EE 360C — Algorithms — Summer 2013
Programming Assignment #1

Due: July 3, 2013 11:59pm (via Blackboard)

Programming assignments are to be done individually. You may discuss the problem and general concepts with other students, but there should be no sharing of code. You may not submit code other than that which you write yourself or is provided with the assignment. This restriction specifically prohibits downloading code from the Internet.

Problem Description

Mr. Edison is very enthusiastic to open a great day-care center in a specific East Austin area and he has successfully hired a number of experienced nannies and his marketing campaign also attracted parents to enroll their kids into the day-care center. Because nannies and kids are all from the same area, they know each other. For each kid, every single nanny has her own preference list and for each nanny, every single kid has his/her own preference list as well. Also each nanny has her own capacity limitation as they can only handle a specific number of kids. Each nanny might have a different capacity in this regard, Ms. Grace can take care of five kids while Ms. Lucy can only take care of two kids without over-stressing herself.

Mr. Edison found out that there are more kids enrolled than can be handled by nannies hired. He regretted not hiring more nannies initially, which is a separate issue. He wants to find a way of assigning each kid to at most one nanny, in such a way that maximum capacity of all nannies can be filled. Since there are a surplus of kids, there would be some kids who do not get assigned to any nanny so Mr. Edison will have to refund these kids.

We say that an assignment of kids to nannies is *stable* if neither of the following situations arise:

- First type of instability: There are kids k and k', and a nanny na, such that
  - k is assigned to na, and
  - k' is assigned to no nanny, and
  - na prefers k' to k
- Second type of instability: There are kids k and k', and nannies na and na', so that
  - k is assigned to na, and
  - k' is assigned to na', and
  - na prefers k' to k, and
  - na' prefers k to k'.

So we basically have the Stable Matching Problem as presented in class, except that (i) nannies generally can accept more than one kid (unless the nanny hired is super-stressed), and (ii) there is a surplus of kids (sorry for their parents and Mr. Edison for lost income and customers).

Your job is to help Mr. Edison by devising and implementing an algorithm for nannies and kids allocation, similar to the Gale-Shapley algorithm. The algorithm must run in $O(n^2)$ time.

Thoroughly read this document, *especially* regarding input and output format before beginning to avoid getting a zero for non-comformance.

Input and output specification

- Kids are identified by an integer in the range 0 to $n-1$ (inclusive). Nannies are identified by an integer in the range 0 to $m-1$ (inclusive). Mr. Edison knows which nanny and kid each number refers to.
- The first line of input has two integers separated by a space: m (number of nannies) and n (number of kids). The second line will be a list of m integers separated by spaces, where the i^{th} integer specifies the number of kids the i^{th} nanny can handle. The next m lines will represent nannies' preference lists, each consisting of n space separated integers which identify kids. The i^{th} line in this set of lines corresponds to the preference list of the i^{th} nanny. The last n lines will represent kids' preference lists, each consisting of m integers which identify nannies. The i^{th} line in this set corresponds to the preference list of the i^{th} kid. The preference lists are given in order of most preferred to least preferred.
- Your program should read from `stdin` and output to `stdout`. For example, to test your program on `3-3-3.in`, we should be able to execute `$ program < 3-3-3.in` and it should output `-1 1 -1 -1 -1 -1 -1 2 0 -1`. The output consists of m space separated integers, where the i^{th} integer specifies the number of the nanny assigned to the i^{th} kid or -1 if no nanny has been assigned to him/her.
- We have provided six input files for you to verify your implementation. We will use different and more comprehensive test cases for grading.

Submission Instructions

- Make sure your program compiles on LRC machines before you submit it. It should compile using the standard commands `javac *.java`, `gcc *.c`, or `g++ *.cc` without any extra switches or any additional libraries.
- You should submit a single zip file titled `eid_lastname_firstname.zip` that contains all your program files and optionally a readme file. Do not put these files in a folder before you zip them (i.e. the files should be in the root of the ZIP archive) and do not include binaries.
- Your solution must be submitted via Blackboard *before* the deadline. No late submissions will be accepted.