A New Data Structure Arrives:

**ADT:**
- insert
- remove
- isEmpty

---

### Implementation of _______________

<table>
<thead>
<tr>
<th>insert</th>
<th>removeMin</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(n)</td>
<td>O(n)</td>
<td>Unsorted Array</td>
</tr>
<tr>
<td>O(1)</td>
<td>O(n)</td>
<td>Unsorted List</td>
</tr>
<tr>
<td>O(lg(n))</td>
<td>O(1)</td>
<td>Sorted Array</td>
</tr>
<tr>
<td>O(lg(n))</td>
<td>O(1)</td>
<td>Sorted List</td>
</tr>
</tbody>
</table>

**Q1:** What errors exist in this table? (Fix them!)

**Q2:** Which algorithm would we use?
template <class T>
void Heap<T>::_insert(const T & key) {
    // Check to ensure there’s space to insert an element
    // ...if not, grow the array
    if ( size_ == capacity_ ) { _growArray(); }
    // Insert the new element at the end of the array
    item_[++size] = key;
    // Restore the heap property
    _heapifyUp(size);
}

What’s wrong with this code?

Heap Operation: removeMin / heapifyDown:

Q: How do we construct a heap given data?

CS 225 – Things To Be Doing:
1. Theory Exam 3 starts next Thursday (April 4th)
2. MP5 +5 deadline is tonight; due next Monday (April 1st)
3. lab_hash is out today, due on Sunday (March 31st)
4. Daily POTDs are ongoing :)