CS062

DATA STRUCTURES AND ADVANCED PROGRAMMING

15: Midterm Review

Alexandra Papoutsaki
LECTURES

Mark Kampe
LABS

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time

Content adopted from Princeton COS226

A refresher

- $a^b = c \to b = \log_a c$
- $\log_a a = 1$
- $\log_a 1 = 0$
- $\log_a \frac{x}{y} = \log_a x - \log_a y$
- $\log_a (x \times y) = \log_a x + \log_a y$
- $\log_a x^y = y \times \log_a x$
- $\log_a x = \frac{\log_b x}{\log_b a}$
- $x^{\log_a y} = y^{\log_a x}$
- $a^{\log_a x} = x$
- $\log n! \sim n \log n$

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops

Summation basics

- $$\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n$$
- $$\sum_{i=1}^{n} c = c + c + \ldots + c = n \times c$$
- $$\sum_{i=1}^{n} (c \times f_i) = c \sum_{i=1}^{n} f_i$$
- $$\sum_{i=1}^{n} (f_i + g_i) = \sum_{i=1}^{n} f_i + \sum_{i=1}^{n} g_i$$

Useful summations to know

- $$\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}$$
- $$\sum_{i=1}^{n} i^2 = 1 + 4 + 9 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$$
- $$\sum_{i=0}^{n} r^i = \frac{r^{n+1} - 1}{r - 1}, r \neq 1$$
- $$\sum_{i=0}^{n} 2^i = 1 + 2 + 4 + \ldots + 2^n = 2^{n+1} - 1$$
- $$\sum_{i=0}^{n} (\frac{1}{2})^i = 1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^n} \sim 2$$
- $$\sum_{i=0}^{n} (\frac{1}{i}) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n} \sim \ln n$$

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time

Example 1

- ```
  for (int i = 5; i < n + 3; i += 2){
          doSomething();
  }
  ```
- Called $\frac{(n+3-5)}{2} = \frac{n-2}{2}$ times~$O(n)$
  ```

Example 2

- ```
  for (int i = 1; i < n; i*= 2){
          doSomething();
  }
  ```
- Called $\log n$ times~$O(\log n)$
  ```

Example 3

- ```
  for (int i = 5; i < n + 3; i *= 3){
          doSomething();
  }
  ```

- Will call it till $5 \times 3^i \geq n + 3$. Solving for i we get $i = log_3 \frac{n+3}{5} \sim O(\log n)$
  ```

Example 4

- ```
  for (int i = 1; i < n*n*n*n; i *= 3){
        doSomething();
  }
  ```
- Called $\log_3 n^4 = 4\log_3 n$ times~$O(\log n)$
  ```

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time

Example 1

- ```
  for (int i = 0; i < n; i++){
      for(int j=0; j<50; j++){
          doSomething();
      }
  }
  ```
- Called $50 \times n$ times~$O(n)$
  ```

Example 2

- ```
  for (int i = 5; i < n; i++){
      for(int j=0; j<n; j+=2){
          doSomething();
      }
  }
  ```

- Called $(n-5)\times\frac{n}{2}=\frac{1}{2}(n^2-5n)$ times~$O(n^2)$
  ```

Example 3

- ```
  for (int i = 0; i < n; i++){
      for(int j=1; j<n; j*=2){
          doSomething();
      }
      for(int j=0; j<n; j++){
          doSomething();
      }
  }
  ```
- Called $n \times (\log n + n) = n^2 + n \log n$ times~$O(n^2)$
  ```

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time

$$\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}$$

Example 1

-
```
for (int i = 1; i <= n; i++){
    for(int j=0; j<=i; j++){
        doSomething();
    }
}
```

- When i = 1, perform 1 inner loop iteration, when i = 2, perform 2 inner loop operations,…, when i = n, perform n inner loop iterations.

- `doSomething()` is called $1 + 2 + 3 + \ldots + n$ times, that is
$$\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2} \sim O(n^2)$$

Example 2

$$\sum_{i=0}^{n} 2^i = 1 + 2 + 4 + \ldots + 2^n = 2^{n+1} - 1$$

- ```
  for (int i = 1; i <= n; i*=2){
      for(int j=1; j<=i; j++){
          doSomething();
      }
  }
  ```
- When i = 1, perform 1 inner loop iteration, when i = 2, perform 2 inner loop operations, when i=4, perform 4 inner loop operations, ..., when i = n=2^x, perform n=2^x inner loop iterations. If we write n as a power of 2:
- `doSomething()` is called $1 + 2 + 4 + \ldots + n = 2^0 + 2^1 + \ldots + 2^x$ times, that is $\sum_{k=1}^{\log n} 2^k = 2^{\log n+1} - 1 = 2 \times 2^{\log n} - 1 = 2n - 1 \sim O(n)$
  ```

Example 3

$$\log_a(x \times y) = \log_a x + \log_a y$$

- ```
  for (int i = 1; i <= n; i++){
        for(int j=1; j<i; j*=2){
              doSomething();
        }
  }
  ```
- When i = 1, perform $\log 1$ inner loop iteration, when i = 2, perform $\log 2$ inner loop operations,…, when i=n perform $\log n$ inner loop operations.
- `doSomething()` is called
  $\log 1 + \log 2 + \ldots + \log n = \log(1 \times 2 \times \ldots \times n) = \log n!$ times
  $\sim O(n \log n)$
  ```

Example 4: The infamous 3-sum brute force algorithm

- ```
  for (int i = 0; i < n; i++){
      for(int j=i+1; j<n; j++){
          for(int k=j+1; k<n; k++){
              doSomething()
          }
      }
  }
  ```
- Called $\sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=j+1}^{n} 1 = \frac{1}{6} n(n^2 - 3n + 2)$ times ~$O(n^3)$.
- How? https://www.wolframalpha.com/input/?i=%28sum+%28sum+sum%281+k%3Dj%2B1+to+n%29+j%3Di%2B1+to+n%29++i%3D1+to+n%29 Not expected to figure out something that complicated by hand. You could approximate summations as integrals.
  ```

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time

Example 1

- ```
  for (int i = n; i > 0; i/= 2){
        doSomething();
  }
  ```
  ```

Example 1 - Answer

- ```
  for (int i = n; i > 0; i/= 2){
          doSomething();
  }
  ```
- Called $\log n$ times~$O(\log n)$
  ```

Example 2

- ```
  for (int i = 10; i < n+5; i*=3){
        doSomething();
  }
  ```
  ```

Example 2 - Answer

- ```
  for (int i = 10; i < n+5; i*=3){
          doSomething();
  }
  ```

- Will call it till $10 \times 3^i \geq n + 5$. Solving for i we get $i = log_3 \frac{n+5}{10} \sim O(\log n)$
  ```

Example 3

- ```
  for (int i = 10; i < n; i++){
      for(int j=0; j<n; j+=2){
          doSomething();
      }
  }
  ```
  ```

Example 3 - Answer

-
```
for (int i = 10; i < n; i++){
    for(int j=0; j<n; j+=2){
        doSomething();
    }
}
```

- Called $(n-10)\times\frac{n}{2}=\frac{1}{2}(n^2-10n)$ times~$O(n^2)$

Example 4

```
for (int i = 1; i <= n*n-10; i++){
    for(int j=1; j<=i; j++){
        doSomething();
    }
}
```

$$\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}$$

Example 4 - Answer

- ```
  for (int i = 1; i <= n*n-10; i++){
        for(int j=1; j<=i; j++){
              doSomething();
        }
  }
  ```
- When i = 1, perform 1 inner loop iteration, when i = 2, perform 2 inner loop operations, …, when i = n*n-10, perform n*n-10 inner loop iterations.
- `doSomething()` is called $1 + 2 + 3 + \ldots + (n^2 - 10)$ times, that is

$$\sum_{i=1}^{n^2-10} i = \frac{(n^2-10) \times ((n^2-10)+1)}{2} \sim O(n^4)$$
  ```

Example 5

```
for (int i = 1; i <= n; i++){
        for(int j=1; j<=i; j++){
                for(int k=1; k<=i; k++){
                        doSomething();
                }
        }
}
```

Example 5 - Answe

$$\sum_{i=1}^{n} i^2 = 1 + 4 + 9 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

```
for (int i = 1; i <= n; i++){
    for(int j=1; j<=i; j++){
        for(int k=1; k<=i; k++){
            doSomething();
        }
    }
}
```

- When i = 1, perform 1x1 inner loop iteration, when i = 2, perform 2x2 inner loop operations, ..., when i = n, perform nxn inner loop iterations.
- `doSomething()` is called $1 + 4 + 9 + \ldots + n \times n = 1^2 + 2^2 + 3^2 + \ldots + n^2$ times, that is $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \sim O(n^3)$

Lecture 15: Midterm Review

- Logarithms
- Summations
- Simple Loops
- Nested Independent Loops
- Nested Dependent Loops
- Practice Time