EECS 233 SI Session One

Leader: Bertram Su

September 8, 2019

Objectives:

Upon completion of this SI session, participants will be able to:

1. Determine the big O expression for a method
2. Determine dominating factors in an equation

Exercises:

Write the simplest big O expression

1. $\cancel{30000^2} + nlog_2n + n$ $O(n \log_2 n)$

   $\cancel{n} \cdot \log_2 n \quad \cancel{n} \cdot 1$

2. $2n^{1/2} + \log_3 n$ $n^k > \log$ $n^{\frac{1}{2}}$

3. $2n^1 + n^{0.5} + 0.5n^{1.25}$ $n^{1.25}$

4. $2^n + \cancel{99999}n^{64}$ 2^n

5. $n*2^n + 4^n$ 4^n

   $n \cdot 2^n + 2^n \cdot 2^n$

6. $\cancel{64^{32}} + nlogn + 2^n + \cancel{n}$ 2^n

   $n > \log$

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

Determine the tightest big O equation from the code samples

```
7. void example(int n){
        for(int i = 0 ; i < n; i++) n
              System.out.println("data structures is fun");
        for(int i = 0 ; i < n; i++) n
              System.out.println("this is why we learn math");
   }
```

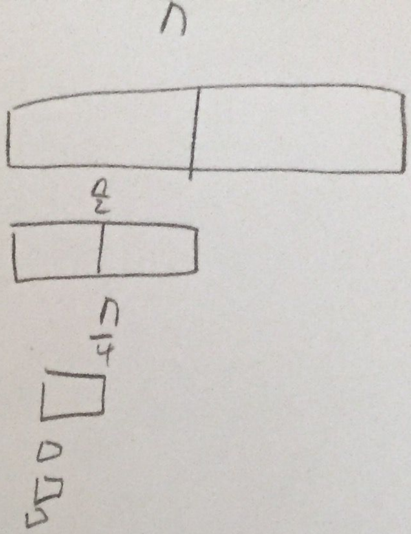
$\rangle\ n + n = 2n = n$

$O(n)$

```
8. void algebra(int n) {
        for (int i = 0; i < n; ++i) {n
              for (int j = 0; j < i; ++j) {n
                    System.out.println("fruit ");
              }
              for (int k = 0; k < n * 3; ++k) { 3n
                    System.out.println("loops");
              }
        }
   }
```

$\rangle (n + 3n)n = (4n)n = 4n^2 = n$

n^2



```
9. void dividing(int n) { //Assume n is positive
        if (n <= 0) return;
                System.out.println(”recursion is useful”);
        dividing(n/2);          n/3
}
```

$\log_2 n$

n/5

```
10! void tricky(int n){
        for(int i = 0 ; i < n; i++){ n
                for(int j =0 ; j<x ;){                    //Assume x has been initialized with a value
                        x--;
                }
        System.out.println(“This question was on a test”);
        }
}
```

n+x

*Would be
n + log x
if we included j++

Summary Checks:

11. Create a big O notation hierarchy that has ~~n~~, ~~k (constant)~~, n^2, log(~~n~~), 2^n, and ~~nlogn~~

K < log(n) < n < n log n < n^2 < 2^n

12. Give the simplest big O expression

```
public void summary(int n){
        for(int j = 0 ; j< n; j++){ n
                int r = y; //assume y has been initialized somewhere else
                for(int k = 0 ; r>0; k++){ log y
                        r = r/2;
                }
        }
}
```

→ n(log(y))

n log y

Upcoming Events and Suggestions for Further Study:

Events:

- Next SI session is Thursday at Sears 336 from 6:00 to 7:30

Further Study:

- “Big O Complexity: Why Base of Logarithm Don’t Matter”
  - great YouTube video that proves why bases don’t matter in Big O
- http://www.austincc.edu/akochis/COSC2415/bigo.htm
  - pros and cons of big O notation
- bigocheatsheet.com
  - A great graph that visualizes the big o complexity chart. It also has the big O time of data structures and algorithms that we will cover in the future.