Homework #5

RELEASE DATE: 05/06/2014

DUE DATE: 05/20/2014, **17:30** (after class) in CSIE R217

As directed below, you need to submit your code to the designated place on the course website.

Any form of cheating, lying or plagiarism will not be tolerated. Students can get zero scores and/or get negative scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

Both English and Traditional Chinese are allowed for writing any part of your homework (if the compiler recognizes Traditional Chinese, of course). We do not accept any other languages. As for coding, either C or C++ or a mixture of them is allowed.

This homework set comes with 200 points and 20 bonus points. In general, every homework set of ours would come with a full credit of 200 points.

5.1 Heap and Hash

(1) (20%) Complete Exercise C-8.19 of the textbook. We need to see your **proof** for the bottom-up construction algorithm.

(2) (20%) Complete Exercise C-8.20 of the textbook.

(3) (20%) We talked about the max-min heap very briefly in class, which is an extension of max heap and takes alternating levels of max and min roots. In the paper that proposed the max-min heap,

Min-max heaps and generalized priority queues, Atkinson et al., 1986.

an extension called min-max-median heap, which allows to extract min, max, and median of a priority queue efficiently, was also proposed. Download and read the paper and illustrate to the TAs what the min-max-median heap looks like in your own words. (There is no need to write down the algorithmic details. We just need the key ideas behind the data structure.)

(4) (20%) Hash function is everywhere. Use any search engine to study the term "MD5." Explain to the TAs what it is and why it is now somewhat deprecated in your own words. Also, cite the website that you learn the term from.

(5) (20%) Suppose there are two strings that differ by only one character. Given a magic function `prefixHash(str, k)`, which returns the hash value on the first k characters (i.e. k-prefix) of the string `str` in $O(1)$ time complexity. Describe an algorithm to find out the position that the two strings differ efficiently. Briefly discuss and justify the time complexity of your algorithm. You can assume that collision does not happen. That is, two different strings will map to two different hash codes.

(6) (Bonus 20%) Construct a perfect hash function that is efficiently computable for the following 32 last names of professors in NTU CSIE. You need to explain why the hash function is perfect and why it is efficiently computable to get the full bonus.

`Chuang`, `Chou`, `Chang`, `Chao`, `Chen`, `Cheng`, `Chu`, `Fu`, `Fuh`, `Hsiang`, `Hsu`, `Hsueh`, `Hung`, `Jang`, `Kao`, `Kuo`, `Lai`, `Lee`, `Liao`, `Lin`, `Liou`, `Liu`, `Lu`, `Lyuu`, `Ouhyoung`, `Oyang`, `Shih`, `Tsai`, `Tseng`, `Wang`, `Wu`, `Yang`

5.2 Printing Queue

In this homework, we ask you to design a system to emulate the job queues of several printers. There are N printers (with $printer_{id}$ between 0 and $N-1$) available, and there are M operations to be processed. The operations we need are as follows:

(1) `add` job_{id} *priority* $printer_{id}$

- action: insert a job with job_{id} to the job queue of printer $printer_{id}$, with priority *priority*
- output: a line of
  num_of_jobs `jobs waiting on printer` $printer_{id}$

(2) `print` $printer_{id}$

- action: let the printer $printer_{id}$ print out the document with the **highest** priority on the job queue of this printer, and remove the job from the queue
- output: if there is a job to be printed, a line of
  job_{id} `printed`
  otherwise, a line of
  `no documents in queue`

(3) `move` $printer_{id1}$ $printer_{id2}$

- action: move all jobs from $printer_{id1}$ to $printer_{id2}$ while keeping their priorities, where the two printers will have different ids
- output: a line of
  num_of_jobs `jobs waiting on printer` $printer_{id2}$ `after moving`

It is guaranteed that all of the priority values and job ids will be distinct 32-bit signed integers, and all the operations will be legal (e.g. all ids will be within range). Please read the operations from the standard input, which would start with two integers N, M, each of which will be positive and will not be larger than 10^6. Then, the following M lines will contain the operations described above.

(1) (10%) Write down (on the written copy) the data structure you choose for this task, and describe how you test whether the data structure is correct.

There are three subtasks that the TAs will test for this problem.

(2) (20%) If your program can perform the `add` and `print` operations (but not `move` operations) correctly within the time limit, you get the score of this subtask.

(3) (20%) If your program can perform all the operations correctly when $M \leq 5000$ within the time limit, you get the score of this subtask.

(4) (40%) If your program can perform all the operations correctly within the time limit, you get the score of this subtask.

As hinted in the class, the usual heap may only work for (1) or maybe (2), but the binomial or leftist heap may work for all subtasks. So part of the homework is to read and decide whether you want to implement the binomial or leftist heap to get all the score.

Submission File (Program) and Written Copy

Please push your program to your forked repository `<user_name>/dsa14hw5` (on GitHub) before the deadline at 5:30pm on Tuesday (05/20/2014). We will use the latest time that you pushed to the repository as your submission time. Please **DO NOT PUT BINARY FILES** in your repository. As usual, the TAs will use CSIE R217 machines to test your code. **10** of the total points will depend on your Makefile (see below). Your repository should contain the following items:

- all the source code and your own `Makefile` such that `make printers` would generate a program named `printers`, which reads from the standard input and outputs as requested
- an optional `README`, anything you want the TAs to read before grading your code

For all the problems that require illustrations, please submit a written (or printed) copy in class or to CSIE R217 before the deadline.

MEDAL USAGE: If you want to use the gold medals for this homework, please visit `http://main.learner.csie.ntu.edu.tw/php/dsa14spring/login_medal.php` and submit the request **before the deadline + 4 days** (5/24).