Name : Sujeewa Sandeepa Kularathna

Date : 2021-07-18

Data structures – part 02

Dynamic and Static Arrays

A static array is a fixed container containing n elements which each one can be referenced with a number from the range [0, n-1].

Where a static array is used,

1) Storing and accessing sequential data
2) Temporarily storing objects
3) Used as buffers to store information from input output routines
4) As lookup tables and inverse lookup tables
   - A really good way to retrieve data from a lookup table if you know where everything is.
5) Used to return multiple values from a function.
   - This is a workaround, when we need to return multiple values but the language only supports one return value. So the workaround is we return a pointer or a reference to an array which contains all the return values.
6) Used in dynamic programming to cache answers to subproblems.

Complexity

Access times for Static and Dynamic arrays are a constant(O(1)), because arrays are indexable.
(Indexable means each capsule can be referenced with a number.)

Searching an array can take up to the linear time (O(n)) because the element we are looking for can be in the opposite end that we are starting to search from. Or in the worst case if the element we are looking for doesn't exist. Then we will have to go through each and every element.

And inserting, appending and deletion are not possible to be done for static arrays. Because those are containers with a fixed size.

But for Dynamic arrays it's very possible, to do those.

So for insertion in Dynamic arrays it can take up to linear time, because in insertion when we insert one element, we are copy pasting and moving all the other elements one step to the right.

Okay so appending on Dynamic arrays I didn't understand the first few times I tried. But now I think I understand it enough to write it here thanks to a good friend of mine. So appending is insertion at the end of the dynamic array right, this is the case so if the array is not full and we have free space in the end, appending one more element is going to be very simple and it will take only a constant time, in Big-O notation O(1).
But it becomes a little bit annoying if the array is already full. Then we will have to create another array and paste all the elements in to the new array and then append the new element. So now in this case it's going to take a linear time, so in Big-O notation it is O(n).
Hope you understood it.

Deletion is linear for the same reason that insertion is linear.

Operations on dynamic arrays

Dynamic arrays can grow and shrink in size as needed.

So how do the dynamic arrays are made? Well typically those are done with a static array.

1) First we need to create an static array with an initial capacity.
2) Add elements to the underlying static array, keeping track of the number of elements.
3) If adding another element will exceed the capacity, then create a new static array with twice the capacity and copy the original into it.