HW1 Q3

Saihaj Gulati

a)

```
void f1(int n)
{
    int i=2;
    while(i < n){
        /* do something that takes O(1) time */
        i = i*i;
    }
}
```

$\Theta(1)$ $\}\Theta(\log(\log n))$

$i = i^2$ starting at 2

2, 4, 16, 256

in 2s: $2, 2^2, (2^2)^2, ((2^2)^2)^2$

$2^1, 2^2, 2^4, 2^8$ → the exp is doubling → lg involved?

if $n = 2$, 2^1 runs 0 times
if $n = 4$, 2^2 runs 1 time
if $n = 16$, 2^4 runs 2 times
if $n = 256$ 2^8 runs 3 times

$\lg 1 = 0$
$\lg 2 = 1$
$\lg 4 = 2$
$\lg 8 = 3$

& in b/w its always round up

to get powers of 2 have to do lg anywaus

→ $\lg(256) = 8$ → $\lg(8) = 3$

$\lg(\lg n))$ → $\boxed{f(n) = \Theta(\log(\log n))}$

HW1 Q3

Saihaj Gulati

how many times hits
ex. $n=9 \Rightarrow$ 3 $\Rightarrow$ hit 3 times
ex. $n=16 \Rightarrow$ 4 $\Rightarrow$ hit 4 times

```
void f2(int n)
{
    for(int i=1; i <= n; i++){
        if( (i % (int)sqrt(n)) == 0){
            for(int k=0; k < pow(i,3); k++) {
              /* do something that takes O(1) time */
            }
        }
    }
}
```

$\Rightarrow \Theta(i^3)$

$$\sum_{k=0}^{i^3-1} \Theta(1) = i^3 - 1 = \Theta(i^3)$$

b)

innermost for loop: runs i^3 times $\Rightarrow \Theta(i^3)$

if statement hit $\sqrt{n}$ times

every time; if statement times

$$\sum_{i=1}^{n} \Theta(1) + \sqrt{n} \cdot \sum_{k=0}^{i^3} \Theta(1) = n + \sqrt{n} \cdot i^3$$

worst case i is n

$$n + \sqrt{n} \cdot n^3 \Rightarrow \boxed{\Theta(n^{3.5})}$$

HW1 Q3

c) innermost:
m: 1, 2, 4, 8, 16 if n was 16, 5 times → lg16+1
really m^2 are in n
$\Theta(\lg(n)+1 \cdot 1) = \Theta(\log n)$

```
for(int i=1; i <= n; i++){
  for(int k=1; k <= n; k++){
    if( A[k] == i){
      for(int m=1; m <= n; m=m+m){
        // do something that takes O(1) time
        // Assume the contents of the A[] array are not changed
      }
    }
  }
}
```

Σ worst case, every single one is equal
→ $\Theta(\log n)$

second for loop → $\sum_{k=1}^{n} \Theta(\log n)$ → $n \cdot \log n$ → $\Theta(n \log n)$ for second for loop

first for loop → $\sum_{i=1}^{n} \Theta(n \log n)$ → $n \cdot n \log n$ → $n^2 \log n$

Runtime: $\boxed{f(n) = \Theta(n^2 \log n)}$

d)

```
int f (int n)
{
  int *a = new int [10];
  int size = 10;
  for (int i = 0; i < n; i ++)
  {
    if (i == size)
    {
      int newsize = 3*size/2;
      int *b = new int [newsize];
      for (int j = 0; j < size; j ++) b[j] = a[j];
      delete [] a;
      a = b;
      size = newsize;
    }
    a[i] = i*i;
  }
}
```

$\Theta(1)$ $\Theta(1)$
→ 10, 15, 22.5, 33.75 3 times → end thing over is 4
$\frac{3 \cdot size}{2}$ $\Theta(1)$
$\Theta(1)$
} $\Theta(size)$
$\Theta(1)$ $\Theta(1)$ $\Theta(1)$

innermost for loop: $\sum_{j=0}^{size-1} (\Theta(1)) = size - 1 \to \Theta(size)$

$33.75/1.5 = 22.5/1.5 = 15/1.5 = 10$

$33.75 = 10 \cdot 1.5^3$ if n was this
# of times run

#

$n = 10 \cdot 1.5^{\#}$

$\frac{n}{10} = 1.5^{\#}$

$\log_{1.5}\left(\frac{n}{10}\right) = \#$ → if statement runs logn times, every time is $\Theta(n)$

$\log_{1.5} n \cancel{- \log_{1.5} 10}$

$\log n$

For loop outside while condition

logn times $\Theta(size)$ when

~~$(1 - \log n)(\Theta(1))$~~

$\log n(\Theta(size)) + \Theta(n)$

$\log n(\Theta(size)) + \Theta(n)$

if $n = 33.75$

size $= 10, 15, 22.5, 33.75 = n$

size is n or smaller

→ n is worst case for size

$10 \cdot 1.5^{\log_{1.5} n}$

$\log n(size) + n = \log n \cdot n \cancel{+ n}$

$\Theta(n \log n)$

$\boxed{f(n) = \Theta(n \log n)}$