Csci 335 Assignment 4

Due Wednesday, December 4th

Mutable Priority Queue

Modify the BinaryHeap implementation given by the book to support the `decreaseKey(key, delta)`, `increaseKey(key, delta)` and `remove(key)` operations. In addition to the binary heap you will need a hash table in order to make these operations logarithmic worst-case time. This additional hash table needs to be updated during every `insert` and `deleteMin` operation on the priority queue. (Note: The hash table will probably have to be updated many times during each operation.) The hash table should hash on the same key used for comparison in the queue and should store the key and the position in the binary heap array where the item is stored.

`decreaseKey` should lower the value of the key by a positive amount delta and restore the heap order property with a percolate up operation. `increaseKey` should raise the value of the key by delta and restore the heap order property with a percolate down operation. `remove` can be implemented by performing `decreaseKey(key, inf)` and then performing `deleteMin`. Instead of using an inf constant you can also peek at the current min in the heap and use that to make sure you remove the correct key.

Testing

Create a test routine that allows a user to input values into a heap until they enter "end". Then let the user remove values from the heap until they enter "end". Finally, perform deleteMin and print the min value until the queue is empty, printing the remaining values in sorted order.

For example:

```
./testHeap

Enter values for heap, enter "end" to stop:
```

- `15 7 32 6 5 20 48 70 end`

```
Enter values you would like to remove, enter "quit" to stop:
```

- `6 20 48 end`

```
Remaining values in sorted order:

5 7 15 32 70
```

This should also work if the user redirects input from a file, e.g. `./testHeap < input.txt`