Final Project Due: Monday, Dec 5, 2011 (11:50 pm submitted electronically)

This project is worth 10% of your grade and will be graded out of 300 points, start ASAP and ask questions in the lab, or in the class as issues comes up or clarifications are needed)

Create a menu driven program to simulate a simple baking system. The program will open a file and reads the data and store them in an array/vector of objects to represent the bank customers information, including customer name, address, city, state, account number, and account balance. The format and a sample of the file are provided below. If you are using arrays make the size 20 and if there are more customers in the file print a message and only process the 20 that are stored in the array. Assume all the data stored in the file are correct, no need to verify them.

Data format for company:	**Sample Data file:**
Last name, First Middle initial Street Address State Zip code City Account Number(9 digit integer) Balance	Jones, Mary K. 56 Third St. # 125 CA 92067 Los Angeles 123456789 4350.78 Karlton Jr., Johnny Jim L. 785 N. Hope St. MN 90506 St. Paul 888777999 786.56

Implement the following menu:

```
    'O' -- Open an existing customer file.
    'A' -- Add new customer to the current list.
    'R' -- Remove customer from the current list.
    'P' -- Print all customers in the current list.
    'W' -- Make a withdraw.
    'D' -- Make a deposit.
    'T' -- Sort the list
    'X' -- Exit the program.
    Selection:
```

COMMENTS :

For O : Prompt the user for the file name, then open it.
Finally, read the entire file into the list.

For A : Add a customer to the bank.

For R : Remove a customer from the bank.

For P : Print all customers' information in the list.
This will display a sub-menu:

- L : Print in a page format to the monitor (refer to samples below)
- T : Print in a table format to the monitor(refer to sample below)
- F : Print in a table format to a file. Format is the same as T above.

For W and D: Prompt the user for the customer name. If name not found, stop.

for 'T' : Sort Customers. This will display a sub-menu:

'N' : Sort on names.
'C' : Sort on City.
'S' : Sort on State.
'B' : Sort on Balance.

For X : Make sure to save the list information to the disk. Print an ending message. Then halt the program.

Split your program into the following files: (We will discuss this in class.)

1. customer.h and Customer.cpp
2. Prog08.cpp

Required reports format:

I. **Sample reports table format:** A customer report in a tabular form: (Shown on the monitor)

```
Name                    Account Number          Balance
Jones, Mary K.              *****6789                   $1234.56
```

II. A detailed report in page format: (Print to monitor)

```
Name:                   Jones, Mary K.
Account number:         *****6789
Balance:                $2234.56
Address:                56 Third St. # 125
                        Los Angeles, CA  92067

******************************************************
Name:                   Karlton Jr., Johnny Jim L.
Account number:         *****7999
Balance:                $1834.60
Address:                785 N. Hope St.
                        St. Paul, MN 90506
```

III. Final report to the file to update the input file: (on exit)

```
Jones, Mary K.
56 Third St. # 125
CA 92067 Los Angeles
123456789  2234.56
Karlton Jr., Johnny Jim L.
785 N. Hope St.
MN 90506 St. Paul
888777999  786.56
```

Requirements:

I. Make sure the classes are separated into header, and implementation files.

II. Include all the necessary getters, setters, constructors and any helper functions you may need.
III. All classes must be documented, algorithms are required only for functions that are substantial and require explanation such as getters, which includes UML.
IV. Driver program must be modular, use as many functions as possible.
V. Must use arrays or vectors to store any list used in the program.
VI. You may assume there are no more than 20 customers are in the file, (but you may have to add new employees and also check for overflow).
VII. You may assume all names, and account numbers are unique.
VIII. You may use any platform to develop the program, but make sure it compiles and run on UNIX before submitting it.
IX. Sample data files customer.txt is posted on BB.
X. Make sure the program is user friendly.

Suggestions:

1) Start with an overall design, customer, list, and options (does not need to be detailed)
2) Design your UML for customer class, and then implement the class and a small main driver to test the class.
    a. Make sure you have all the necessary member functions:
        i. Getters
        ii. Setters
        iii. Constructors
        iv. Utilities (such as)
            1. Read
            2. Print
                a. Process (Withdraw, Deposit)
3) Create a list of customers and start implementing the menu options one by one and test them as you go using the small main test driver.
    a. Read into the list
    b. Print the list
    c. Process a customer
    d. Search list
    e. Sort list
4) Develop a menu function
5) Make the final main function

Additional Requirements for all CS or any CS combination(CECS, CSBA, CS Games) students . None CS students may implement as much as they want for extra credit.

1) Must use a vector
2) Create a class called bank to process the list of customers to make it easier to manage.
3) Search can be not case sensitive (lower case or upper case is not important)
4) Partial key search (if part of a search key is typed find all possible matches and let user chose the one they want).
5) Keeping track of number of transactions (withdrawals and deposits) for each account
6) Account info inquires. Print details of an account to the monitor using account number, prompt user for account number.
7) Adding different types of accounts (Saving, checking, credit cards, …)

Extra Credit for everyone: Add new features to the menu to improve the program, check with me first before doing anything. Think what other services a bank would provide.