Mohammed Touhid Chowdhury
Mtc405@nyu.edu
N14108583
Hw#5

1.

(a) 1. Create variable to count how many times the first functor comes true.

2. loop through the vector using the iterators in the parameter

3. if the first functor returns true, only then pass to second functor and increase the counter

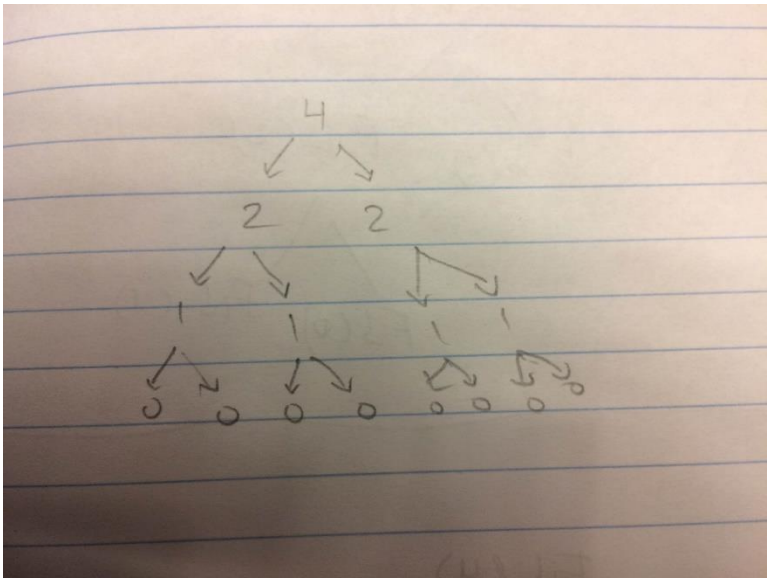
4. when loop ends return the counting var

(b) Precondition: start != end

Postcondition: counter <= size of vector

(c) O(n)

2.



(b)

4: 2: 1: 0: 0:

1: 0: 0:

*

2: 1: 0: 0:

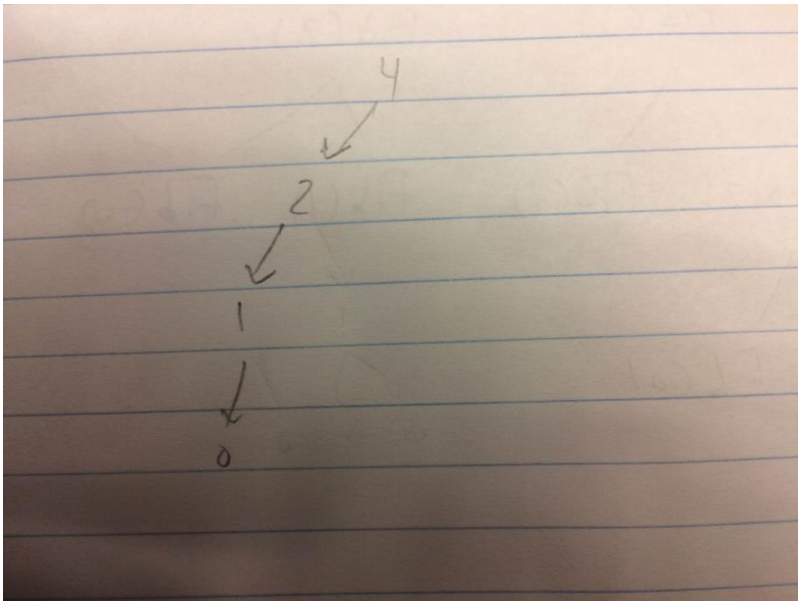
1: 0: 0:

*

***

(c) O(n*log(n)

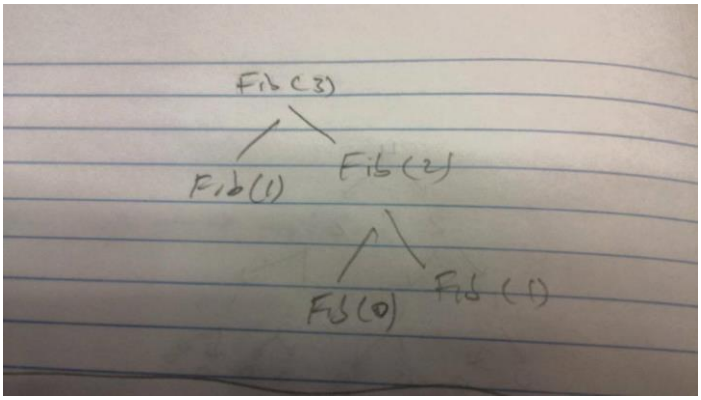
3.



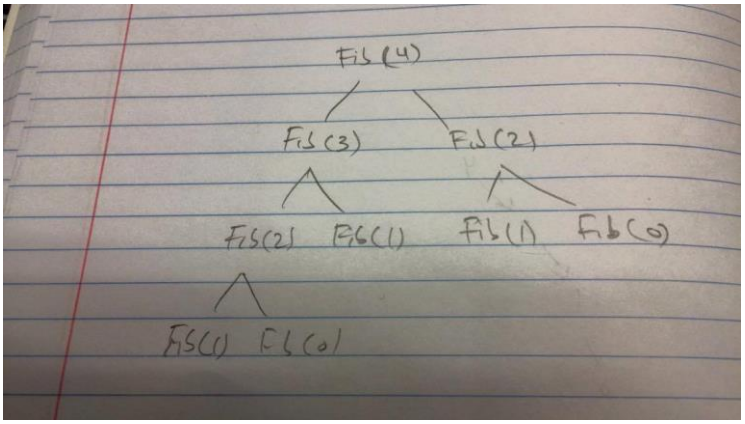
(b) 4: 2: 1: 0:

(c) O(n)

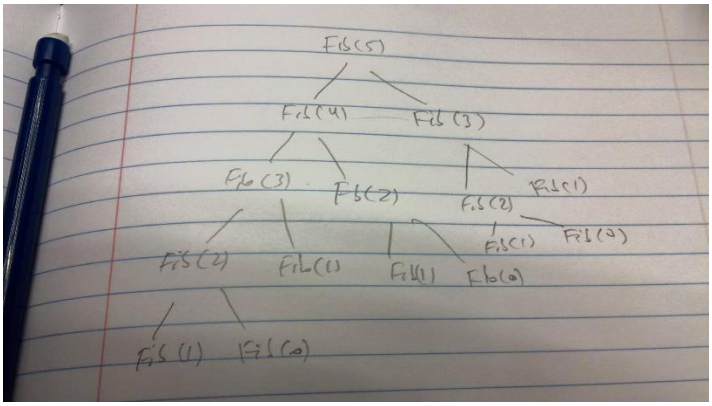
4.



So, Fib(3) calls 4 functions



So, Fib(4) calls 8 functions



So, Fib(5) calls 14 functions

5.

8, 9, -11 ,2 ,0 ,3

-11, 8, 9, 2, 0, 3

-11, 2, 8, 9, 0, 3

-11, 0, 2, 8, 9, 3

-11, 0, 2, 3, 8, 9

6.

7.

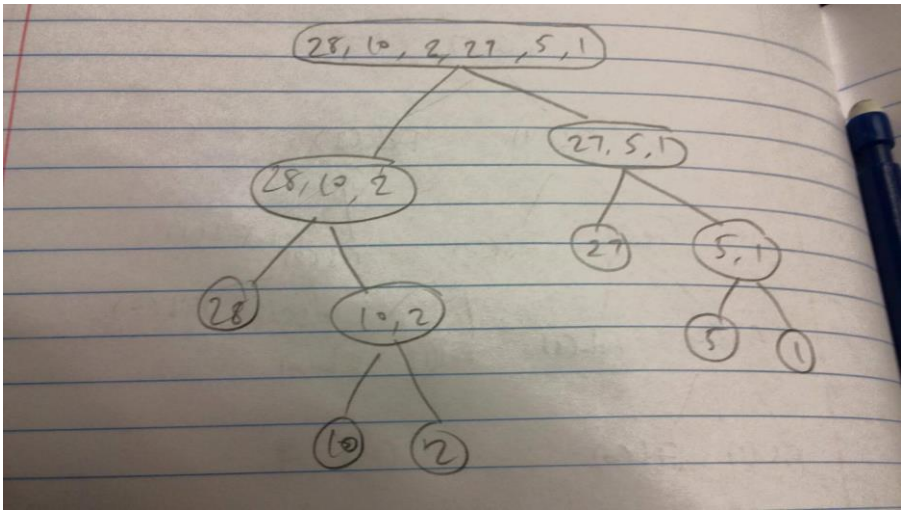
-11, 8,3,2,0,9

-11,0,2,9,8,3

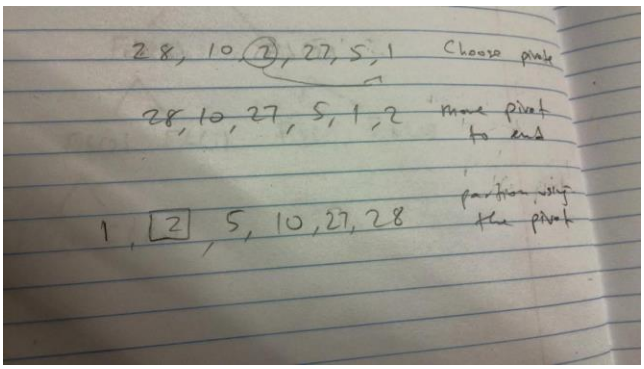
-11,0,2,3,8,9

8.

(a)



(b)



9.

insertion sort O(n^2)

merge sort O(n*log(n))

quick sort O(n*log(n))

10. O(n)

11. i + 1 cannot to be equal to k because the conditions are k<=i and k > i+1. If k is equal to i+1, then it would not call the recursive function.