

### HW3: Bank Accounts - using Classes with Member Functions and Separate Compilation:

You have been hired as a programmer by a major bank. Your first project is a small banking transaction system. Each account consists of a number and a balance. The user of the program (the teller) can create a new account, as well as perform deposits, withdrawals, balance inquiries, close accounts, etc.

For this assignment, **you must use Classes** and move functionality into the classes. Specifically, you should have at least the following classes:

1. A **Bank** class which consists of an **array of Accounts** and the **number of accounts** currently active in the bank.
- 2 An **Account** class which consists of a **Depositor, an account number, an account type, and a balance.**
- 3 A **Depositor** class which has a **Name and a social security number.**
- 4 A **Name** class which consists of **first and last names.**

You must add appropriate member functions (or methods) to each class so as to implement the functionality of each of the functions of the previous assignment (HW2). Each of the functions of HW2 should find a new identity as a class member. (You will have to decide as to which class each function belongs.) In addition, each class should minimally have a default constructor and possibly additional parametrized constructors. (Some of the HW2 functions may become constructors.) Also, consider whether any of your classes needs a destructor.

The data members of each class must be private. As such, you may need to provide accessor and mutator functions.

You must use **separate compilation** of your program modules: **Bank.h, Bank.cpp, Account.h, Account.cpp, Depositor.h, Depositor.cpp, Name.h, Name.cpp, as well as the module containing the main() function.**

As in previous assignments, initially, the account information of existing customers is to be read into the database. The bank can handle a maximum of **MAX\_NUM** accounts. The program keeps tracks of the actual number of currently active accounts. **A table of the initial database of active accounts should be printed.**

As before, the program then allows the user to select from the following menu of transactions:

Select one of the following:

- W - Withdrawal
- D - Deposit
- N - New account
- B - Balance
- I - Account Info
- C - Close Account
- Q - Quit

Once the user has entered a selection, appropriate functions should be called to perform the specific transaction. **At the end, before the user quits, the program prints the contents of the database.**

**As in previous assignments, make sure to use enough test cases so as to completely test program functionality.**