CS 201 Data Structures Library Phase 3 Due 4/20

Phase 3 of the CS201 programming project, we will be built around heaps. In particular, you should implement both a standard binary heap and binomial heap. You will implement a class for each heap type.

You should create a class named Heap for the binary heap with public methods including the following (elmtype and valuetype are types from the template). You should use your dynamic array class for the heap storage.

Function	Description	Runtime
Heap();	Default Constructor. The Heap should be empty	O(1)
Heap(keytype k[], valuetype V[], int s);	For this constructor the heap should be built using the arrays K and V containing s items of keytype and valuetype. The heap should be constructed using the O(n) bottom up heap building method.	O(s)
~Heap();	Destructor for the class.	O(1)
keytype peekKey();	Returns the minimum key in the heap without modifiying the heap.	O(1)
valuetype peekValue();	Returns the value associated with the minimum key in the heap without modifiying the heap.	O(1)
keytype extractMin();	Removes the minimum key in the heap and returns the key.	O(lg n)
void insert(keytype k, valuetype v);	Inserts the key k and value v pair into the heap.	O(lg n)
void printKey()	Writes the keys stored in the array, starting at the root.	O(n)

Your class should include proper memory management, including a destructor, a copy constructor, and a copy assignment operator.

You should create a class named BHeap for the binomial heap with public methods including the following (elmtype and valuetype are types from the template). You should use dynamic allocation for the binomial trees.

Function	Description	Runtime
BHeap();	Default Constructor. The Heap should be empty	O(1)
BHeap(keytype k[], valuetype V[], int s);	For this constructor the heap should be built using the arrays K and V containing s items of keytype and valuetype. The heap should be constructed using repeated insertion.	O(s)
~BHeap();	Destructor for the class.	O(n)
keytype peekKey();	Returns the minimum key in the heap without modifiying the heap.	O(lg n)
valuetype peekValue();	Returns the value associated with the minimum key in the heap without modifiying the heap.	O(lg n)

keytype extractMin();	Removes the minimum key in the heap and returns the key.	O(lg n)
void insert(keytype k, valuetype v);	Inserts the key k and value v pair into the heap.	O(lg n)
void merge(BHeap<keytype, valuetype> &H2);	Merges the H2 into the current heap	O(lg n)
void printKey()	Writes the keys stored in the heap, printing the smallest binomial tree first. When printing a binomial tree, print the size of tree first and then use a modified preorder traversal of the tree. See the example below.	O(n)