**Data Structures (Course 2 Week 2 Quiz) (100% on first attempt)**

1. So we have 2 functions – one is Pushback () which does reallocation, and PopBack() that doesn’t do reallocation.

At the end of 24 pushback and 24 popback operations, the maximum possible final capacity would be 24 pushback operations in a row – which would mean that the array has a capacity of 32 (25) , while the minimum size would be to alternate a pushback and a popback, which would result in a capacity of 1 (20).

**Maximum: 32, Minimum: 1**

1. Now popback does a resize when the dynamic array has a size of capacity/2. Give an example of n operations starting from an empty array that require O(n2) copies.

**Let be a power of 2. Add elements, then alternate times between doing a**

**PushBack of an element and a PopBack.**

Once we’ve added elements, the dynamically allocated array is full (size capacity . When we add 1 more element we resize and copy elements and now size is When we then remove an element with popback, we reallocate the dynamically allocated array and copy elements. So each of the final operations costs copies, for a total of moves, or

1. Now popback does reallocation when size <= capacity/4. We want to consider the worst case sequence of any pushback and popback operations, starting with an empty dynamic array.

What potential function would work best to show an amortised O(1) cost per operation?

I am guessing

When doing a pushback with resize, capacity/2 – size would be negative, so phi of h would be the first argument.

When doing a popback with resize, 2\*size – capacity where size = capacity /4 would be negative, so phi of h would be the 2nd argument.

1. Answers:

* O(1) because the sum of the costs of all PopMany operations in a total of n operations is O(n).
* O(1) because we can place one token on each item in the stack when it is pushed. That token will pay for popping it off with a PopMany.
* O(1) if we define = size.

Since Ci = i , let sizei-1 = k, and sizei = k-i

AMC = i + k – i – k = 0