DSA Homework 1

Adi Ramachandran | 02/07/2021

Question 1:

Each list takes up a continuous segment in memory.

With concatenation, assuming the arrays are able to be concatenated (they are of objects of the same type, etc), you must find a new place in memory (O(1) time) to store the concatenated array, and you must copy over every item from the first list, and then every item from the 2nd list right afterwards. If there are A items in the first list and B items in the second list, then the time complexity will be `O(A+B + 1)`. Removing the constant operation results in a final time complexity of `O(A+B)`.

Question 2:

a)

A reminder that unimodal means an increasing followed by decreasing sequence. Perform a binary search for the maximum element - AKA, use the property that it is sorted, and we can always tell which 'half' of the array we're in when given an element and it's surrounding elements. We can also use the fact that the largest item has a smaller items on both the left and right.

----- pseudocode -------

```
define a top and bottom and middle index.
while not (end condition)

    - redefine a top and bottom index based on the elements surrounding the
middle index. Choose if the bottom index should shift up to the current
middle index, or if the top index should shift down to the current middle
index.
    - recompute the middle index, it's in the middle of the top and bottom
(rounded down for odd numbers)
    - compute the end condition

when you exit the while loop, we are left with the largest item as the
middle index, and we return it
```

Why does it run in `O(log(N))` time: with every step it cuts in half the number of elements it needs to search through. It takes `log2(N)` iterations to return a value. Thus it's complexity is `O(log(N))`.

b)

Loop invariant - the condition that holds true every iteration is that imid is in the middle of the top and bottom index, and the top index or bottom index will always shift to move imid closer to the highest element in the array. THe end condition, while the function is looping, will always also be false - being that the imid is greater than the item to its left and the item to the right (as long as there exists a value on the left and right.)

- Initialization - itop and ibtm are initialized as the highest element and lowest in the array, and imid is in the middle
- Maintenance - with a conditional we shift itop or ibtm to the value of imid and recalculate imid.
- Termination - When the exit_case is met, the while loop is terminated and the value of the array at imid index is returned.

Question 3:

Answer to question 3 can be found in `hw1` file. Note that I wasn't able to get `pytest` working and that I just ran manual tests, instead of usign the `assert` statement. I would love to spend more time wiht a ninja to get the `pytest` implementation working without any issues.