HW #1 CSCI 104

Problem 1

K: iteration

a. Beginning : i=2

↳ K=1

- i: 2*2=4 => 2^{2^1}

↳ K=2

- i: 4*4 = 16 => 2^{2^2}

↳ K=3

- i: 16*16 = 256 => 2^{2^3}

↳ K=4

- i: 256*256 = 65,536 => 2^{2^4}

General case: $\underbrace{2^{2^K}}_{i} \geq n$

loop ends when condition is true

↳ $\log_2(2^{2^K}) \geq \log(n)$

↳ $\log_2(\log_2(2^{2^K})) \geq \log\log(n)$

↳ $K \geq \log\log(n)$

$\boxed{O(\log\log(n))}$

b.
```
for (i=1; i<=n; i++) {
    if (i % sqrt(n) == 0) {
        for (K=0; K<pow(i,3); K++) {
            //O(1) operation
        }
    }
}
```

when $i = m\sqrt{n}$, if statement is true ∴ $n = m\sqrt{n}$ represents how many times if statement is true

↳ $n/\sqrt{n} = m \Rightarrow m = \sqrt{n}$

↳ there are $\sqrt{n}$ multiple of $\sqrt{n}$ that satisfy if statement

$\sum_{i=1}^{n} \Theta(1)$ is outer for loop work

↳ $\Theta(n) + \ldots$ (rest)

how many times for loop is executed

$\sum_{m=1}^{\sqrt{n}} \sum_{K=0}^{i^3} \Theta(1)$

iterations for for loop

↳ when we get to inner for, i must be multiple of $\sqrt{n}$

↳ $\sum_{m=1}^{\sqrt{n}} \sum_{K=0}^{(m\sqrt{n})^3} \Theta(1) \Rightarrow \sum_{m=1}^{\sqrt{n}} \sum_{K=0}^{m^3 n^{3/2}} \Theta(1) \Rightarrow \sum_{m=1}^{\sqrt{n}} \Theta(m^3 n^{3/2}) \Rightarrow$

↳ the multiple we are currently on!

Problem 2b Cont...

$$\sum_{m=1}^{\sqrt{n}} \Theta(m^3 n^{3/2}) \Rightarrow \Theta(n^{3/2}) \sum_{m=1}^{\sqrt{n}} m^3$$

use formula : $\sum_{m=1}^{M} m^3 = \Theta(M^4)$

$\hookrightarrow \sum_{m=1}^{\sqrt{n}} m^3 = \Theta(\sqrt{n})^4 = \Theta(n^2)$

$\hookrightarrow \Theta(n^{3/2})\,\Theta(n^2) = \Theta(n^{7/2})$

Final expression : $\Theta(n) + \Theta(n^{7/2}) = \boxed{\Theta(n^{7/2})}$

1c.

```
for(i=1; i≤n; i++){
    for(k=1; k≤n; k++){
        if (A[k] == i){
            for(m=1; m≤n; m = m+m){
                // O(1) operation
            }
        }
    }
}
```

the if condition is an $\Theta(1)$ operation that is done n^2 times.

$\hookrightarrow \sum_{i=1}^{n} \sum_{k=1}^{n} \Theta(1) \Rightarrow \sum_{i=1}^{n} \Theta(n)$ ... $\Theta(n)$ operation done n times $= n^2$

$\Theta(n^2)$ + ... (rest)

$\hookrightarrow$ worst case : if statement = true always, so for each iteration of k loop we do $\log_2(n)$ operations, so must multiply

Answer : $\boxed{\Theta(n^2 \log_2(n))}$

#3

#2

HW #1 CSCI 104

Problem 1D

```
int f (int n)
{
    int* a = new int[10];      ] constant time
    int size = 10;
    for (i=0; i<n; i++) {      ] exectes "n" times
        if (i==size) {         ] constent time opeation
            int newsize = (3*Size)/2;
            int* b = new int[newsize];
            for (j=0; j<size; j++) {
                b[j] = a[j];
            }
            delete [] a;
            a=b;
            size = newSize;
        }
        a[i] = i*i;
    }
}
```

↳ outer for loop = $\sum_{i=0}^{n} \Theta(1) = \Theta(n)$

↳ use a geometric progression to model how many resizes there will be... we resize until the size of array is greater than n at smallest possible value

- geometric progression: $S_k = S_0 \times (r)^k$ ... $(S_k \geq n)$
  - Solve for k because thats the # of resizes (# of times i == size)
  - $S_0 \times (r)^k \geq n \Rightarrow 10 \cdot (3/2)^k \geq n \Rightarrow (3/2)^k \geq n/10$

$\Rightarrow \log([3/2]^k) \geq \log(n/10) \Rightarrow k = \frac{\log(n/10)}{\log(3/2)} \Rightarrow \log(n) - \cancel{\log(10)}$

$\Rightarrow \log(n)$ resizes... if statement is true $\log(n)$ times

Key difference

↳ ~~[illegible]~~ when work is multiplying @ each step; we are concerned w/ number of times an inner loop executes. Instead, here we are focused on the total # of elements copied during each resize.

use nested summations

↳ Since total # elements copied grows geometrically, we model w/ geometric progression, not nested summation

We want $S_n = (a_0)\left[\frac{r^{k+1}-1}{r-1}\right] = 10\left[\frac{(3/2)^{\log(n)+1}-1}{3/2-1}\right] =$

$$10 \times 2 \times \left(\left(\frac{3}{2}\right)^{O[\log(n)]}-1\right) = \left(\frac{3}{2}\right)^{O(\log(n))}$$

$$a^x = e^{x\log(a)} \quad \ldots \quad \therefore \left(3/2\right)^{O[\log(n)]} = e^{O(\log(n))\cancel{\cdot\log(3/2)}}$$

$$= e^{O(\log(n))}, \text{ e cancels out log}$$

$$= \boxed{O(n)}$$

#3

HW #1 CSCI 104

PROBLEM 2

1 -> 2 -> 3 -> 4 -> null   5 -> 6 -> null

a. in1 = 1, 2, 3, 4 & in2 = 5, 6   iteration : k

↳ llrec (in1 , in2) ... initial call (K=1)

- 1-2-3-4-null   5-6-null

- else : in1 -> next = llrec (in2 , in1 -> next)

5-6-null   2-3-4-null

1 - 5 - 2 - 6 - 3 - 4 - null !

↳ llrec (in1 , in2) , K = 2

5-6-null   2-3-4-null

- else : in1 -> next = llrec (in2 , in1 -> next)

2-3-4-null   6 -> null

5 - 2 - 6 - 3 - 4 - null

↳ llrec (in1 , in2) , k = 3

2-3-4-null   6 -> null

6 - 3 - 4 - null

- else : in1 -> next = llrec (~~in2~~ , in1 -> next)

2 - = 6-3-4-null ~~6 -> null~~   3-4-null

2 - 6 - 3 - 4 - null

↳ llrec (in1 , in2) , K = 4

6 -> null   3-4-null

3-4-null

- else : in1 -> next = ~~llrec (in2 , in1 -> next)~~

↳ 6 - 3 - 4   ~~3-4-null~~   null

6 - 3 - 4 - null

↳ llrec (in1 , in2)

3-4-null   null

3-4-null

- else if : return in1 , go up recursion stack

Answer : 1 -> 5 -> 2 -> 6 -> 3 -> 4 -> null

b in1 = null , in2 = 2

↳ k = 1

o return in2 ... function returns **2 -> null**