Printout for cs320-09-A08-Heapifier.hpp

```
// File: ASorter/Heapifier.hpp

#ifndef HEAPIFIER_HPP_
#define HEAPIFIER_HPP_

#include "ASeq.hpp"

// ========= siftUp =========
template<class T>
void siftUp(ASeq<T> &a, int lo, int i) {
    // Pre: maxHeap(a[lo..i - 1]).
    // Post: maxHeap(a[lo..i]).
    T temp = a[i];
    int parent = (i + lo - 1) / 2;
    while (lo < i && a[parent] < temp) {
        cerr << "siftUp: Exercise for the student." << endl;
        throw -1;
    }
    a[i] = temp;
}

// ========= siftDown =========
template<class T>
void siftDown(ASeq<T> &a, int lo, int i, int hi) {
    // Pre: maxHeap(a[i + 1..hi]).
    // Pre: lo <= i <= hi.
    // Post: maxHeap(a[i..hi]).
    T temp = a[i];
    int child = 2* i - lo + 1;
    bool done = hi < child;
    while (!done) {
        if (child < hi && a[child] < a[child + 1]){
            child++;
        } if (temp < a[child]) {
            a[i] = a[child];
            i = child;
            child = 2*i - lo+1;
            done = hi < child;
        } else {
            done = true;
        }
    }
    a[i]=temp;
}



#endif

// new page
```

Name:
Date:
Assignment:
-1

```
// File ASorter/InsertSorter.hpp
// Olivia Lara
// Sept. 26, 2017

#ifndef INSERTSORTER_HPP_
#define INSERTSORTER_HPP_

#include "ASorter.hpp"

template<class T>
class InsertSorter : public ASorter<T> {
public:

    ~InsertSorter() {
    }

protected:
    virtual void split(ASeq<T>&, int lo, int &mid, int hi) override;
    virtual void join(ASeq<T>&, int lo, int mid, int hi) override;
};

template<class T>
void InsertSorter<T>::split(ASeq<T> &, int, int &mid, int hi) {
    // Post: mid == hi.
    mid = hi;
}

template<class T>
void InsertSorter<T>::join(ASeq<T> &a, int lo, int mid, int hi) {
    // Pre: mid == hi && sorted(a[lo..hi - 1]).
    // Post: sorted(a[lo..hi]).
if ((mid != hi) || a[lo] > a[hi-1]){
        cerr << "Preconditions (mid == hi && sorted(a[lo..hi - 1])) not valid" << endl;
    }
    int j = mid;
    T key = a[mid];
    while (j > 0 && a[j-1] > key){
        a[j] = a[j-1];
j--; }
a[j] = key;
}

#endif

// new page
```

Indentation.
Use NetBeans to indent.

```
// File ASorter/MergeSorter.hpp
// Olivia Lara
// 09/19/2017

#ifndef MERGESORTER_HPP_
#define MERGESORTER_HPP_

#include "ASorter.hpp"
#include "ArrayT.hpp"
template<class T>
class MergeSorter :public ASorter<T> {

private:
    ArrayT<T> _tempA;

public:
    MergeSorter(int cap);
    ~MergeSorter() {
    }

protected:
    virtual void split(ASeq<T> &a, int lo, int &mid, int hi) override;
    virtual void join(ASeq<T> &a, int lo, int mid, int hi) override;
};
template<class T>
MergeSorter<T>::MergeSorter(int cap) :
    _tempA(cap) {
}
template<class T>
void MergeSorter<T>::split(ASeq<T> &, int lo, int &mid, int hi) {
    // Post: mid ==(lo + hi + 1) / 2
    mid = (lo + hi + 1) / 2;
}
template<class T>
void MergeSorter<T>::join(ASeq<T> &a, int lo, int mid, int hi) {
    int i = lo;
    int j = mid;
    for(int k = lo; k < (hi + 1); k++) {
        if (i == mid) {
            _tempA[k] = a[j];
            j++;
        } else if (j == (hi+1)) {  //*
            _tempA[k] = a[i];
            i++;
        } else if (a[i] < a[j]) {
            _tempA[k] = a[i];
            i++;
        } else {
            _tempA[k] = a[j];
            j++;
        }
    }
    for (int g = lo; g < hi+1; g++) {
        a[g] = _tempA[g];
    }
}

#endif

// new page
```