Homework #5
RELEASE DATE: 05/12/2020
DUE DATE: 06/09/2020, **noon** on Gradescope/CEIBA

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 R-8.21 of the textbook.

(2) (20%) Complete Exercise C-8.4 of the textbook.

(3) (20%) Complete Exercise C-8.20 of the textbook. We don't need your proof of efficiency/correctness, but please describe your algorithm clearly for the TAs to grade.

(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 useful. 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 `postfixHash(str, k)`, which returns the hash value on the last k characters (i.e. k-postfix) 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 minimal perfect hash function that is efficiently computable for the following 32 standard keywords in C++. You need to explain why the hash function is perfect and why it is efficiently computable to get the full bonus.

```
auto, bool, case, char, const, continue, default, do, double, else, enum,
extern, float, for, goto, if, int, long, register, return, short, signed,
sizeof, static, struct, switch, template, union, unsigned, void, volatile,
while
```

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, each of which of quota M initially, and there are M operations to be processed. The operations we need are as follows:

(1) `add` job_id priority printer_id

- action:
  - If the current number of jobs equals `quota` of printer printer_id, execute `drop` on printer printer_id once.
  - Then, insert a job with job_id to the job queue of printer printer_id with priority priority
- output: after the line outputted from the `drop` operation (if any), a line of

  job_id added to printer printer_id with priority priority

(2) `quota` q printer_id

- action: execute `drop` printer_id multiple times until printer printer_id has no more than q jobs, and set the quota of printer_id to q
- output: after any lines outputted from the `drop` operations (if any), a line of

  quota q set for printer_id

(3) `drop` printer_id

- action: let printer printer_id drop the **lowest** priority job
- output: if there is a job to be dropped, a line of

  job_id dropped on printer printer_id

  otherwise, a line of

  no jobs in printer_id

(4) `print` printer_id

- action: let the printer printer_id print out the document with the **highest** priority on the job queue of printer printer_id, and remove the job from the queue
- output: if there is a job to be printed, a line of

  job_id printed on printer printer_id

  otherwise, a line of

  no jobs in printer_id

It is guaranteed that all of the priority values and job ids will be distinct 32-bit signed integers, all quotas will be positive 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.

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

(2) (30%) If your program can pass the cases with only `add` and `print` operations correctly within the time limit, you get the score of this subtask.

(3) (30%) If your program can perform all the operations without any memory constraint correctly within the time limit, you get the score of this subtask.

(4) (40%) If your program can perform all the operations, with a memory constraint that is slightly bigger than the maximum total number of waiting jobs, and complete correctly within the time limit, you get the score of this subtask.

The usual heap (you can directly call STL `priority_queue` if you want) should work for (1); using two usual heaps should work for (2). If you want to solve (3), one possibility is to use a min-max heap,

`https://en.wikipedia.org/wiki/Min-max_heap`

which is similar to the usual heap that we have taught but behaves differently in even and odd layers. Part of the homework is to learn the min-max heap by yourself to get all points for the problem.

Submission File

Please submit your written part of the homework to Gradescope before the deadline. Also, you need to upload your coding part as a single ZIP compressed file to CEIBA before the deadline. The zip file should contain no directories and only the following items:

- all the source code and your own `Makefile` such that `make printer` would generate a program named `printer`, which that reads the input from `stdin` and outputs to `stdout` for Problem 5.2.
- an optional `README`, anything you want the TAs to read before grading your code

Please make sure that your code can be compiled and run with the Makefile on CSIE R217 linux machines. Otherwise your program "fails" its most basic test and can result in ZERO!