Assignment A6

Due Date: December 7

Purpose
This is it! Finally!

The purpose of this last (and easier?) project is to implement a priority queue in the form of a heap. This will be done using a standard array.

Problem
For no reason whatsoever, Myquel Gousilini, a professor at a prestigious New England college, needs a priority queue of integers, where the smallest value has the highest priority. You, the owner of a software company called Algo-Rhythms, are called in to write this software. You decide to implement a heap, storing integers in a dynamic array.

Input
Your program should first prompt for two integers, \( n \) and \( m \) (see below). You may assume these will be input correctly. The program should then repeatedly show a menu of the following options:

- \( i \ x \) - insert \( x \) into its proper place in the heap, where \( x \) is an integer and \( x > 0 \).
- \( d \) - delete the highest priority item. A message should appear if the heap is empty.
- \( v \) - view the highest priority item. A message should appear if the heap is empty.
- \( s \) - display (show) the entire array. Values should be displayed in tree-level order (see Output, below).
- \( n \) - make a new array. This entails deleting the old array and creating a new one.
- \( q \) - quit.

Output
The program should display the items specified in the input. If the \( s \) (show tree) option is chosen, the array should be displayed in tree-level order, with two spaces between each integer:

```
1
2  3
4  5  6  7
8  9  10  11  12  13  14  15
```

etc.
Specifics

- The program should be written in C++ using the following class declaration:

  ```cpp
  class heap {
      int * theHeap;
      int n, m;
  public:
      heap ();
      void insertValue (int);
      void deleteValue ();
      int getValue ();
      void display ();
      void newHeap ();
  }
  ```

- You may add additional **private** data members or methods.

- The heap should be stored as an array. The array initially should be size \( n \). If the array is full and an integer is to be added, then the array should be increased by \( n \) each time such an addition is to take place. If the array has \( m \) locations empty, where \( m > n \), then the array size should be decreased by \( n \). This functionality will necessitate additional private methods in the class. **Whenever the array size is changed, display an appropriate message**, such as “Array increased to size 12.”

- Use good commenting techniques.

Notes

- Implementing this program should be much easier than previous projects.

- Email the program to me as usual. Remember to use your last name and A6 in your submission file, as in `gousieA6.cpp`. Hand in hard copy in the envelope on my door by 5:00 PM on Monday, December 10th.

- Write/print and sign the Wheaton Honor Code Pledge on what you turn in: “I have abided by the Wheaton College Honor Code in this work.”

---

*I got distracted by learning.*

– Amy Hopkinson ’09, in a Theory of Computation class.