HW2 - Question 4

a)
```
for (int i = n-1; i >= 0; i--){
    for( int k=0; k < i*n; k++) {
        // something                    O(1)
    }
}
```

$$T(n) = \sum_{i=0}^{n-1}\left[\sum_{k=0}^{i \cdot n} c\right] \quad i = n \therefore i \cdot n = n \cdot n = n^2$$

$$= \sum_{i=0}^{n-1} cn^2 = n(cn^2) = cn^3 \therefore \Theta(n^3)$$

b)
```
for(int i=0; i< n; i++){
    for ( int k= 1;  k <= n; k++){
        if( A[k] == i){
            for (int m = 1; m <= n; m = m+m){
                O(1)
            }
        }
    }
}
```

$$T(n) = \sum_{i=0}^{n-1}\left[\sum_{k=1}^{n} c + n \log n\right]$$

$$= \sum_{i=0}^{n-1} cn + n \log n$$

$$= (n-1)(cn) + n \log n$$

$$cn^2 - cn + n \log n$$

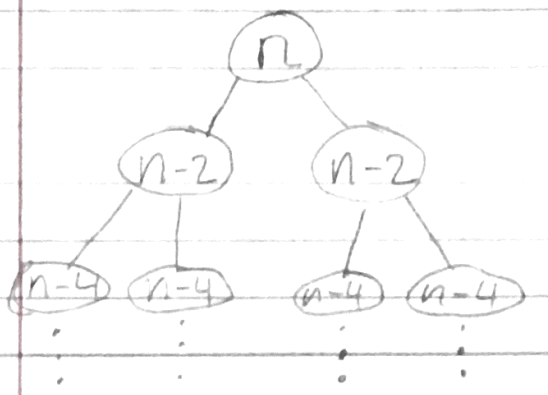
$$n^2 > n \log n$$

$\therefore \boxed{\Theta(n^2)}$

In this algorithm the two nested for loops have a complexity of n^2 while the if statement and its nested for loop have a combined time complexity of $n \log n$, which is intrinsically less than n^2 as input n expands. Therefore theta is n^2.

4c) $f(n) = f(n-2) + O(1) + f(n-2) =$

$= O(1) + 2f(n-2)$



Based off this tree and its rate of expansion, it appears we are dealing with a geometric series here

$2^0 + 2^1 + 2^2 + \ldots = 2^{n/2} \therefore \Theta(2^{n/2})$

HW2

Problem 4d)

When looking at this code we must determine how often the 'inner' if code evaluates. This first occurs at 10, then at 15 $(10 \cdot \frac{3}{2})$ and so on.

This relation can be illustrated as the following progression:

$$10 \cdot \left(\frac{3}{2}\right)^0 \quad 10\left(\frac{3}{2}\right)^1 \quad 10\left(\frac{3}{2}\right)^2 \cdots$$

This relation can be illustrated in the following manner:

$$10 \sum_{i=0}^{\log_{\frac{3}{2}} n} 10\left(\frac{3}{2}\right)^i$$ which has a $\boxed{\Theta(n)}$

Help from CP Xiaowen Zhu