ENSF 594 – Principles of Software Development II
Summer 2022

Project: module 3 handout

The objectives of this module:

Create a library for vector-based heap

a. Min heap
b. Max heap

This library implements two heap classes, namely:

1. Min Heap (MinH)
2. Max Heap (MaxH)

Implementation

Both MinH and MaxH have the same structure, the difference is in some of the functionality to ensure heap rules for either one of them

Design NOTES: Both classes are very similar with identical implementation for many of their methods, hence a good design choice is to create a heap class that contains all the common functionalities for the other two classes to inherit. Not that this heap class should not have full implementation of the methods that are not identical for both children and you should not be able to instantiate any objects of this parent class

- Member variables:

The only member variable of these class is an object of the class vector to hold the content of the heap. the vector object is of type integer and is named ***elements***

```
private Vector<Integer> elements;
```

- Member functions:
  - Constructors:
    - Default constructor initializes vector to no size
    - Overload constructor initializes vector to a size passed as argument
    - Overload constructor creates a heap from an array and stores it in elements
  - getSize():
    - returns the size of the vector containing elements
  - isEmpty():
    - returns a Boolean True if the vector is empty, False otherwise
  - clear():
    - clears the elements of the vector
  - contains(Integer i)
    - searches for the value " i " in the heap
  - insert(Integer key)
    - inserts the value key to the vector and maintains heap properties
  - delete(Integer key)
    - removes the value key from the vector and maintains heap properties
  - sort()
    - applies heapsort to the vector content

- print()
  - displays the content of the heap vector over 2 lines. First line is the index of the parent of each element. Second line are the elements themselves
  - for example:

    -1 0 0 1 1 2 2

    15 10 14 7 5 12 1

- Helper functions:

These functions will be defined as private or protected (***depends on your design choices***) and will be accessible only within the class

- int parent(int i)
  - return parent of elements[i]
- int left(int i)
  - returns left child of elements[i]
- int right(int i)
  - returns right child of elements[i]
- swap(int x, int y)
  - swaps contents of indeces x and y
- heapifyDown(int i)
  - heapification process after deletion
- heapifyUp(int i)
  - heapification process after insertion
- Vector<Integer> heapify(int[] array)
  - takes in an array of values and returns a valid heap
  - this can be used by sort() and the third overload constructor

Library content after completion of this module:

- myLib
  - datastructures
    - nodes
      - SNode.java (optional)
      - DNode.java
      - TNode.java
    - Linear
      - SLL.java
      - DLL.java
      - CSLL.java → extends SLL
      - CDLL.java → extends DLL
      - StackLL.java → extends SLL
      - QueueLL.java → extends SLL
    - Trees
      - BST.java
      - AVL.java → extends BST
    - Heap
      - Heap.java → optional, to be extended by other two classes
      - MinH
      - MaxH
  - graphalgo