CSE3080-3: Data Structures (Spring 2020)
Midterm Project 2: Heap Sort

Handed out: May 20, Due: May 31, 11:59PM (KST)

1. Problem Description

In this project, we will implement a sorting program using a **heap**. Specifically, we will write a program that reads a list of commands from an input file and execute them, while maintaining a min heap and a max heap. In contrast to mp1, duplicate data can be inserted into the heap.

There are three commands that need to be supported by the program:

(1) insert

- This command inserts an integer into both the min heap and the max heap. If the line reads "INSERT 3", the program should insert a new element with data = 3 to the min heap and the max heap.

(2) print data in the ascending order

- The command "ASCEND" prints the elements in the heap in the ascending order. At this command, the program should write a line which contains the sorted elements in the output file named "mp2_result.txt". (Try running the given binary file.)

- You will need to use the min heap to execute this command. After printing data in the ascending order, you should still have the min heap with all the elements in it. Since you will need to delete elements from the min heap in order to accomplish this task, you should make a copy of the min heap before you start deleting the elements.

(3) print data in the descending order

- The command "DESCEND" prints the elements in the heap in the descending order. At this command, the program should write a line which contains the sorted elements in the output file named "mp2_result.txt". (Try running the given binary file.)

- You will need to use the max heap to execute this command. After printing data in the descending order, you should still have the max heap with all the elements in it. Since you will need to delete elements from the max heap in order to accomplish this task, you should make a copy of the max heap before you start deleting the elements.

※ In contrast to mp1, you should write your own code for printing data in the ascending and the descending order.

2. Problem Description (Read Carefully!)

(1) You should write a single C program, named **mp2**.

(2) The program takes one command-line argument, which is the input file name. For example, the user will run the program like this:

```
./mp2 input.txt
```

The first argument is the name of the input file.

(3) If the number of command-line arguments is not 1 (zero or more than 1), then you should print a usage string on the display and exit the program. The usage string looks like this:

```
usage: ./mp2 input_filename
```

(4) If the input file does not exist, your program should display an error message on the screen and terminate. The error message should be

```
The input file does not exist.
```

(5) The input file format is like the following. Each line contains one command.

```
INSERT 5
INSERT 8
INSERT 3
ASCEND
INSERT 10
DESCEND
```

Example input files will be given for your reference. At the evaluation, the TA will use a different input file. You may assume that the format of the input file is always correct; You do not need error handling for the contents of the input file.

(6) At the end of the program, you should print the following lines on the screen.

```
output written to mp2_result.txt.
running time: 0.052411 seconds
```

The actual running time printed will be different on each run. In order to print out the running time, use the function `clock()`.

(7) Once you implement the program, compare the execution time of mp1 with mp2 using the given input file "input_long.txt". Do they produce the same result? Which program runs faster? Can you reason why?

(8) Similar to other homework, you should write a Makefile. The TA will build your code by running "make". It should create a binary file, "mp2".

※ Test-run the given binary code and write your program so that it produces the same result as the given binary code.

※ Your implementation of min heap and max heap should support up to 1,000,000 elements.

3. Submission

You should submit your source codes and the Makefile. (You do not need to submit compiled binary files.) Combine your files into a zip file named cse3080_mp2_**20190001**.zip. The red numbers should be changed to **your student ID**. Submit your files on the cyber campus.

4. Evaluation Criteria

(1) Your program should compile well to produce the binary file.

(2) If the user does not give correct number of arguments, your program should print the usage message on the display and terminate.

(3) If the input file does not exist, your program should correctly print an error message on the display and terminate.

(4) When the user correctly executes the program with a command-line argument, your program should produce the result file "mp2_result.txt".

(5) The contents of your output file should be the same as the output file created from running the given binary file.

(6) The running time on the "input_long.txt" input file should be similar (on the cspro machine) to the given binary file "mp2_example". Points will be deducted if the running time of your program is close to the binary file "mp1_example".

5. Notes

- You must write your own code. You may discuss ideas with other students but must not copy their work. Similarly, you can find ideas on the Internet, but must not copy the code from the sources obtained from the Internet.

- Our department uses a software that detects duplicate codes. Since the software uses an assembly code level detection, just changing the variable names will not bypass the software. Make sure you write your own code from the scratch. Then, you will have no problem.

- Explained earlier, but your program should compile and run on the **cspro** machine. Make sure you check before you submit your work. The TA will download your code, run “make” and execute your binary files to check if they produce correct outputs.

6. Late Policy

10% of the score is deducted for each day, up to **three days**. Submissions are accepted up to three days after the deadline.