This is C++, but I bet you can still read it!

**Announcements**

- **Hangman** – due today

- **Apt Set 2** – due Sept 16
  - Make ONE APT project
  - Add new class for each APT
  - Submit to Apt2
  - ALL APTs (even those from APT set 1)
### Feedback
- Anonymous feedback
- UTA feedback
  - If someone is missing in the Link, let us know
  - If someone is GREAT, let us know

### From last time
- Array – ordered, indexed, fixed length
- List – ordered, indexed, adjustable length
- Set – unordered, adjustable length, no doubles
- Map – unordered, pairs (key, value)
Map

• Unordered collection of values mapped to keys
  • dictionary
    • key – word
    • value - definition

Map

• Map<Double, Integer> map =
  new HashMap<Double, Integer>();

  for(double d: map.keySet()){
    System.out.println(d + ": " + map.get(d));
  }

http://docs.oracle.com/javase/6/docs/api/java/util/HashMap.html
public class References {
    public static void randomFunction2() {
        String a = "Hello";
        String b = "Goodbye";
        a = b;
        a = a.concat(" CS201");
        System.out.println(a);
        System.out.println(b);
    }
    public static void main(String[] args){
        randomFunction2();
    }
}

References

• http://www.youtube.com/watch?v=vm5MNP7pn5
Big-Oh

• Big-Oh
  • Estimate time required for a program
  • No units of time!!!!!!!
  • Count operations

Assign costs to operations

• Declarations cost 0 units
  • double aDouble;

• Operations cost 1 unit
  • aDouble = 4.56 //assignment
  • aDouble * 5 //mathematical operation
  • return aDouble; //returns
```java
public double getArea(double r) {
    double pi;
    pi = 3.14;
    double area;
    area = pi * r * r;
    return area;
}
```

**Big-Oh**

- Assign costs to operations
  - Declarations cost 0 units
  - Operations cost 1 unit
- Write in Big-Oh notation
Big-Oh

1 public double getArea(double r) {
2    double pi; 0
3    pi = 3.14; 1
4    double area; 0
5    area = pi * r * r; 3
6    return area; 1
7 } Total: 5

O(5)

Big-Oh

- Assign costs to operations
  - Declarations cost 0 units
  - Operations cost 1 unit
- Write in Big-Oh notation
- Simplify
  - Remove constants
    - O(6) = O(1)
    - O(4N) = O(N)
    - O(3N^2 + 5) = O(N^2)
  - Remove lower order terms
    - O(N^2 + N) = O(N^2)
1 public double getArea(double r) {
2     double pi;
3     pi = 3.14;
4     double area;
5     area = pi * r * r;
6     return area;
7 }

Total: 5

$O(1)$  $O(5)$

Big-Oh

- Assign costs to operations
  - Declarations cost 0 units
  - Operations cost 1 unit
- Write in Big-Oh notation
- Simplify
  - Remove constants
    - $O(6) = O(1)$
    - $O(4N) = O(N)$
    - $O(3N^2 + 5) = O(N^2)$
  - Remove lower order terms
    - $O(N^2 + N) = O(N^2)$
Your turn

```java
public static int sum( int n)
{
    int partialSum;
    partialSum = 0;
    for(int i = 1; i <= n; i++)
        partialSum += i * i * i;
    return partialSum;
}
```

O(1+N*4 + 1) = O(4N+2)
• Rules
  • for-loops
    • (statements in for-loop) * iterations
  • Nested for-loops (inside-out)
    • (statements in innermost for-loop) * iterations * iterations
  • Consecutive statements
    • Add them
  • If/else
    • Test + max(if, else)

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c$</td>
<td>Constant</td>
</tr>
<tr>
<td>$\log N$</td>
<td>Logarithmic</td>
</tr>
<tr>
<td>$\log^2 N$</td>
<td>Log-squared</td>
</tr>
<tr>
<td>$N$</td>
<td>Linear</td>
</tr>
<tr>
<td>$N \log N$</td>
<td></td>
</tr>
<tr>
<td>$N^2$</td>
<td>Quadratic</td>
</tr>
<tr>
<td>$N^3$</td>
<td>Cubic</td>
</tr>
<tr>
<td>$2^N$</td>
<td>Exponential</td>
</tr>
</tbody>
</table>
The traveling salesperson

- **Brute-Force Solution:** $O(n!)$
- **Dynamic Programming Algorithms:** $O(n^2 2^n)$
- **Selling on eBay:** $O(1)$

Practice

go.gl/dGwqAL
numberOne

1 public int numberOne(int n) {
2    return n;
3 }

numberTwo

1 public int numberTwo(int n) {
2    int answer = 1;
3    for(int i = 0; i < n; i++)
4        answer *= n;
5    return answer;
6 }
public int numberThree(int n){
    int answer = 1;
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            answer *= n;
    return answer;
}

public int numberFour(int n){
    int answer = 1;
    for(int i = 0; i < n; i++)
        answer *= n;
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            answer *= n;
    return answer;
}
public int numberFive(int n) {
    int answer = 1;
    for (int i = 1; i <= n; i = i*2) {
        answer *= n;
    }
    return answer;
}

public int numberSix(int n) {
    int answer = 1;
    for (int i = 1; i <= n; i = i*2) {
        for (int j = 0; j < n; j++) {
            answer *= n;
        }
    }
    return answer;
}
```java
public int numberSeven(int n) {
    if (numberTwo(n) > 10000) {
        return n;
    } else {
        return numberFive(n);
    }
}
```

```java
public int numberTwo(int n) {
    int answer = 1;
    for (int i = 0; i < n; i++) {
        answer *= n;
    }
    return answer;
}
```

```java
public int numberFive(int n) {
    int answer = 1;
    for (int i = 1; i <= n; i = i * 2) {
        answer *= n;
    }
    return answer;
}
```

**Announcements**

- Hangman – due today
- Apt Set 2 – due Sept 16
  - Make ONE APT project
    - Add new class for each APT
  - Submit to Apt2
    - ALL APTs (even those from APT set 1)