```
My nested class:
  class node { // An inner class within heap
  public:
    std::string id; // The id of this node
    int key; // The key of this node
    void *pData; // A pointer to the actual data
  };
The declarations of my data vector and hash table pointer:
  std::vector<node> data; // The actual binary heap
  hashTable mapping; // maps ids to node pointers
Private member functions of my heap:
  void percolateUp(int posCur);
  void percolateDown(int posCur);
  int getPos(node *pn);
The start of my heap constructor:
  heap::heap(int capacity):mapping(capacity*2)
    // Allocate space for the nodes (0 slot is not used)
    data.resize(capacity+1);
    ... The heap constructor continues ...
A simple getPos implementation:
  int pos = pn - \&data[0];
  return pos;
An example of a call to the hash table's setPointer member function:
  mapping.setPointer(data[posCur].id, &data[posCur]);
An example of a call to the hash table's getPointer member function:
  node *pn = static_cast<node *> (mapping.getPointer(id, &b));
Filling in ppData in deleteMin:
  *(static_cast<void **> (ppData)) = data[1].pData;
```