CS 3050: Group Project

CS 3050

March 31, 2015

DUE: 05/04/2015

Topic: Allocating applicants to departments

Description

This is the final project of the course. The project can be done alone or in a group of 2 or 3 people. You should send an email to Avimanyou and myself by April 14 with the subject "CS3050 project" indicating whether you are choosing to do the project alone or in a group. If you choose to do the project in a group, please include the name of your teammates in the project (one email per team is fine).

The purpose of this project is to utilize and build upon your knowledge in algorithm design and implementation. You are going to design an algorithm that will allocate a group of job applicants to job vacancies in various departments of a company. You can assume that the total number of job applicants are less than the total number of job vacancies. If it is not the case then you should throw an error.

Your input file will have the following information:

- A list of departments along with the number of job vacancies in that department.
- A list of job applicants.
- For each department, a ranking of job applicants in decreasing order of preference.
- For each applicant, a ranking of departments in decreasing order of preference.

You have to find a way of allocating each job applicant to at most one department in such a way that all the vacancies in all the departments shall be filled. Note that since there are more job applicants than the number of vacancies, there would some job applicants who do not get a job. Furthermore, your allocation should be *well-formed*. An allocation of job applicants to departments is well-formed if none of the following two situation arises:

1. There are two job applicants A_1 and A_2 and a department D such that
   - A_1 is allocated to D, and
   - A_2 is not allocated to any department, and
   - D prefers A_2 to A_1.
2. There are two job applicants A_1 and A_2 and two departments D_1 and D_2 such that
   - A_1 is allocated to D_1, and
   - A_2 is allocated to D_2, and
   - D_1 prefers A_2 to A_1 and
   - A_2 prefers D_1 to D_2.

Input

The input is specified as follows.

- Each Input file starts with the keyword "Vacancies and Departments" in the first line. The next line is blank.

  The next few lines will contain the list of departments and the vacancies, which are specified as follows. Each line contains exactly information of only one department. The line will first contain the number of vacancies in a department and then the name of the department. Note that a number is any sequence of digits. The department name can consist of letters from the English Alphabet, underscores and spaces.

  The listing of departments and vacancies ends with a blank line.

- After the listing of the departments and vacancies, the listing of job applicants begin with the keyword "Job Applicants." The next line is blank.

  The next few lines will contain the list of job applicants. Each line will list exactly one job applicant. The name of a job applicant can consist of letters from the English Alphabet and spaces. The listing of job applicants ends with a blank line.

- After the listing of job applicants, we will start listing the preferences of each department. Preferences of department named D starts with the line "Preferences D." The line after this will be blank.

  The next few lines will list the job applicants in decreasing order of preference. Note that each line contains the name of one job applicant only. The listing of preferences ends with a blank line.

  Furthermore, if department D_1 is listed before department D_2 in "Vacancies and Departments" then preferences of D_1 must be listed before preferences of D_2.

- After the listing of preferences of each department, we will list the preferences of each applicant. Preferences of applicant named A starts with the line "Preferences A." The line after this will be blank.

  The next few lines will list the departments in decreasing order of preference. Note that each line contains the name of one department only. The listing of preferences of "A" ends with a blank line.

  Furthermore, if applicant A_1 is listed before applicant A_2 in "Job Applicants" then preferences of A_1 must be listed before preferences of A_2.

- The input ends with a line with the keyword "END INPUT."

A sample input file is included in Blackboard and also included in this document on Page 4.

Constraints

The constraints of the program include:

1. The program can be written in C or C++. If you want to use a different programming language or an IDE such as NetBeans, please check with me first.
2. Use a Makefile to compile and run the program.
3. Include a README file to tell the TAs how to run your program.
4. You should also include a project report in which you detail your efforts, the algorithms used (along with their complexity) and the contribution of your team members. We shall use this report primarily to give you partial credit if your program does not work and to check if all the team members contributed to the project.
5. Your program must work on Babbage.
6. At the end of the program, output on stdout, each department name and the job applicant allocated to it.
7. If you want to use libraries other than the standard input/output libraries, please check with us.
8. A moderate amount of error checking and resource management is required. Your application should check that each line from the input file is properly formatted, that the file is successfully opened, the file is successfully closed upon reading of the file and that all allocated space is de-allocated at the exit. If the input is not properly formatted, you should report the error on stdout and exit the program.

9. A moderate amount of formatting and documentation is required. Comments should be descriptive and used to illustrate the purpose and inner workings of an algorithm or function; they should not be used to annotate each line or self-evident logic.

Due Dates

- April 14 is the date that you must inform us about the composition of your team.
- During the week of April 20-24, please try to meet one of the TAs or myself during office hours and show the progress of the project. If the times do not work, please send us an email and we will try to find additional time to check the progress.
- The final submission is due May 4^{th} at 11:59:59 pm.

Submission

You should place all your submission material into a folder named as follows: *pawprint1_pawprint2_pawprint3_project*. Where the *pawprint1_pawprint2_pawprint3* is the list of you and your group member pawprints. To prepare your submission, you must remove any data files, object files and executables from the folder.

You will submit one file, a tar-ball of the directory containing all the source code and appropriate Makefile(s). The naming convention should be: *pawprint1_pawprint2_pawprint3_project.tgz*
Use

```
tar czvf pawprint1_pawprint2_pawprint3_project.tgz pawprint1_pawprint2_pawprint3_project
```

YOU MUST RUN *make clean* PRIOR TO SUBMITTING THE PROGRAM. Submission of object files or program executable will result in deducted points.

Grading

There are 100 points possible for this assignment. The grade breakdown is as follows:

- 10 points for correct design of Makefiles.
- 10 points for README, error checking and resource management.
- 10 points for general programming style and adherence to the constraints.
- 20 points for the project report.
- 15 points for correctly inputting the file.
- 35 points for outputting a well-formed allocation of applicants to departments.

Sample Input File

```
Vacancies and Departments

2 Ax
1 By

Job Applicants

John Doe
Jane Doe
Max Doe
Min Doe

Preferences Ax

John Doe
Jane Doe
Max Doe
Min Doe

Preferences By

Jane Doe
Min Doe
Max Doe
John Doe

Preferences John Doe

By
Ax

Preferences Jane Doe

Ax
By

Preferences Max Doe

Ax
By

Preferences Min Doe

By
Ax

END INPUT
```