Week 9 - Memory Debugging

Daniel Alyoshin

CSCA48 - TUT002

March 13, 2026

Content Covered so Far

- C programming
- Memory model
- Arrays and strings
- Pointers
- CDTs & ADTs
- Linked lists
- Testing & Debugging
- Binary Trees & BSTs
- Big O & Complexity Analysis
- Recursion

Memory Debugging Tools

- Final stage of the design/implement/debug cycle.
- Useful for checking various memeory issues:
  - Array indexing issues
  - Pointer problems
  - Memory leaks
  - Uninitialized memory
  - etc.

Your First Task

csca48.alyoshin.dev

Your Second Task

1. Create a new **dynamic array** of length 100 and of type either `int` or `float` using `malloc()` within the same program.

2. Compute the sum of the entries in the array (you should not initalize the values, this is why we use `malloc()`).

3. Run the program with Dr. Memory and see what happens...

Your Second Task

1. Create a new **dynamic array** of length 100 and of type either `int` or `float` using `malloc()` within the same program.

2. Compute the sum of the entries in the array (you should not initalize the values, this is why we use `malloc()`).

3. Run the program with Dr. Memory and see what happens...

4. Write a loop to fill the entries in the array *beyond the limits* of the array. Run the **code** and see what happens.

Your Second Task

1. Create a new **dynamic array** of length 100 and of type either `int` or `float` using `malloc()` within the same program.
2. Compute the sum of the entries in the array (you should not initalize the values, this is why we use `malloc()`).
3. Run the program with Dr. Memory and see what happens...
4. Write a loop to fill the entries in the array *beyond the limits* of the array. Run the **code** and see what happens.
5. Run the program with Dr. Memory again and see what happens...

Suggestions

- Start using these tools (Dr Memory, Valgrind, or the options discussed on piazza) on your assignments to ensure you avoid losing marks.
- Try using it on Exercise 8 and Assignment 2.

Reminders

- Each student to submit their own `.c` file which can be the same as that submitted by members of their group.
- Ensure your file is named **exactly** as instructed on Quercus.
- You are **not** supposed to keep working on this, you will not be marked on correctness so as long as you submit work that shows you were engaged during this tutorial you will get full marks.