CMSC 123: Data Structures

1st Semester AY 2019-2020

Prepared by: CCS Templado & KBP Pelaez

[In Lab] Exercise 06: Heap ADT

Heap ADT

Even though the heap is another binary tree data structure, for this exercise, you must implement heaps using arrays.

To create a `HEAP ADT` that will hold integers, we define the following structure (defined in `heap.h`):

```c
typedef struct heap_tag{
    int *heap;    //pointer to an array of integers
    int size;     //size of the heap
    int maxSize;  //max allowed size of heap
    htype type;   //type of the heap (i.e., min or max)
} HEAP;
```

The type of a created `HEAP ADT` depends on the value of the `type` member of the structure. The possible values for `type` is specified by the `enum` declaration below (defined in `heap.h`):

```c
typedef enum HeapType {MINHEAP, MAXHEAP} htype;
```

`enum` declarations allow us to give names to values for a defined type. In this case, variables of type `htype` can be given the `MINHEAP` and `MAXHEAP` values (internally represented as 0 and 1, respectively). This allows the program to be more readable and easily maintainable.

Tasks

Implement and test the following functions (also listed in `heap.h`):

1. `HEAP* createHeap(int maxSize, htype type);` - a function returns a pointer to a heap whose maximum size is `maxSize` and whose type is defined by `type`.
2. `int isFull(HEAP *H);` - a function that returns 1 if the heap is full, otherwise, 0.
3. `int isEmpty(HEAP *H);` - a function that returns 1 if the heap is empty, otherwise, 0.
4. `void insert(HEAP *H, int key);` - a function that properly inserts `key` to the heap.
5. `int delete(HEAP *H);` - a function that deletes the root node in the heap and returns the deleted value. If the heap is a min heap, the deleted value must be the smallest, else if it's a max heap, return the largest value.
6. `void clear(HEAP *H);` - a function that deletes all the contents of he heap.
7. `int* heapSort(HEAP *H);` - a function that returns the sorted values of the heap. If the heap is a min heap, this must return an array in descending order, else if it is a max heap, this must return an array in ascending order.

The implementation for `printHeap` is already in `heap.c`.

Make sure to test your program using a shell file. Format is as follows:

1. Line 1 should contain either 0 or 1. 0 for MINHEAP and 1 for MAXHEAP.
2. Succeeding lines must contain one of the following commands:
    - `+ i` - inserts `i` to the heap
    - `-` - deletes the root node
    - `~` - prints the sorted version of the heap

   - `p` - prints the heap
   - `E` - checks if the heap is empty
   - `F` - checks if the heap is full
   - `C` - clears the contents of the current heap
3. The last line in the file must contain the `Q` command for the program to terminate.

Submission

Submit your `heap.c` to Google Classroom.

Questions?

If you have any questions, approach your lab instructor.