**Assignment 3**

**Problem 1 (25 pts): Assigning jobs.**

The following is a common optimization problem. The student union of Antai College has 8 new recruits; it also has 8 job openings; each job can only take one student recruit, and each student recruit can only work one job. worker.txt has 8 lines, and each line has 8 numbers. Each line stands for a student, and each number in the line indicates the “efficiency” of that student working on each job. Write a program to assign jobs to students such that the efficiency is maximized. Print out the assignment and the maximum efficiency. (Hint: use itertools.permutations)

**Problem 2 (25 pts): Data format conversion between long and wide format.**

We frequently encounter the problem of converting data format between long and wide when doing panel data analysis. The first table below is in long format, and the second table is the same dataset in wide format.

|  |  |  |
| --- | --- | --- |
| household | year | income |
| 1 | 2009 | 1000 |
| 1 | 2010 | 2000 |
| 2 | 2009 | 3000 |
| 2 | 2010 | 4000 |

|  |  |  |
| --- | --- | --- |
| Household | Income2009 | Income2010 |
| 1 | 1000 | 2000 |
| 2 | 3000 | 4000 |

Sometimes the data we get is in long format, but we need it to be in wide format to do certain analyses; and vice versa.

In this problem, you are dealing with a dataset containing the scores for three exams for a class of students (exam.txt). The dataset is in long format; you task is to convert it to wide format.

Note that the fields are separated by Tab (i.e. .split(‘\t’)). You don’t need to write the output to a file. Simply print out the converted table to the screen.

Source format:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | gender | exam | scores |
| 1 | f | 1 | 45.0 |
| 1 | f | 2 | 33.0 |
| 1 | f | 3 | 29.5 |

Output format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | gender | scores1 | scores2 | scores3 |
| 1 | f | 45.0 | 33.0 | 29.5 |

**Problem 3 (25 pts): College admissions.**

Economists Alvin Roth and Lloyd Shapley were awarded 2012 Nobel Prize in Economics for their contribution to the matching theory. Matching theory deals with problems such as marriage, job choice, college admissions, etc. This time we will write our own Python program to deal with the matching problem of college admissions.

In this problem, we have 1000 students (with ID 1~1000) and 10 colleges (with ID 1~10), and each college has an admission quota of 10 students.. Each student can list four schools. We now have collected the preferences submitted by the students, stored in college.txt. Each line of the file consists of:

Student ID, college entrance exam score, first choice (第一志愿), second choice, third choice, and fourth choice. Fields are separated by space.

This problem uses the “sequential mechanism” (the mechanism used in Shanghai Gaokao prior to 2008), that is, every college prioritizes the admission of students who list it as their first choice, rather than students with higher test scores but list it as their second choice.

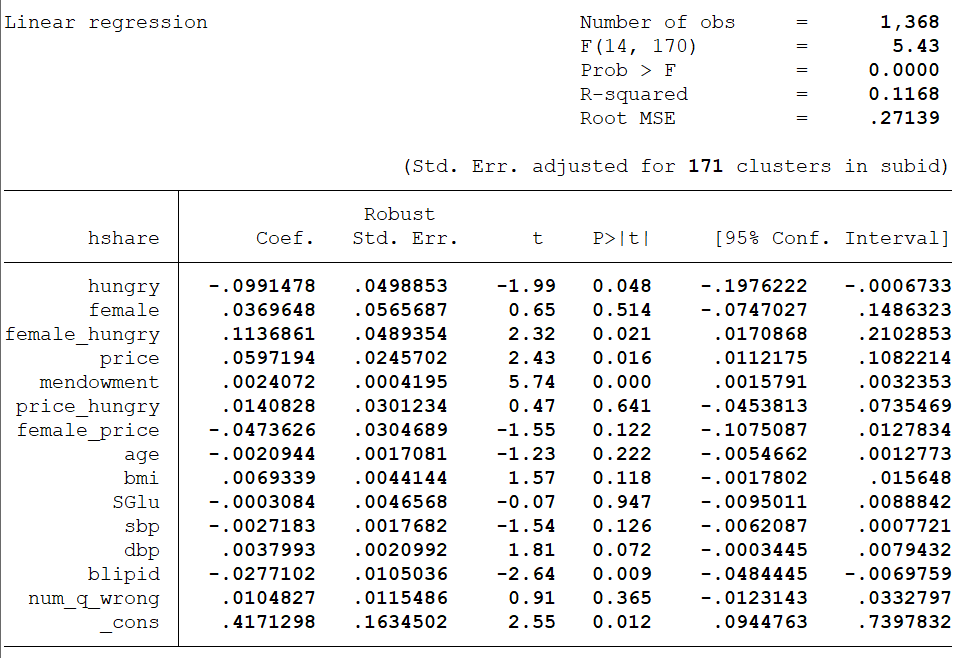
We assume every college has an admission quota of 10 students; that is, each college admits the top 10 students with highest exam scores and with this college as his first choice. The output should have 10 lines, each representing a college. Each line should include:

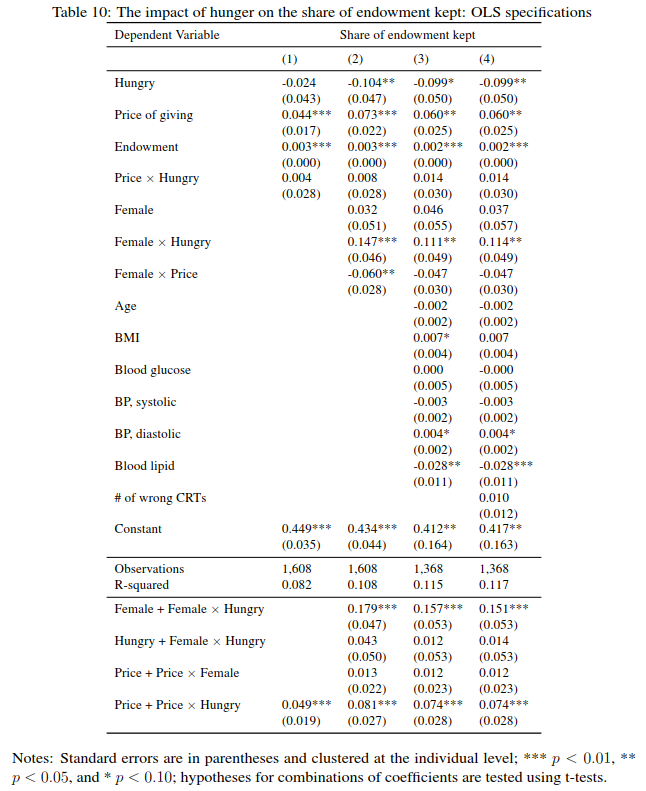
College ID, the number of students who list this college as their first choice, and cutoff score (录取分数线，the exam score of the 10th ranked admitted student). Fields should be separated by space.

Again, you don’t need to write the results to a file. Simply print out to the screen.

**Problem 4 (25 pts): Extracting useful information from regression reports.**

When writing research papers in (empirical) economics, often there is a long way to go from the output from statistical software to the actual tables in the paper. The next to figures show the difference:

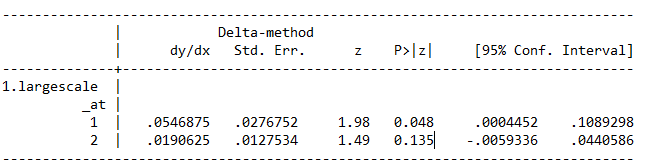




The first figure is the output from the software Stata. The second figure is how you present your results in your paper. We could save a lot of time and work by automating the extraction of useful data.

inteff.txt contains the output log of my non-linear regression analysis. In order to extract useful information out of it, we need the following steps:

First, find tables like this:



What’s to the left of the vertical line doesn’t have to be 1 or 2 (sometimes it can be Period or something else), but what’s to the right has to include 6 columns of data.

Next, you need to output the number in the dy/dx column, whose corresponding P>|z| is less than 0.01 (i.e. statistically significant).

To reduce your uncertainty, I include the reference output in inteff\_out1.txt. I recommend you use **regular expressions** for this task.

(Hint: briefly look through the file and look for patterns. It will significantly reduce your programming workload.)