CSE 208 Offline 2

Offline on Advanced Heaps (Binomial Heap)

Deadline: 11:55 PM, 24 June 2025

Functionality:

In this assignment, you need to implement the Binomial (Min) Heap data structure. The required functionalities are specified below. The single letters in parentheses will be used to denote an operation during I/O. The amortized runtimes are specified at the end of each line. Some examples are also provided.

Your implementation must support these operations and maintain the amortized runtimes specified.

1. **Find Min (F):** Returns the node with the minimum key [O(log N)]
2. **Extract Min (E):** Returns the node with the minimum key and deletes it from the heap [O(log N)]
3. **Insert (I):** Inserts a new node in the heap [O(log N)]
    - E.g., **I 20** -> inserts a node with key 20
4. **Update Key (D):** Decreases the key of a particular node in the heap [O(log N)]
    - E.g., **D 22 14** -> Decrease the key from 22 to 14. The new key will always be smaller than the old key.
5. **Remove Key (R):** Delete the node with the specified key [O(log N)]
    - E.g., **R 14** -> Remove the node with key 14
6. **Print (P):** Prints the level order traversal with the level no. of each of the binomial trees in the heap. You must follow the format specified in the sample I/O.

Apart from these, you must properly implement all supporting functions like Make Heap, Union, etc.

Input/Output:

You will take input from a text file, where each line will specify one of the aforementioned operations. The operations are denoted by their corresponding letters, i.e., **I** for **Insert**, **R** for **Remove Key**, etc. Then the operands will follow where necessary. You can assume that all the operands will be integers.

You have to print the output to a text file. However, keep the provision of printing output to the console as well (but printing both to file and to console simultaneously will not be necessary).

Check out the Sample I/O for further clarification.

Special Instructions:

Use **OOP** principles. Write **readable, re-usable, well-structured, quality** code. This includes, but is not limited to, writing appropriate functions for implementation of the required algorithms, meaningful naming of the variables, suitable comments where required, proper indentation, etc.

Please DO NOT COPY solutions from anywhere (your friends, seniors, the internet, etc.). Any form of plagiarism (irrespective of source or destination) will result in -100% marks in the offline. Also, be informed that for repeated offences of plagiarism, the departmental policies suggest stricter measures.

You must understand all concepts clearly to be able to perform well in the viva. Failing in the viva will impact the offline marks too.

Submission Guidelines:

1. Create a directory with your 7-digit student id as its name
2. Put the source files only into the directory created in step 1
3. Zip the directory (compress in .zip format; .rar, .7z, or any other format is not acceptable)
4. Upload the .zip file on Moodle within the specified deadline

For example, if your student id is 2205xxx, create a directory named 2205xxx. Put only your source files(.c, .cpp, .java, .h, etc.) into 2205xxx. Compress 2205xxx into 2205xxx.zip and upload the 2205xxx.zip to Moodle.

Failure to follow the above-mentioned submission guidelines may result in up to 10% penalty.

Sample I/O:

Input	Output
I 5 I 2 P I 10 P F E P I 3 I 20 P D 10 1 P I 16	Printing Binomial Heap… Binomial Tree, B1 Level 0: 2 Level 1: 5 Printing Binomial Heap… Binomial Tree, B0 Level 0: 10 Binomial Tree, B1 Level 0: 2 Level 1: 5 Find Min returned: 2 Extract Min returned: 2 Printing Binomial Heap… Binomial Tree, B1

P R 20 P	Level 0: 5 Level 1: 10 Printing Binomial Heap… Binomial Tree, B2 Level 0: 3 Level 1: 5 20 Level 2: 10 Printing Binomial Heap… Binomial Tree, B2 Level 0: 1 Level 1: 3 20 Level 2: 5 Printing Binomial Heap… Binomial Tree, B0 Level 0: 16 Binomial Tree, B2 Level 0: 1 Level 1: 3 20 Level 2: 5 Printing Binomial Heap… Binomial Tree, B2 Level 0: 1 Level 1: 3 16 Level 2: 5