CS 2401 Fall 2021

**Lab 4: Dynamic Arrays and The Big 3**

**Due: Friday 9/24 at 11:59 PM**

Type your answers on this sheet making sure to list the question numbers.

1. What do you see?

2

4

6

8

10

12

14

2

4

6

5

10

15

14

Total bytes allocated = 60

1. Is this a problem and why?

Yes, this is a problem. Since the main sets the second class equal to the first, they both reference the same memory locations. Therefore, when N2 deletes four items and then overwrites them, N2 overwrites memory shared with N1. Meaning that N1 displays altered data.

1. What caused this to happen?

The pointer inside N1, (“data”), was copied directly into N2, meaning that N2 accesses the same memory as N1.

1. What do you see?

2

4

6

8

10

12

14

2

4

6

8

10

12

14

Total bytes allocated = 60

1. Is this different from what you saw before?

Yes, the data displayed is the same for both display function calls.

1. What caused this to happen? What is different?

When N2 is created, and with the addition of an overloaded assignment operator, N2 is given a pointer to a section of memory not used by N1. Meaning that when N2 deletes and rewrites the last four array slots, N1’s array is unchanged.

1. Write down the addresses and the byte output

0xa68da8

0xa6a158

0xa6b508

0xa6c8b8

0xa6dc68

Running took 0 minutes.

Total bytes allocated = 2060

1. How many bytes apart are the memory cells?

5040 bytes

1. What addresses did you see this time and how far apart?

0x748da8

Data removed.

0x74a158

Data removed.

0x74b508

Data removed.

0x74c8b8

Data removed.

0x74dc68

Data removed.

5040 bytes apart.

1. What is the byte count?

60 bytes

1. Explain why adding the destructor resulted in this different behavior.

The program creates five items but deallocates the previous item when a new one is declared, meaning that the program does allocate additional memory every function iteration. Preventing the program from throwing a bad\_alloc call.