Introduction to Algorithms	**Problem Set 5**
CS 4820 Spring 2016	**Due 11:59pm Thursday, March 17**

(1) *(10 points)* There are n jobs, n people, and for each person, you know the list of jobs that they could do. Now you want to match the jobs to the people such that every person does one job and every job has one person. Someone already computed a perfect matching M. Unfortunately, it turned out that the matching M has a mistake: an edge $(p, j) \in M$ matching a person p to a job j, but j is not on p's list. All other edges are OK.

Give an $O(m)$ algorithm for finding a new perfect matching if one exists, where m is the combined length of all lists, and you can assume all lists are non-empty, so $m \geq n$.

Examples:

- If $n = 3$, and the list for p_1 is $\{j_1, j_2\}$, the list for p_2 is $\{j_3\}$, and the list for p_3 is $\{j_3\}$, and the matching M is $p_1 \to j_1, p_2 \to j_2, p_3 \to j_3$, then the mistake is $p_2 \to j_2$ as p_2 doesn't do j_2, and there is no perfect matching since for the set of jobs $\{j_1, j_2\}$, only one person p_1 can do it.
- If $n = 3$, and the list for p_1 is $\{j_1, j_2\}$, the list for p_2 is $\{j_1, j_3\}$, and the list for p_3 is $\{j_3\}$, and the matching M is $p_1 \to j_1, p_2 \to j_2, p_3 \to j_3$, then the mistake is is $p_2 \to j_2$ as p_2 doesn't do j_2, but there exists a perfect matching. For example: $p_1 \to j_2, p_2 \to j_1, p_3 \to j_3$.

Your algorithm needs to use the network flow framework to solve this problem. You will need to construct the network. You will need to use $O(m)$ time and $O(m)$ memory to solve this problem. Please use an $O(m)$ size representation for your network, otherwise you will get Runtime Error (memory limit exceeded) in the test. For example, you should **not** define an array like $a[n][n]$, because that needs too much memory. n and m could be as large as the order of $20,000$ in the test data.

For the code, you must use the framework code (*Framework.java*) we provide on CMS. We will put the code you have submitted in the part of the framework that we have specified by the commented line
//YOUR CODE GOES HERE
and run the code.

We require you to submit both *Fragment.txt* and *Framework.java*. We will first test your program using *Fragment.txt*, and if it fails to compile, we will use your *Framework.java* file (which is a whole java program that **includes** your *Fragment.txt*, and works well on your own computer) to test instead. However, you will get decreased credit if your *Fragment.txt* file fails to compile, even if your *Framework.java* file passes all the test data.

Please read the provided framework carefully before starting to write your code. You can test your code with the test cases provided on CMS. *Framework.java* will take two command line arguments. The first one is the name of the input file, and the second is the name of the output file. The input file should be in the same folder in which your compiled java code is. After you compile and run your code, the output file will also be in the same folder. In order to test your code with the provided test cases, copy the test cases in the folder in which you have compiled your code, and set the name of the input file to be the name of one of the sample inputs (*SampleTest_i.txt* in which $0 \leq i \leq 5$). You can compare your output with the provided sample outputs (*SampleOutput_i.txt* in which $0 \leq i \leq 5$).

The format of the input file is the following:

- The first line has one number, n, which is the number of jobs and people. Jobs are labeled as $j_0, \cdots, j_{n-1}$ and people are $p_0, \cdots, p_{n-1}$.
- In the i-th line (let $i = 0, \cdots, n-1$) of the next n lines, the first number is s_i, which means the number of jobs that p_i does, followed by s_i numbers $k_1, \cdots, k_{s_i}$, denoting the index of those jobs.
- In the last line, there are n numbers, where the i-th number k_i (let $i = 0, \cdots, n-1$) means p_i is doing job j_{k_i} in the wrong matching M.

In the output file, there are two possibilities.

1. If there exists one perfect matching, you should output "Yes" in the first line, and then output n numbers in the second line, where the i-th number k_i (let $i = 0, \cdots, n-1$) means p_i is doing job j_{k_i} in your perfect matching.
2. If there does not exist one perfect matching, you should output "No" in the first line.

(2) *(10 points)* You are advising a growing startup that just outgrew their space in the building they have been for the last year. They are renting some additional space in a building a couple blocks away. The following principles have been agreed on for the move:

1. The company is arranged in groups, and no group should get split between two buildings.
2. we can think of the company Headquarters as its own group, and this group needs to remain in the old building.
3. There is one group who was most involved in negotiating rental of the new space, lets call them group A, they will move to the new rental.

Beyond these principles the move should be arranged to minimize total cost. While moving just group A is enough to have the rest fit in the old space, this may not be the optimal arrangement. Costs arise in a number of ways. If two groups collaborate a lot, there is a cost associated with breaking them up across buildings. For each pair of groups (B, C), we have an estimate c_{BC} for the cost incurred if B and C are not in the same building. Further, space in the new building needs to be rented, and depending on the size of each group C, there is a cost r_C for renting space for group C in the new building if they move. Give an algorithm to find the minimum cost way to split groups across the two buildings. You may assume that the numbers c_{BC} and r_C are integral, and the running time can depend polynomially on the number of groups and the maximum of the numbers c_{BC} and r_C.

(3) *(10 points total)* You are asked to help coordinate an international student exchange program among n universities $u_1, \ldots, u_n$. Each university u_i has exchange agreements with a subset L_i of the universities in the program. (We will assume that the agreements are symmetric, that is, if university u_i accepts exchange students from university u_j, then u_j also accepts exchange students from u_i.)

Next semester, every university u_i is willing to host a total of at most h_i exchange students. At each university u_i, there are g_i students who would like to participate in the exchange program next semester.

Given the numbers $g_1, \ldots, g_n$, $h_1, \ldots, h_n$, and the lists $L_1, \ldots, L_n$, the goal is to assign exchange students to suitable host universities so as to maximize the total number of participating students. (For simplicity, we will not consider preferences of the students for different universities.)

(a) *(6 points)* Present an algorithm for this problem. The running time of your algorithm may also depend on the numbers of the students, in addition to the number of universities n.

(b) *(4 points)* After presenting the solution to the exchange program coordinators, you are told that the solution is too "unbalanced", as too many of the exchange students are visiting universities in the same country. As part of the input, we also have the country that every university is in. Give an algorithm to assign exchange students to suitable host universities, maximizing the total number of participating students, with the additional requirement that of the g_i possible students participating from university u_i at most $g_i/3$ of them will do an exchange with a different university in the same country.
Hint: you may want to look at Solved Exercise 2 after Chapter 7 in the book.