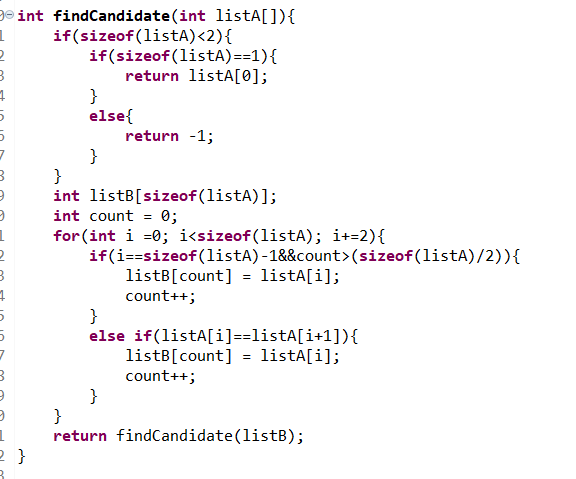
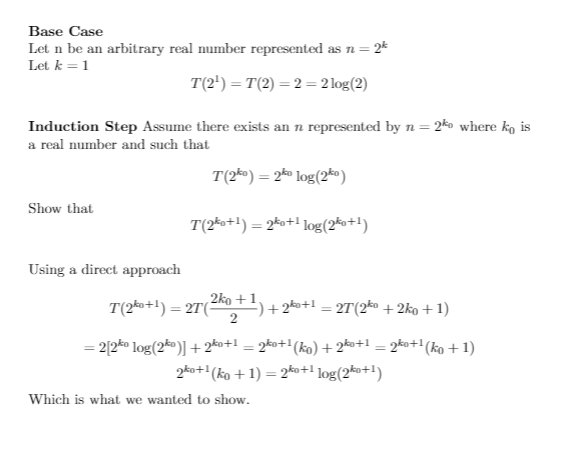
William Pramono

Assignment 2

COSC 160: Data Structures

1. 1. The recursion terminates when the array size is smaller than 2.  
   2. When N is odd, add the last element if no majority element has been found. If a majority element has been found already, ignore the last element.   
   3. O(n)  
   4. Override the second array instead of creating new arrays every single time it’s recursively called.  
   5. 

2) I typed it up in Latek because I wanted it to look really nice.

3)

|  |  |  |
| --- | --- | --- |
|  | Operation A | Operation B |
| Time | O(n) | O(2^n) |
| Space | Constant | 2^n |

The advantages of Operation A is that it grows at a slower rate than Operation B as well as uses less space. In a lot of cases, recursive problems are easier to write and understand than iterative solutions even if it takes up more time and space. Sometimes it can be really difficult to write Iterative functions but for this fibonacci it’s not that bad.

4) I would put the entire string into the stack so that the first character is in the bottom of the stack and the last character of the string is in the top of the stack. You would peep then pop the stack. If the top of the stack is the same character as the first character of the string, you would recursively call the function to get to the next letter in the string and the next character in the stack. If they’re not the same return false.

5) By using a single stack, it’s possible to determine whether the number of 0’s or 1’s in a binary string are equal or which number occurs more often. The goal is to keep the stack consisting of a single type of number. When the stack is empty, push whatever number is next in the binary string. If the stack has a series of numbers in it already, push the next number in the string if it’s the same as the numbers already in the stack or pop the stack if the numbers are different. If the binary string has an equal amount of 0’s and 1’s, then the stack should be empty at the end. If there are more 0s or 1s than the other, the size of the stack is the number of 0s or 1s that occur more frequently than the other.