A Heap Data Structure  
(specifically a minHeap in this example, as the minimum element is at the root)

Given an index $i$, its parent and children can be reached in $O(1)$ time:  
- $\text{leftChild} := 2i$  
- $\text{rightChild} := 2i + 1$  
- $\text{parent} := \text{floor}(i / 2)$

Formally, a complete binary tree $T$ is a minHeap if:
- 
- 

Inserting into a Heap

What's wrong with this code?

What's wrong with this code?

Running time of insert?

Heap Operation: removeMin / heapifyDown:
```cpp
// Heap.cpp (partial)

// Heap.cpp (partial)

template <class T>
void Heap<T>::_removeMin() {
    // Swap with the last value
    T minValue = item_[1];
    item_[1] = item_[size_];
    size--;
    // Restore the heap property
    heapifyDown();
    // Return the minimum value
    return minValue;
}
```

```cpp
// Heap.cpp (partial)

template <class T>
void Heap<T>::_heapifyDown(int index) {
    if ( !_isLeaf(index) ) {
        T minChildIndex = _minChild(index);
        if ( item_[index] ___ item_[minChildIndex] ) {
            std::swap( item_[index], item_[minChildIndex] );
            _heapifyDown( ________________ );
        }
    }
}
```

Q: How do we construct a heap given data?

Theorem: The running time of buildHeap on array of size n is: __________.

Strategy:

Define S(h):
Let S(h) denote the sum of the heights of all nodes in a complete tree of height h.

S(0) =
S(1) =
S(h) =

Proof of S(h) by Induction:

Finally, finding the running time:

```
// Heap.cpp (partial)

// Heap.cpp (partial)

template <class T>
void Heap<T>::buildHeap() {
    for (unsigned i = parent(size); i > 0; i--) {
        heapifyDown(i);
    }
}
```

Running Time?

**CS 225 – Things To Be Doing:**

1. Theory Exam 3 starts next week (Tuesday, April 3rd)
2. MP5 deadline is Monday, April 2nd
3. lab_hash is due Sunday, April 1st
4. Daily POTDs are ongoing!