Programming Design, Spring 2015
Homework 11

Instructor: Ling-Chieh Kung
Department of Information Management
National Taiwan University

To submit your work, please upload one PDF file for Problem 1 and two CPP files for Problems 2 and 3 to PDOGS at http://pdogs.ntu.im/judge/. Each student must submit her/his individual work. No hard copy. No late submission. The due time of this homework is ***8:00am, June 1, 2014***. Please answer in either English or Chinese.

Before you start, please read Chapters 15 and 18 of the textbook.[1] The TA who will prepare the solution for this homework is ***Tammy Chang***.

Problem 1

(30 points; 10 points each) Being able to search for information online and teach yourself new things is very important. This is especially true when you have a specific task that can be completed by some standard library functions. In this problem, let's try to do some practices.

(a) In `<cstdlib>`, there is a function `atoi()`. Go online to search for this function. Then use YOUR OWN WORDS to explain what this function does and how to use this function.

(b) In the class `string`, there is a member function `c_str()`. Go online to search for this function. Then use YOUR OWN WORDS to explain what this function does and how to use this function.

(c) You are given a text file "test.txt" containing several lines of integers between 0 and 100. Each number is a student's score. Without using the operator `>>`, write a C++ program that can find the maximum score.

Hint. One way is to combine `atoi()`, `c_str()`, and `length()`, where the last two are member functions of `class`.

Problem 2

(70 points) In this problem, let's warm up for the final project. Your task will be part of your task for the final project. you will be given a set of data, which is in one of the formats of the data for the final project. However, the scale is much smaller.

You are running a retail store. A customer comes in, selects some items, checks them out, and then the next customer comes in, and so on. You have experienced this for two years (from 2013/5/1 to 2015/4/30). Your point-of-sales (POS) information system records all the sales transactions in the past two years, including some basic information about consumers and what they purchase. The data are there, though you did not touch them in the past two years.

One day you suddenly realize that these historical sales data can help your business. In particular, you plan to use them to help you upsell, i.e., to recommend a consumer one additional item based on what she/he wants to buy. Instead of do it randomly, you want to utilize your sales data to make better recommendations, i.e., to recommend an item that is more likely to be purchased. Unfortunately, your POS system can only record transactions; it cannot give you suggestions on what to recommend. The good news is that you know computer programming. Now it is the time to implement your own recommendation system and increase your sales volume.

[1] The textbook is *C++ How to Program: Late Objects Version* by Deitel and Deitel, seventh edition.

Obviously, the first step to do data-driven recommendations is to be able to process and analyze past sales data. In this problem, you will be given past sales data in one specific format. You will write a program to read those data to know more about your past sales. More precisely, you need to find the most popular products among males, females, members, non-members, and all consumers for all months.

Input/output formats

In each transaction, one consumer buys one or multiple items. To make our lives easier, let's assume that no one buys more than one units of the same item in a transaction. Each item has a unique ID, which is an integer from 1 to 10. A customer may be a member or not. If she/he is a member, her/his member ID is recorded in a transaction as a string. Each member's ID is a string of English letters (uppercase or lowercase) and numbers of at most ten characters. The system also allows the clerk to enter the (estimated) gender and age of a consumer. The gender is recorded as a single letter: M for male, F for female, and O for others. The age is recorded as a single integer between 0 and 255. Finally, the date for that transaction is recorded in the YYYYMMDD format.

You have upgraded your POS system twice. Therefore, the system records the set of items purchased in three different ways. While in the project you will work with all three formats, in this problem we only work on one format, in which a Boolean vector $x = (x_1, x_2, ..., x_{10})$ is recorded with $x_i = 1$ meaning that item i is purchased. Below are some examples:

```
20130516,abcd1234,M,,1,0,0,1,0,1,1,0,0,0
20130516,,F,20,0,0,1,1,0,0,0,0,1,1
20130624,,,,0,0,0,0,1,1,1,0,0,1
```

Each row represents a transaction, where values are separated by commas. These values are the transaction date, member ID (if there is one), gender (if there is one), age (if there is one), and the ten Boolean values. Please note that in the second transaction, there is no member ID. This happens when the consumer is not a member. Please note that in the third transaction, the gender and age are also missing. This happens if the clerk does not enter her/his estimation into the POS system.

These transactions are provided to you in 24 text files, one for each month. In each file, the number of transactions is no greater than 1000. Though some values may be missing as described above, you may assume that all the information are recorded correctly.

In each of the 20 input files that your program will read, there is $k + 1$ lines. The first line contains a positive integer k that is no greater than 24. Each of the following line then contains a file name (including ".txt" as the extension file name) as a string with no white spaces. That is the file containing the sales data in one month. Your program should read the data contained in these file. Then for each of the following group, your program should find the list of items that are purchased for the most times: all male consumers, all female consumers, all members, all non-members, and all consumers. To print out a list of most popular items, print their IDs out from the smallest to the largest, separated with white spaces. Print out the five lists in five lines, one for each group in the above order. As always, there should be no white space after the last ID in each line.

As an example, suppose the input file contains

```
2
D:/file1.txt
D:/file8.txt
```

your program should first read data in those two files sequentially (all you need to do is to read a line of file path and name into a C++ string, convert it to a C string, and then use the C string to open a file connection). Suppose the data (in the two files) under consideration are

```
20130516,abcd1234,M,,1,0,0,1,0,1,1,0,0,0
20130516,abcd5566,F,,0,0,0,1,0,1,1,0,0,0
20130516,,F,20,0,0,1,1,0,0,0,0,1,1
```

```
20130624,,,,0,0,1,0,1,1,0,0,0,1
```

then the output should be

```
1 4 6 7
4
4 6 7
3 10
4 6
```

These are the lists of the most popular IDs for male consumers, female consumers, members, non-members, and all consumers, respectively.

What should be in your source file

Your .cpp source file should contain C++ codes that will both read testing data and complete the above task. For this problem, you are NOT allowed to use techniques not covered in lectures. You should write relevant comments for your codes.

Grading criteria

40 points for this program will be based on the correctness of your output. PDOGS will compile your program, feed testing data into your program, and check the correctness of your outputs. For each set of input data, if your program outputs correctly without violating the space limit, you get 2 points.

30 points for this program will be based on how you write your program, including the logic and format. Please try to write a robust, efficient, and easy-to-read program.

Problem 3 (bonus)

(30 points) Continue from Problem 2. In this problem. there will be 15 input files. In an input file, there are $k + 2$ lines. The first $k + 1$ lines are the same as those for Problem 2. The last line are two integers, i and j, where $1 \leq i \leq 10$, $1 \leq j \leq 10$, and $i \neq j$, separated by a white space. Now you need to calculate the conditional probability of purchasing item i given that a consumer purchases item j in the same transaction. The answer should be expressed as an integer for the first two digits (after natural rounding[2]) after the decimal point.

As an example, suppose the sales data under consideration are

```
20130516,abcd1234,M,,1,0,0,1,0,1,1,0,0,0
20130516,abcd5566,F,,0,0,0,1,0,1,1,0,0,0
20130516,,F,20,0,0,0,1,1,0,0,0,0,1,1
20130624,,,,0,0,1,0,1,1,0,0,0,1
```

and the given $i = 3$ and $j = 4$. In three transactions, item 4 is purchases. Among those three transactions, one also include item 3. Therefore, the conditional probability of purchasing item 3 given that item 4 is purchased is around 0.33, and the output should be

```
33
```

All the 30 points will be given based on the correctness of your program. For this problem, you may use any technique you like.

[2]For example, 0.345 rounds to 0.35 and 0.344 rounds to 0.34.