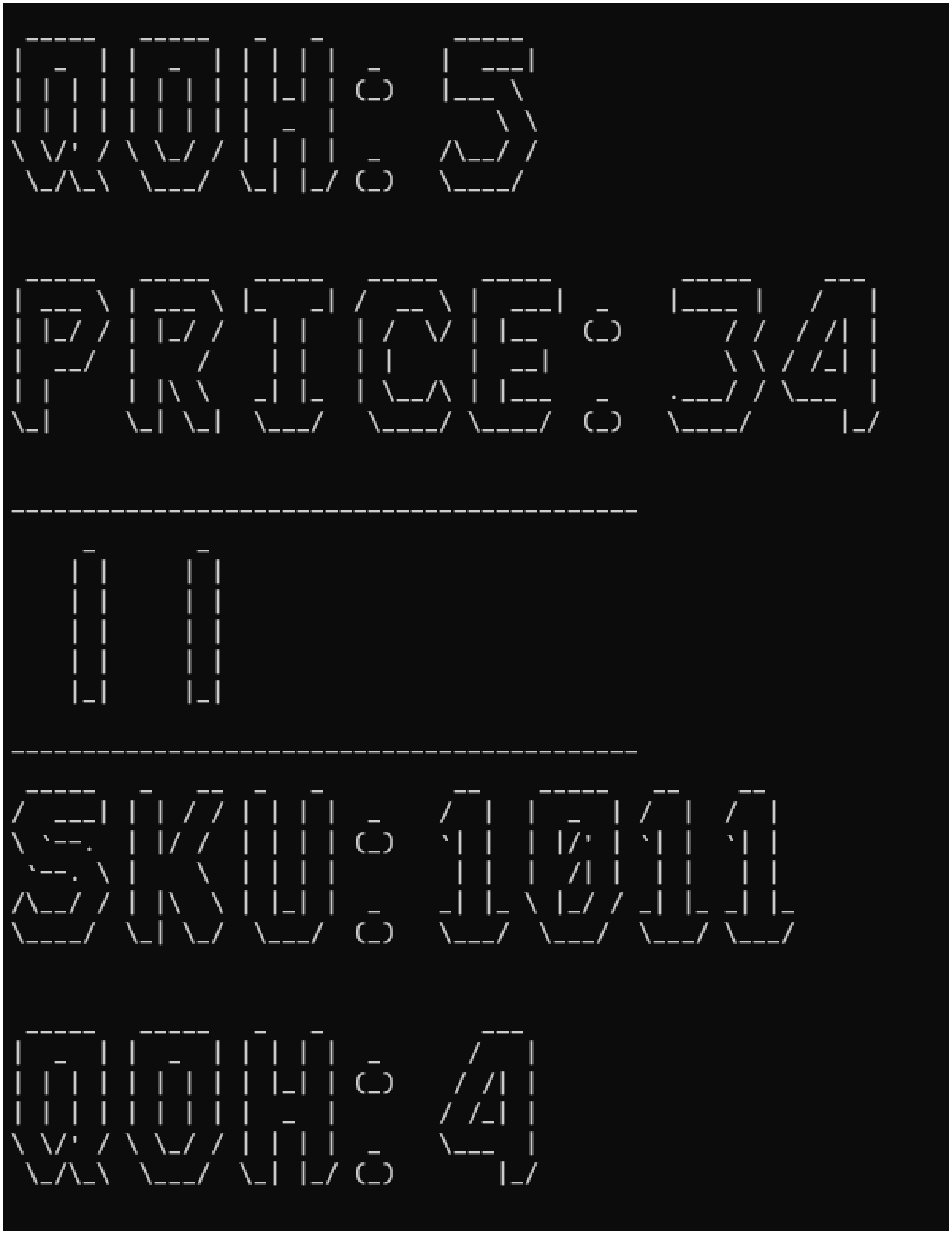
Our display function uses an ascii art function created by Lord Hypersonic (https://lordhypersonic.blogspot.com/2019/02/c-ascii-art-generator.html). This function takes a string as an input, and writes the ascii art for that string to the screen. We implemented this function along with our own class member functions to create the display function. The output was split into multiple screenshots because the ascii art function generates a rather large output. We chose to display the SKU, quantity on hand, and price for each part in the list. We also decided to use the | character to represent a pointer to the next and previous nodes, with the head and tail of the list each having a pointer to null.



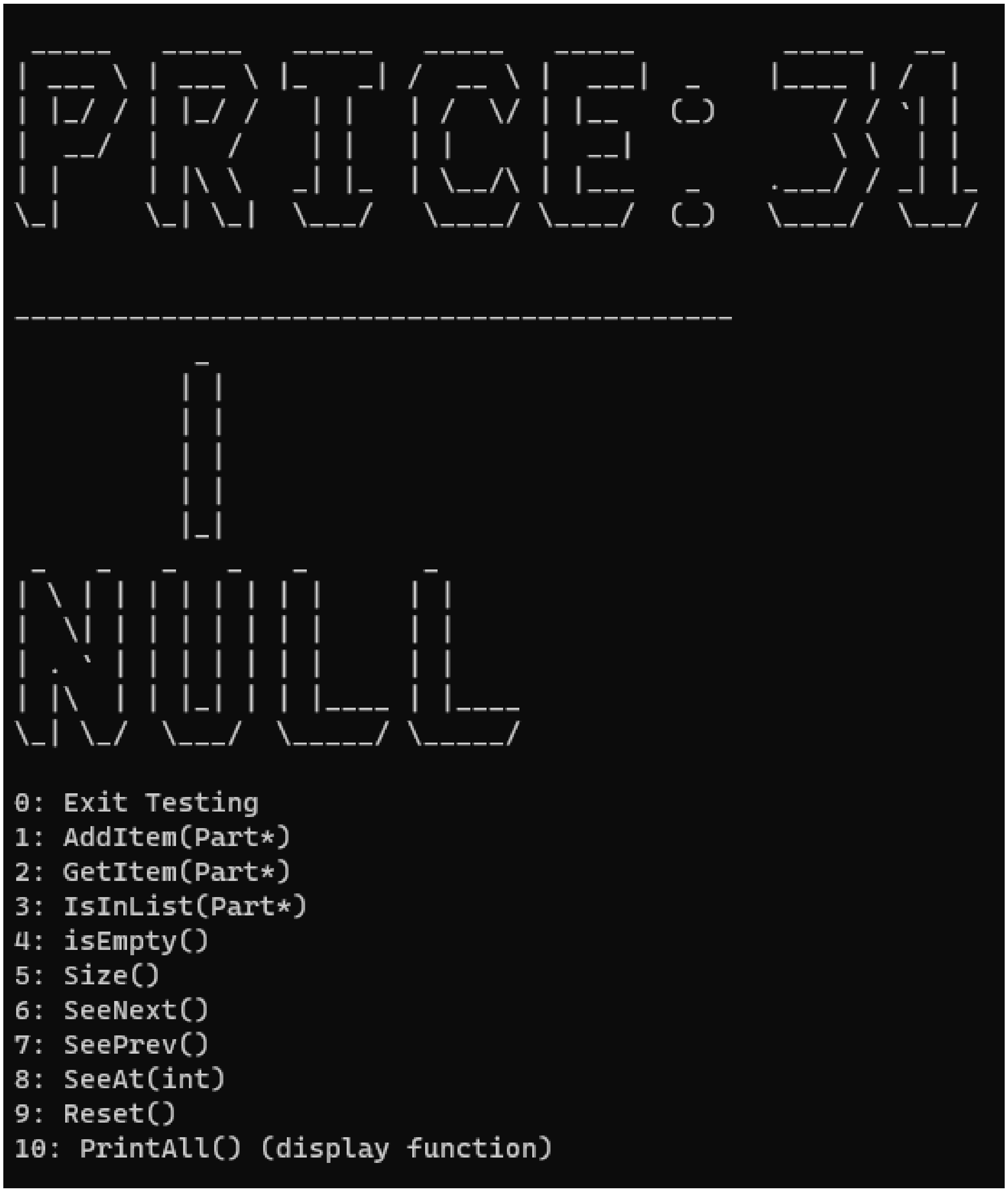
```
9: Reset()
10: PrintAll() (display function)
10

 _   _   _   _   _        _
| \ | | | | | | | |      | |
|  \| | | | | | | |      | |
| . ` | | | | | | |      | |
| |\  | | |_| | | |____  | |____
\_| \_/  \___/  \_____/  \_____/
    _
   | |
   | |
   | |
   | |
   |_|
--------------------------------------------

 _____   _   __  _   _          __    _____   _____
/  ___| | | / / | | | |  _     /  |  / __  \ |____ |
\ `--.  | |/ /  | | | | (_)    `| |  `' / /'     / /
 `--. \ |    \  | | | |         | |    / /       \ \
/\__/ / | |\  \ | |_| |  _     _| |_ ./ /___ .___/ /
\____/  \_| \_/  \___/  (_)    \___/ \_____/ \____/


 _____   _____   _   _           _____
|  _  | |  _  | | | | |  _      |____ |
| | | | | | | | | |_| | (_)         / /
| | | | | | | | |  _  |             \ \
\ \/' / \ \_/ / | | | |  _      .___/ / 
 \_/\_\  \___/  \_| |_/ (_)     \____/


 _____   _____   _____   _____   _____         ___   _____
| ___ \ | ___ \ |_   _| /  __ \ |  ___|  _    /   | |____ |
| |_/ / | |_/ /   | |   | /  \/ | |__   (_)  / /| |     / /
|  __/  |    /    | |   | |     |  __|      / /_| |     \ \
| |     | |\ \   _| |_  | \__/\ | |___   _  \___  | .___/ /
\_|     \_| \_|  \___/   \____/ \____/  (_)     |_/ \____/


--------------------------------------------
    _       _
   | |     | |
   | |     | |
   | |     | |
   | |     | |
   |_|     |_|
```



```
------------------------------------------
 _____   _   __  _   _          ___    _____   ____
/  ___| | | / / | | | |  _     /   |  |  ___| / ___|
\ `--.  | |/ /  | | | | (_)   / /| |  |___ \ / /___
 `--. \ |    \  | | | |      / /_| |      \ \| ___ \
/\__/ / | |\  \ | |_| |  _   \___  |  /\__/ /| \_/ |
\____/  \_| \_/  \___/  (_)      |_/  \____/ \_____/


 _____   _____   _   _        _____
|  _  | |  _  | | | | |  _   |  ___|
| | | | | | | | | |_| | (_)  |___ \
| | | | | | | | |  _  |          \ \
\ \/' / \ \_/ / | | | |  _   /\__/ /
 \_/\_\  \___/  \_| |_/ (_)  \____/


 _____   _____   _____   _____   _____        _____     ___
| ___ \ | ___ \ |_   _| /  __ \ |  ___|  _   |  ___|   /   |
| |_/ / | |_/ /   | |   | /  \/ | |__   (_)  |___ \   / /| |
|  __/  |    /    | |   | |     |  __|           \ \ / /_| |
| |     | |\ \   _| |_  | \__/\ | |___   _   /\__/ / \___  |
\_|     \_| \_|  \___/   \____/ \____/  (_)  \____/      |_/


------------------------------------------
     _       _
    | |     | |
    | |     | |
    | |     | |
    | |     | |
    |_|     |_|
------------------------------------------
 _____   _   __  _   _         ______  _____  _____
/  ___| | | / / | | | |  _   |___  / |  _  ||  _  |
\ `--.  | |/ /  | | | | (_)     / /   \ V / | |_| |
 `--. \ |    \  | | | |        / /    / _ \ \____ |
/\__/ / | |\  \ | |_| |  _   ./ /    | |_| |.___/ /
\____/  \_| \_/  \___/  (_)  \_/      \_____/\____/
```



```
 _____  _____  _   _        _____ 
|  _  ||  _  || | | |  _   |  ___|
| | | || | | || |_| | (_)  |___ \ 
| | | || | | ||  _  |          \ \
\ \/' /\ \_/ /| | | |  _   /\__/ /
 \_/\_\ \___/ \_| |_/ (_)  \____/ 


 _____ ______  _____  _____  _____         _____    ___ 
| ___ \| ___ \|_   _|/  __ \|  ___|  _    |____ |  /   |
| |_/ /| |_/ /  | |  | /  \/| |__   (_)       / / / /| |
|  __/ |    /   | |  | |    |  __|            \ \/ /_| |
| |    | |\ \  _| |_ | \__/\| |___   _    .___/ /\___  |
\_|    \_| \_| \___/  \____/\____/  (_)   \____/     |_/


-----------------------------------
   _     _ 
  | |   | |
  | |   | |
  | |   | |
  | |   | |
  |_|   |_|
-----------------------------------
 _____  _   __ _   _          __   _____  __   __  
/  ___|| | / /| | | |  _     /  | |  _  |/  | /  | 
\ `--. | |/ / | | | | (_)    `| | | |/' |`| | `| | 
 `--. \|    \ | | | |         | | |  /| | | |  | | 
/\__/ /| |\  \| |_| |  _     _| |_\ |_/ /_| |__| |_
\____/ \_| \_/ \___/  (_)    \___/ \___/ \___/\___/


 _____  _____  _   _          ___ 
|  _  ||  _  || | | |  _     /   |
| | | || | | || |_| | (_)   / /| |
| | | || | | ||  _  |      / /_| |
\ \/' /\ \_/ /| | | |  _   \___  |
 \_/\_\ \___/ \_| |_/ (_)      |_/
```



```
______  ______   _____   _____   _____        _____   __  
| ___ \ | ___ \ |_   _| /  __ \ |  ___|  _   |____ | /  | 
| |_/ / | |_/ /   | |   | /  \/ | |__   (_)      / / `| | 
|  __/  |    /    | |   | |     |  __|           \ \  | | 
| |     | |\ \   _| |_  | \__/\ | |___   _   .___/ / _| |_
\_|     \_| \_|  \___/   \____/ \____/  (_)  \____/  \___/


--------------------------------------------

           _
          | |
          | |
          | |
          | |
          |_|

 _   _   _   _   _       _     
| \ | | | | | | | |     | |    
|  \| | | | | | | |     | |    
| . ` | | | | | | |     | |    
| |\  | | |_| | | |____ | |____
\_| \_/  \___/  \_____/ \_____/

0: Exit Testing
1: AddItem(Part*)
2: GetItem(Part*)
3: IsInList(Part*)
4: isEmpty()
5: Size()
6: SeeNext()
7: SeePrev()
8: SeeAt(int)
9: Reset()
10: PrintAll() (display function)
```