CSE115 / CSE503
Introduction to Computer Science I

Dr. Carl Alphonce
343 Davis Hall
alphonce@buffalo.edu

Office hours:
Thursday 12:00 PM – 2:00 PM
Friday 8:30 AM – 10:30 AM
OR request appointment via e-mail
Turn off and put away electronics:

- cell phones
- pagers
- laptops
- tablets
- etc.
Today
binary search

Coming up
CEN demonstration project
review
BINARY SEARCH
(data must be sorted: ordered from smallest to largest)
Let’s find 53 using binary search.
Initial interval is [0, 8)
Compare the target to the value at the mid-point index of the interval. For interval [0,8), midpoint index is \((0+8)/2 = 4\).
Since $53 > 29$, focus attention on interval to the right of 29, $[5, 8)$. 

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>-18</td>
<td>-3</td>
<td>12</td>
<td>17</td>
<td>29</td>
<td>34</td>
<td>47</td>
<td>53</td>
</tr>
</tbody>
</table>
Compare the target to the value at the mid-point index of the interval.
For interval [5,8), midpoint index is (5+8)/2 = 6.
Since 53 > 47, focus attention on interval to the right of 47, [7,8).
Compare the target to the value at the mid-point index of the interval.  
For interval [7,8), midpoint index is \((7+8)/2 = 7\).

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>-18</td>
<td>-3</td>
<td>12</td>
<td>17</td>
<td>29</td>
<td>34</td>
<td>47</td>
<td>53</td>
</tr>
</tbody>
</table>
We found 53!

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>-18</td>
<td>-3</td>
<td>12</td>
<td>17</td>
<td>29</td>
<td>34</td>
<td>47</td>
<td>53</td>
</tr>
</tbody>
</table>
public boolean isMemberOf(int target, List<Integer> list) {
    int left = 0;
    int right = list.size();
    while (left != right) {
        int mid = (left + right) / 2;
        if (list.get(mid) == target) { return true; }
        if (target < list.get(mid)) { right = mid; }
        if (list.get(mid) < target) { left = mid + 1; }
    }
    return false;
}