Dynamic Array

Explanation and theory.

Difference between dynamic and static array

- Static array are created onto a stack, and do not require memory management.
- They are destroyed once the function they are in is completed. (Function is no longer in scope).
- Have a predefined size, ie int a[10];
- Dynamic array (not to be confused with dynamically allocated array), have *infinite size, it grows as more space is needed. (Think of std::vector in C++, or ArrayList in Java)

Why dynamic array?

Reasons to use a dynamic

- Not limited to a defined array size.
- Insertion and removal functionality.
- Still has O(1) accessing/indexing time complexity compared to a regular array.

Cons to using a dynamic array

- Can potentially use up a lot of excess space.
- Removal and insertion at certain indexes are costly operations.
- Better to use a regular static array when the needed size is known.

Implementation Plan

- We will use C++ in this video (implementation theory is pretty much the same with Java or another OO language)
- We will be holding integer data types.
- We will double the capacity of the array when the capacity is reached.
- We will shrink the capacity by half when the number of elements in the array reaches floor(capacity/4).
- We will include the following methods, add, remove, searching, accessing/indexing, printing and getting the current number of elements or “size”.