05/24/17 01:58:52 /Users/hostname/Desktop/CSE 330/Lab7/heap_binary_tree.cpp

```cpp
// Adnar Lozano
// CSE 330 Data Structures
// Lab 7
// 5/20/17

#include <iostream>
#include <vector>
using namespace std;

class MinHeap {
private:
    vector<int> _vector;
    void BubbleDown(int index);
    void BubbleUp(int index);
    void Heapify();
public:
    MinHeap(int* array, int length);
    MinHeap(const vector<int>& vector);
    MinHeap();
    void Insert(int newValue);
    int GetMin();
    void DeleteMin();
};


MinHeap::MinHeap(int* array, int length) : _vector(length) {
    for(int i = 0; i < length; ++i)
        _vector[i] = array[i];
    Heapify();
}

MinHeap::MinHeap(const vector<int>& vector) : _vector(vector) {
    Heapify();
}

MinHeap::MinHeap() {}

void MinHeap::Heapify() {
    int length = _vector.size();
    for(int i=length-1; i>=0; --i)
        BubbleDown(i);
}

void MinHeap::BubbleDown(int index) {
    int length = _vector.size();
    int leftChildIndex = 2*index + 1;
    int rightChildIndex = 2*index + 2;
    if(leftChildIndex >= length)
        return;
    int minIndex = index;
    if(_vector[index] > _vector[leftChildIndex])
        minIndex = leftChildIndex;
    if((rightChildIndex < length) && (_vector[minIndex] > _vector[rightChildIndex]))
        minIndex = rightChildIndex;
    if(minIndex != index) {
        int temp = _vector[index];
        _vector[index] = _vector[minIndex];
        _vector[minIndex] = temp;
        BubbleDown(minIndex);
    }
}

void MinHeap::BubbleUp(int index) {
    if(index == 0)
        return;
    int parentIndex = (index-1)/2;
    if(_vector[parentIndex] > _vector[index]) {
        int temp = _vector[parentIndex];
```

```
69          _vector[parentIndex] = _vector[index];
70          _vector[index] = temp;
71          BubbleUp(parentIndex);
72      }
73  }
74
75  void MinHeap::Insert(int newValue) {
76      int length = _vector.size();
77      _vector[length] = newValue;
78      BubbleUp(length);
79  }
80
81  int MinHeap::GetMin() {
82      return _vector[0];
83  }
84
85  void MinHeap::DeleteMin() {
86      int length = _vector.size();
87      if(length == 0)
88          return;
89      _vector[0] = _vector[length-1];
90      _vector.pop_back();
91      BubbleDown(0);
92  }
93
94  int main() {
95      int array[] = {10, 4, 5, 30, 3, 300};
96      MinHeap minHeap(array, 6);
97      for(int i=0; i<3; ++i) {
98          cout << minHeap.GetMin() << "  ";
99          minHeap.DeleteMin();
100     }
101     return 0;
102 }
```