Today's announcements:

MP6 available, due 4/12, 11:59p.

Exam 2: returned in section this week.

SECRET MYSTERY DATA STRUCTURE...

Secret mystery data structure

ADT - _____

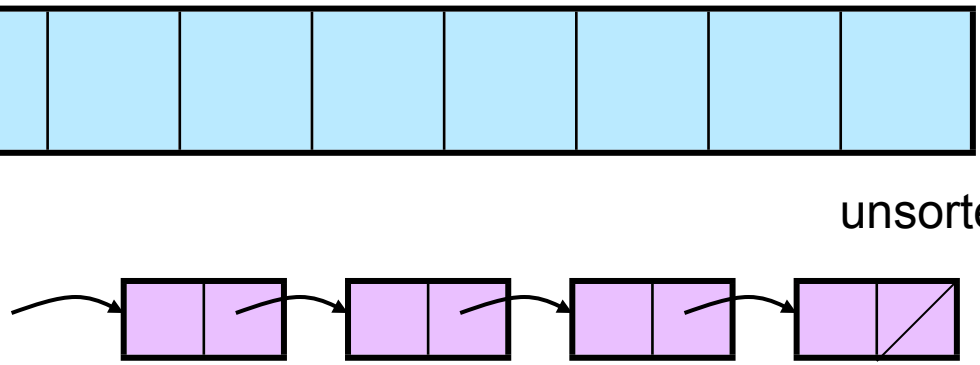
insert

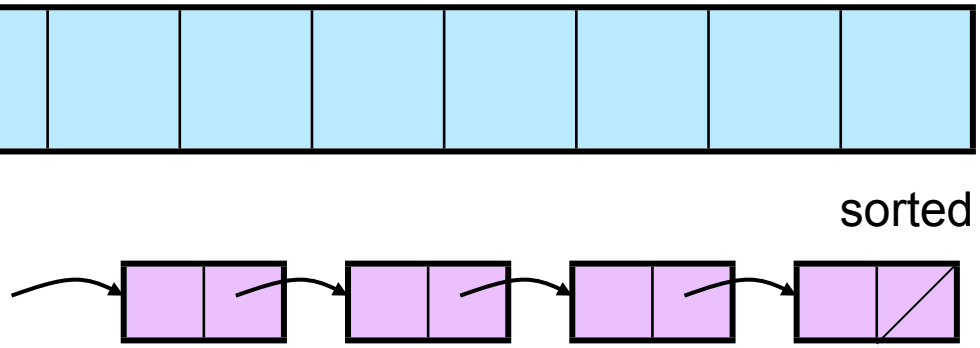
remove

getSize

Priority Queue ADT:

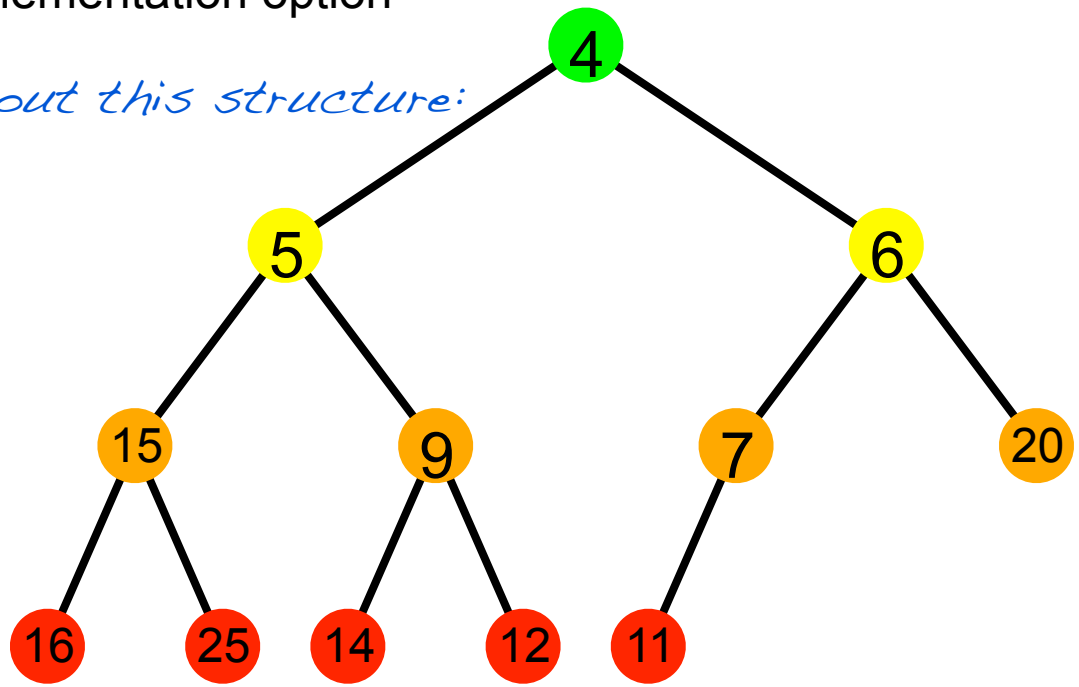
insert	removeMin	implementation
O(n)	O(n)	unsorted
O(1)	O(n)	
O(log n)	O(1)	sorted
O(log n)	O(1)	



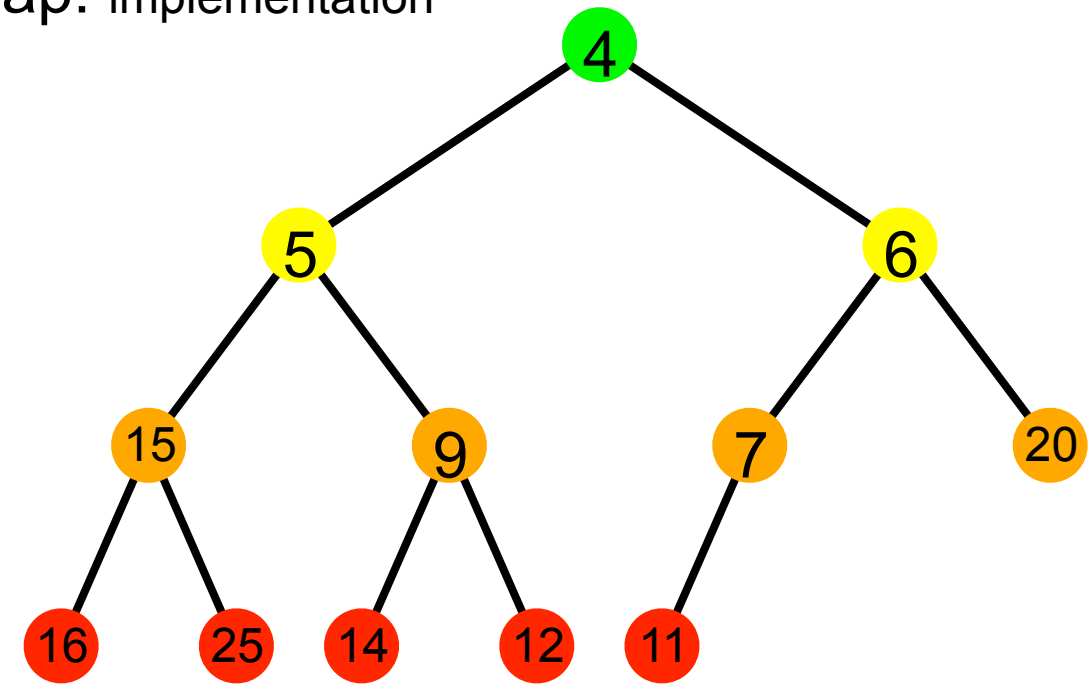


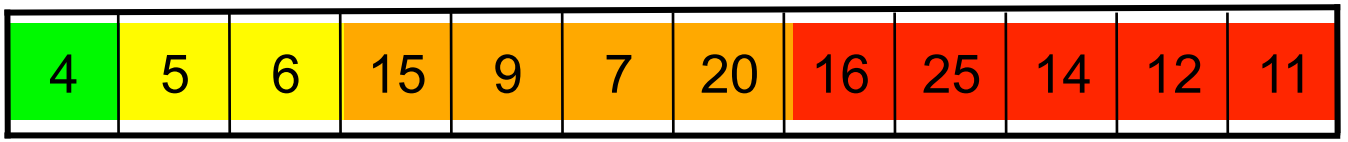
Priority Queue: another implementation option

Tell me everything you can about this structure:

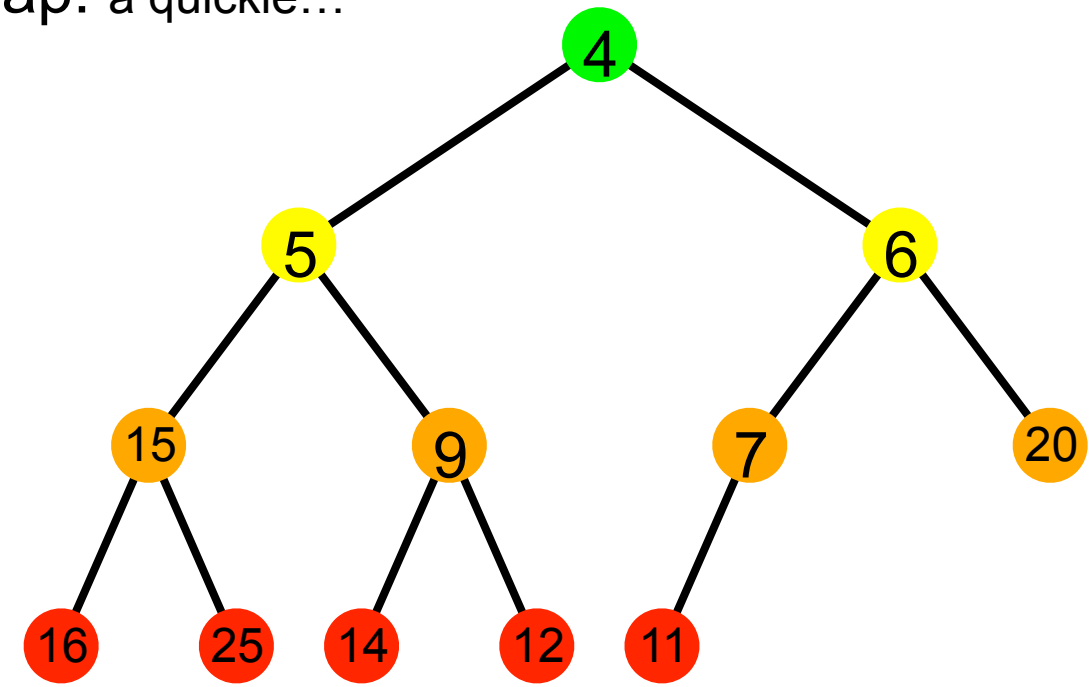


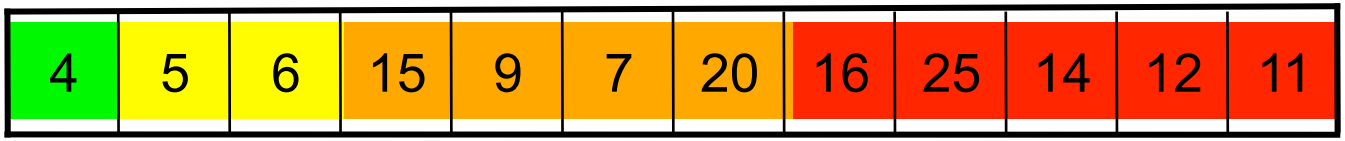
(min)Heap: implementation





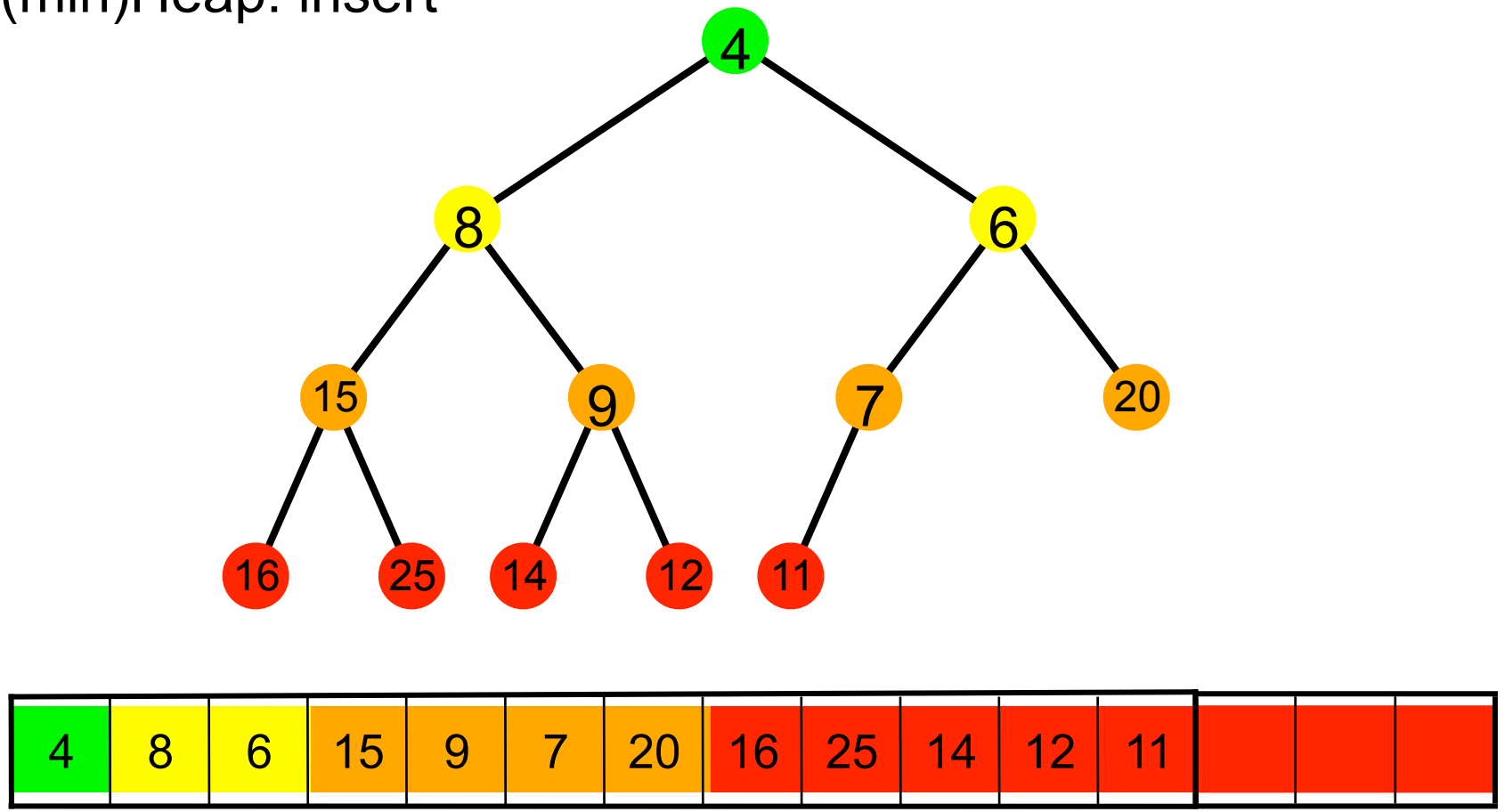
(min)Heap: a quickie…





What is the max height of a complete tree containing n nodes?

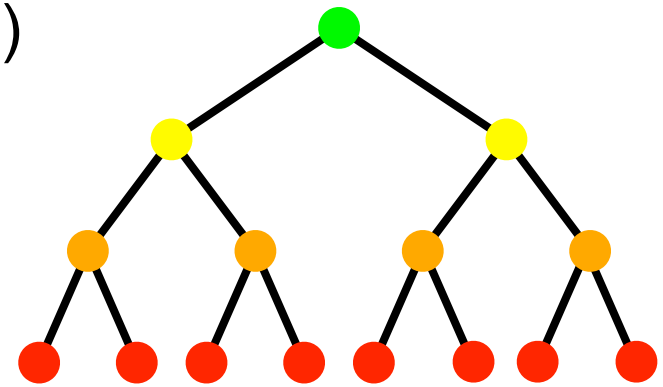
(min)Heap: insert



Code:

```
template <class T>
void Heap<T>::insert(const T & key){

   if (size==capacity) growArray();
   size++;
   items[size] = key;
   heapifyUp(size);

}
```

growArray()

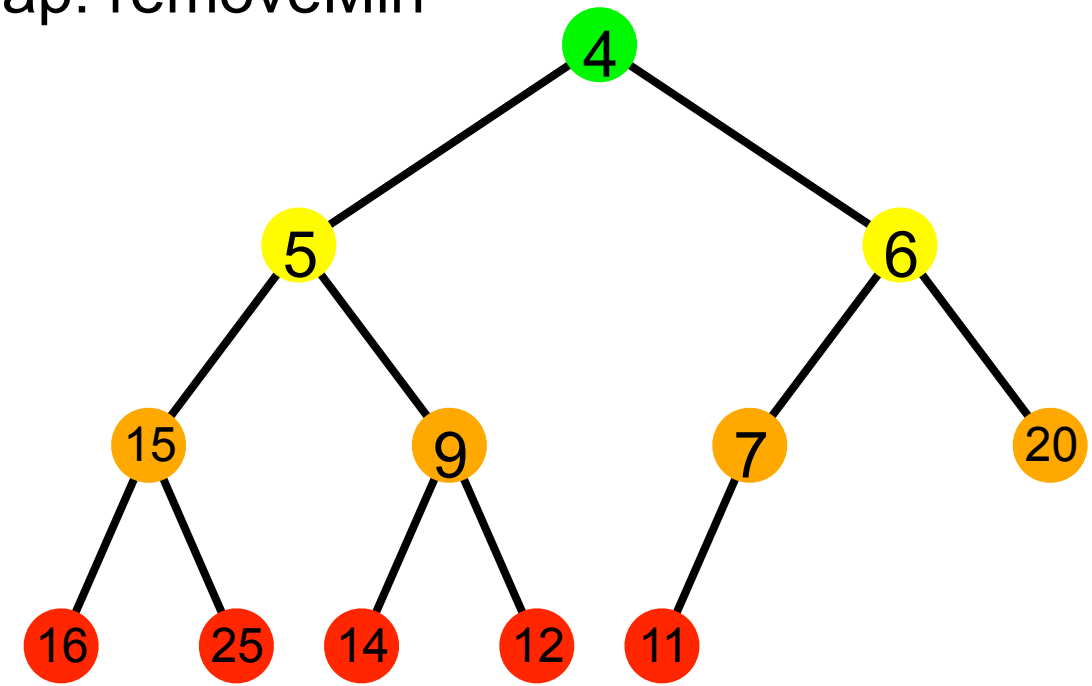


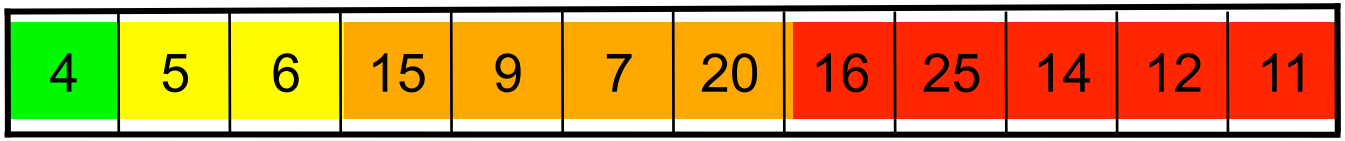
Code:

```
template <class T>
void Heap<T>::insert(const T & key){

   if (size==capacity) growArray();
   size++;
   items[size] = key;
   heapifyUp(size);

}
```

```
template <class T>
void Heap<T>::heapifyUp(int cIndex){
   if (cIndex > ___){
      if (items[cIndex] ___ items[parent(cIndex)]{
         swap(_________________,_________________);
         heapifyUp(_________________);
      }
   }
}
```

(min)Heap: removeMin





Code:

```
template <class T>
void Heap<T>::removeMin(){
   items[1] = items[size];
   size--;
   heapifyDown(1);
}
```

Code:

```
template <class T>
void Heap<T>::removeMin(){
   items[1] = items[size];
   size--;
   heapifyDown(1);
}
```

```
template <class T>
void Heap<T>::heapifyDown(int cIndex){
   if (hasAChild(cIndex)){
      minChildIndex = minChild(cIndex);
      if (items[cIndex] ____ items[minChildIndex]{
         swap(_____________,_____________);
         _________________________;
   }
}
```