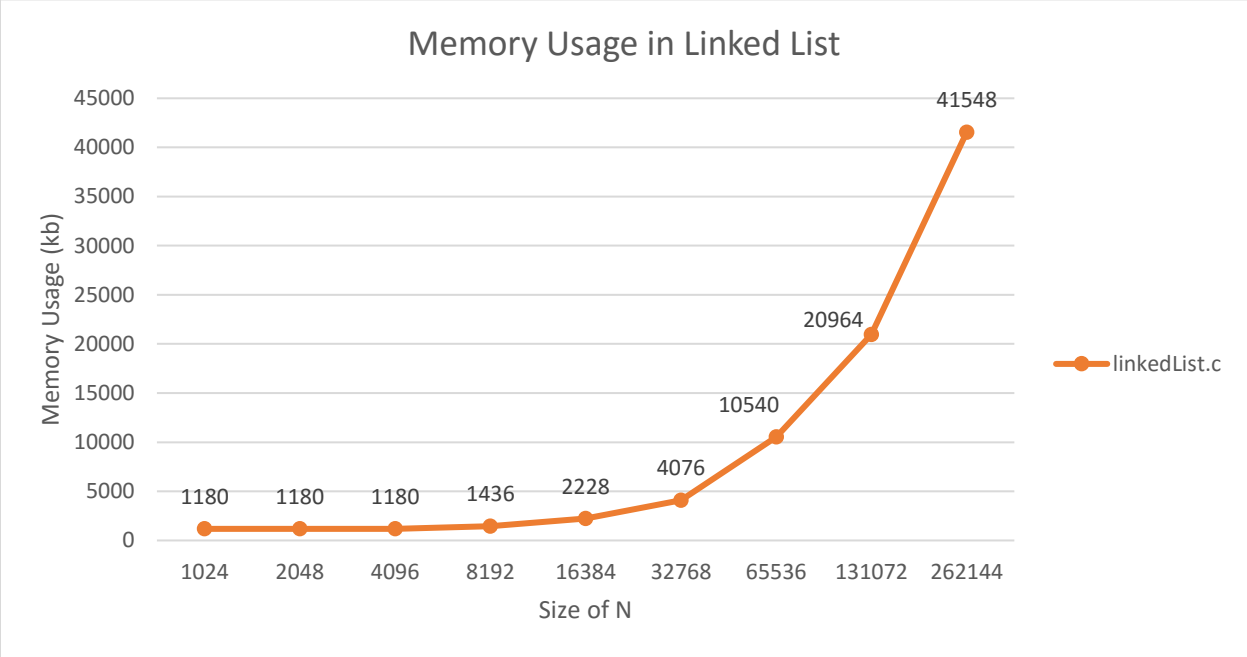
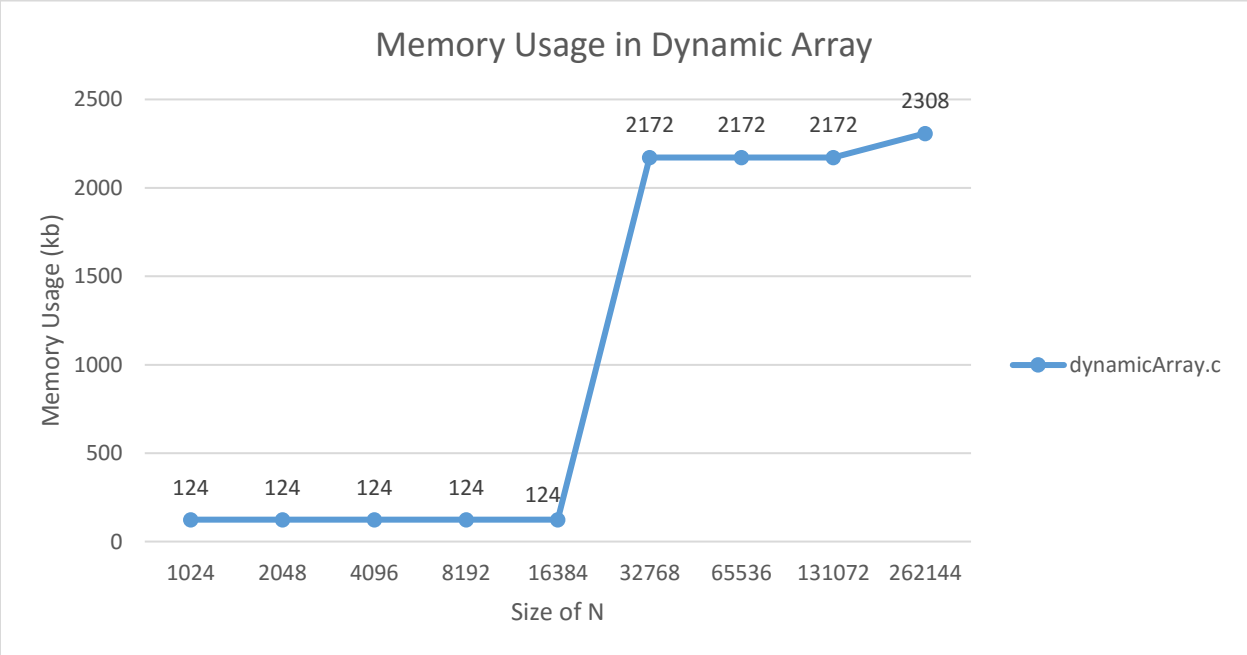
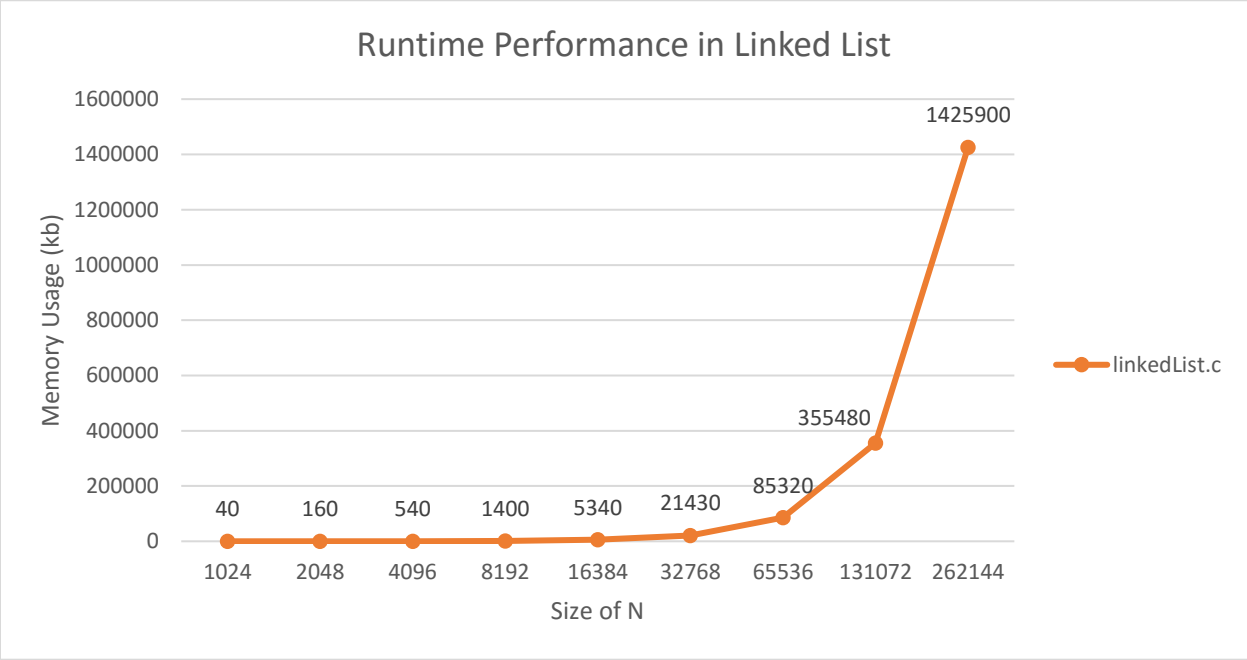
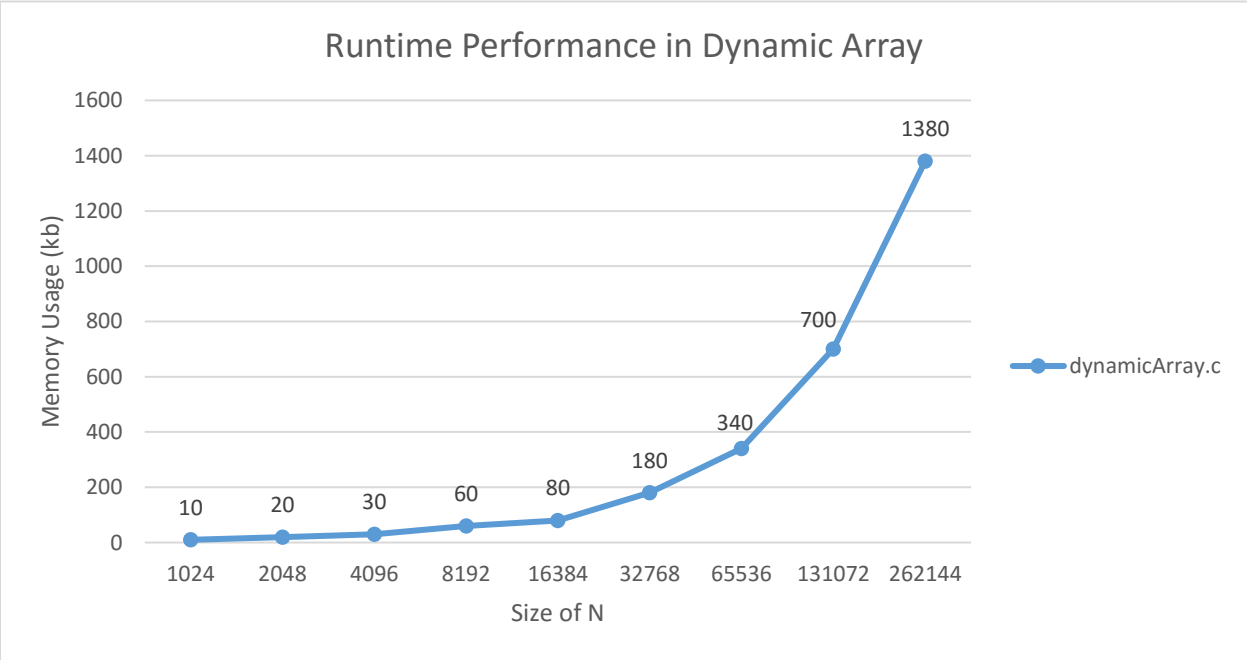
Name: John Carrabino
ONID: carrabij@oregonstate.edu
Term: Fall 2016
Course: CS 261

# Assignment 3 Problem 2:
# Linked List vs Dynamic Array Performance




Runtime Performance in Linked List
Memory Usage (kb)
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
40
160
540
1400
5340
21430
85320
355480
1425900
1024
2048
4096
8192
16384
32768
65536
131072
262144
Size of N
linkedList.c


Runtime Performance in Dynamic Array
Memory Usage (kb)
1600
1400
1200
1000
800
600
400
200
0
10
20
30
60
80
180
340
700
1380
1024
2048
4096
8192
16384
32768
65536
131072
262144
Size of N
dynamicArray.c

| Size of N from 2^10 to 2^18 | linkedList.c | | dynamicArray.c | |
|---|---|---|---|---|
| | Mem Usage (kb) | Time (ms) | Mem Usage (kb) | Time (ms) |
| 1024 | 1180 | 40 | 124 | 10 |
| 2048 | 1180 | 160 | 124 | 20 |
| 4096 | 1180 | 540 | 124 | 30 |
| 8192 | 1436 | 1400 | 124 | 60 |
| 16384 | 2228 | 5340 | 124 | 80 |
| 32768 | 4076 | 21430 | 2172 | 180 |
| 65536 | 10540 | 85320 | 2172 | 340 |
| 131072 | 20964 | 355480 | 2172 | 700 |
| 262144 | 41548 | 1425900 | 2308 | 1380 |

## Questions:

1. **Which of the implementations uses more memory? Explain why.**

   The Linked List implementation uses significantly more memory then the Dynamic Array. I believe this is due to the fact that each value stored in the Linked List requires its own node ("link"), opposed to the Dynamic Array, which stores all of its elements in a single array. Therefore, each time a new element is added to the Linked List it must allocate memory for a new node to story that element in. Unlike the Linked List, the Dynamic Array is able to continuously store values until its array is full, then the array will double in size. This causes the overall memory requirements of the Dynamic Array to be much lower because it only needs to reallocate memory each time the array is full (doubles in size).

2. **Which of the implementations is the fastest? Explain why.**

   Once again the Dynamic Array implementation is much faster than that of the Linked List. I believe this goes hand-in-hand with how much higher the memory requirements arefor the Linked List implementation. Since the Dynamic Array only needs to allocate memory when the array is full, the amount of calls between doubling the array also double in size. However, with the linked list, memory allocation is required for each item added to the list. Along with that, when memory is allocated for an array it is done so in consecutive memory addresses, but with a linked list each node has a pointer to the next/previous nodes and they are not located at consecutive memory addresses, which in turn could also affect runtime performance.

3. **Would you expect anything to change if the loop performed remove() instead of contains()? If so, why?**

Yes, because unlike the dynamic array, when a link list removes an item it only has to perform that one operation, which involves changing the address of the pointers pointing to the node to be removed and freeing the memory allocated for the removed node, giving it a big-o complexity of O(1). However, when a dynamic array removes an item that is not located at the end of the array it has to re-index all the elements that occur after it. This means that if the first item is removed from a dynamic array, then it will have to perform N operations (in order to move every element in the array down one index position), which gives the remove() function for the dynamic array a worst-case complexity of O(N), which is much worse than that of the linked list.