An Extreme Course Project

CSCI 3060U/SOFE 3980U – Winter 2016

Project Teams

You are to form a team of three people to design, implement, document and deliver a two-part software product. All phases to follow Extreme Programming philosophy as much as it applies - in particular,

- continuously maintained test suites as requirements and quality control
- pair programming of all code
- simplest possible solution to every problem
- continuous redesign and rearchitecting
- automation in testing and integration
- frequent integration and complete releases

Every two weeks (or so, see the schedule below) you will deliver concrete evidence of your team's progress as required by project assignments.

Project Phases

The project will be done in six phases, each of which will be an assignment. Phases will cover steps in the process of creating a quality software result in the context of an Extreme Programming process model.

Assignments will be on the quality control aspects of requirements, rapid prototyping, design, coding, integration and analysis of the product you are building. Throughout the project, you should keep records of all evidence of your product quality control steps and evolution, in order to make the marketing case that you have a quality result at the end of the course.

Your final products will be tested and evaluated.

Project Schedule

The course project consists of six assignments, with separate handouts for each one.

Assignments are scheduled to be due as follows:

A1. Monday, Feb. 7	A4. TBD
A2. TBD	A5. TBD
A3. TBD	A6. TBD

Assignment Hand-Ins

- Unless otherwise specified, all assignments *must* be handed in through Blackboard by 4pm on the due date.

For all submissions indicate clearly your team name and all member names and on every hand in.

Project Requirements

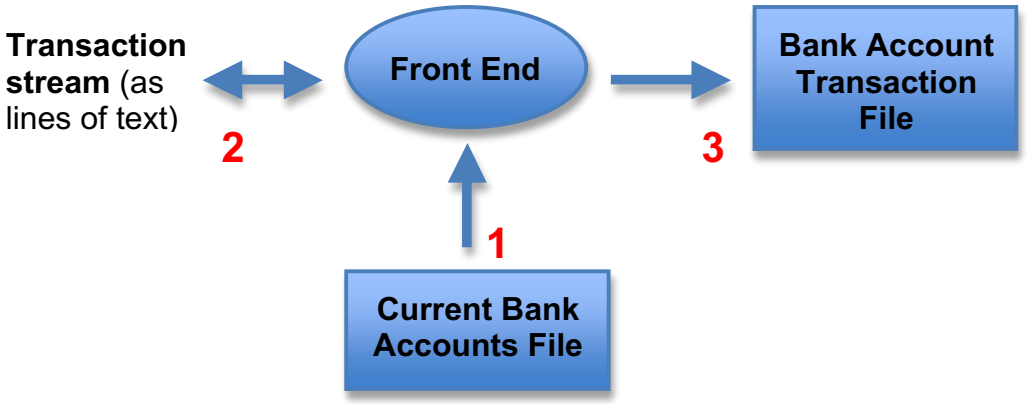
The product you are to design and build is a Banking System. The system consists of two parts:

- the Front End, an automated teller machine terminal for simple banking transactions
- the Back End, an overnight batch processor to maintain and update a master banking account file

Both parts will be run as console applications, that is, they are to be invoked from a command line and use text and text file input/output only (this is an important requirement for assignments later in the project, so don't ask for exceptions)

The Front End

The Front End reads in a file of bank accounts that contains information on each account including balance and account owner. The Front End (1), processes a stream of bank account transactions one at a time (2), and writes out a file of banking account transactions at the end of the session (3).



Informal Customer Requirements for the Front End

The Front End handles a sequence of banking transactions, each of which begins with a single transaction code (word of text) on a separate line

The Front End must handle the following transaction codes:

login	- start a Front End session
withdrawal	- withdraw money from a bank account
transfer	- transfer money from one bank account to another
paybill	- pay a bill from a bank account
deposit	- deposit money into a bank account
create	- create a new bank account (privileged transaction)
delete	- delete a bank account (privileged transaction)
disable	- disable all transactions in a bank account (privileged transaction)
changeplan	- change the transaction plan of a bank account (privileged transaction)
logout	- end a Front End session

Transaction Code Details

login - start a Front End session

- should ask for the kind of session, which can be either:
  - standard, which means bank account user mode, or
  - admin, which means privileged mode (e.g., bank employee)
- If the kind of session is standard, should ask for the account holder's name
- after the kind (and optional name) is accepted, reads in the current bank accounts file (see below) and begins accepting transactions
- Constraints:
  - no transaction other than login should be accepted before a login
  - no subsequent login should be accepted after a login, until after a logout
  - after a standard login, only unprivileged transactions are accepted
  - after an admin login, all transactions are accepted

logout - end a Front End session

- should write out the bank account transaction file (see below) and stop accepting any transactions except login
- Constraints:
  - should only be accepted when logged in
  - no transaction other than login should be accepted after a logout

withdrawal – withdraw money from an account

- should ask for the account holder's name (if logged in as admin)
- should ask for the account number (as a text line)
- then should ask for the amount to withdraw
- should save this information for the bank account transaction file
- Constraints:
  - Bank account must be a valid account for the account holder currently logged in.
  - Maximum amount that can be withdrawn in current session is $500.00 in standard mode
  - Account balance must be at least $0.00 after withdrawal

transfer – transfer money between two accounts

- should ask for the account holder's name (if logged in as admin)
- should ask for the account number that money will be transferred from (as a text line)
- should ask for the account number that money will be transferred to (as a text line)
- then should ask for the amount to transfer
- should save this information for the bank account transaction file
- Constraints:
    - Bank account that the funds are being transferred from must be a valid account for the account holder currently logged in.
    - Bank account that the funds are being transferred to must be a valid account in the Bank System.
    - Maximum amount that can be transferred in current session is $1000.00 in standard mode
    - Account balance of both accounts must be at least $0.00 after transfer

paybill – pay bill with money from an account

- should ask for the account holder's name (if logged in as admin)
- should ask for the account number (as a text line)
- should ask for the company to whom the bill is being paid
- should ask for the amount to pay
- should save this information for the bank account transaction file
- Constraints:
    - Bank account must be a valid account for the account holder currently logged in.
    - The company to whom the bill is being paid must be "The Bright Light Electric Company (EC)", "Credit Card Company Q (CQ)" or "Low Definition TV, Inc. (TV)"
    - Maximum amount that can be paid to a bill holder in current session is $2000.00 in standard mode
    - Account balance must be at least $0.00 after bill is paid

deposit – deposit money into an account

- should ask for the account holder's name (if logged in as admin)
- should ask for the account number (as a text line)
- then should ask for the amount to deposit
- should save this information for the bank account transaction file
- Constraints:
    - Bank account must be a valid account for the account holder currently logged in.
    - Deposited funds should not be available for use in this session

create - create a new bank account with an initial balance

- should ask for the name of the account holder (as a text line)
- then should ask for the initial balance of the account
- should save this information to the bank account transaction file
- Constraints:
    - privileged transaction - only accepted when logged in admin mode
    - bank account numbers must be unique in the Bank System
    - new account holder name is limited to at most 20 characters
    - account balance can be at most $99999.99
    - this account should not be available for other transactions in this session

delete – delete a bank account

- should ask for the bank account holder's name (as a text line)
- should ask for the account number (as a text line)
- should save this information for the bank account transaction file
- Constraints:
    - privileged transaction - only accepted when logged in admin mode
    - Account holder's name must be the name of an existing account holder
    - Account number must be the number of the account holder specified

    - no further transactions should be accepted on a deleted account

disable – disable a bank account

- should ask for the bank account holder's name (as a text line)
- should ask for the account number (as a text line)
- should change the bank account from active (A) to disabled (D)
- should save this information for the bank account transaction file
- Constraints:
    - privileged transaction - only accepted when logged in admin mode
    - Account holder's name must be the name of an existing account holder
    - Account number must be the number of the account holder specified
    - no further transactions should be accepted on a disabled account

changeplan – change the transaction payment plan for an account

- should ask for the bank account holder's name (as a text line)
- should ask for the account number (as a text line)
- should set the bank account payment plan from student (SP) to non-student (NP)
- should save this information for the bank account transaction file
- Constraints:
    - privileged transaction - only accepted when logged in admin mode
    - Account holder's name must be the name of an existing account holder
    - Account number must be the number of the account holder specified

General Requirements for the Front End

The Front End should never crash or stop except as directed by transactions.

The Front End cannot depend on valid input - it must gracefully and politely handle bad input of all kinds (note: but you can assume that input is at least lines of text).

Bank Account Transaction File

At the end of each session, when the logout transaction is processed, a bank account transaction file for the day is written, listing every transaction made in the session.

Contains fixed-length (40 characters) text lines of the form:

```
CC_AAAAAAAAAAAAAAAAAAA_NNNNN_PPPPPPPP_MM
```

Where:

`CC`
: is a two-digit transaction code, 01-withdrawal, 02-transfer, 03-paybill, 04-deposit, 05-create, 06-delete, 07-disable, 08-changeplan, 00-end of session

`AAAAAAAAAAAAAAAAAAA`
: is the account holder's name

`NNNNN`
: is the bank account number

`PPPPPPPP`
: is the amount of funds involved in the transaction (in Canadian dollars)

`MM`
: is any additional miscellaneous information that is needed in the transaction but does not fit in any of the other fields.

`_`
: is a space

Constraints:

- every line is exactly 40 characters (plus newline)
- numeric fields are right justified, filled with zeroes (e.g., 00023 for bank account 23)
- alphabetic fields are left justified, filled with spaces (e.g. John_Doe_____ for bank account holder John Doe)
- unused numeric fields are filled with zeros (e.g., 0000)
- In a numeric field that is used to represent a monetary value, ".00" is appended to the end of the value (e.g. 00110.00 for 110)
- unused alphabetic fields are filled with spaces (blanks) (e.g., _____)
- the sequence of transactions ends with an end of session (00) transaction code

Current Bank Accounts File

Consists of fixed length (37 characters) text lines in the form:

```
NNNNN_AAAAAAAAAAAAAAAAAAA_S_PPPPPPPP
```

Where:

`NNNNN`
: is the bank account number

`AAAAAAAAAAAAAAAAAAA`
: is the account holder's name

`S`
: is the bank account status – active (A) or disabled (D)

`PPPPPPPP`
: is the current balance of the account (in Canadian dollars)

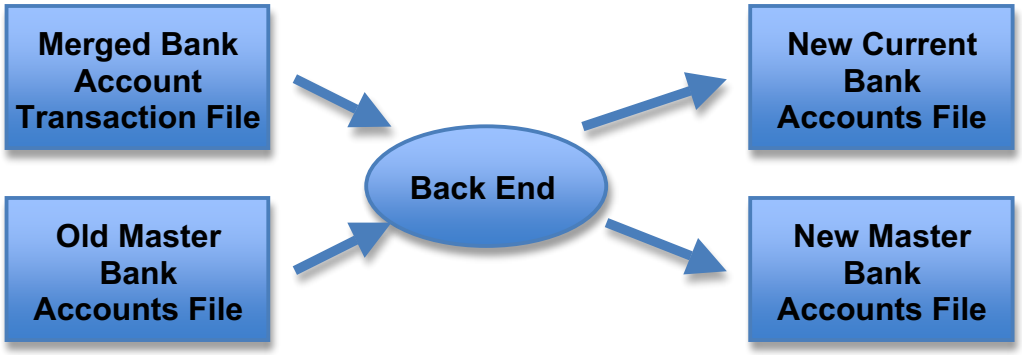
`_`
: is a space

Constraints:

- every line is exactly 37 characters (plus newline)
- numeric fields are right justified, filled with zeroes (e.g., 00023 for bank account 23)
- alphabetic fields are left justified, filled with spaces (e.g. John_Doe_____ for bank account holder John Doe)
- unused numeric fields are filled with zeros (e.g., 0000)
- In a numeric field that is used to represent a monetary value, ".00" is appended to the end of the value (e.g. 00110.00 for 110)
- unused alphabetic fields are filled with spaces (blanks) (e.g., _____)
- file ends with a special bank account with account holder's name END_OF_FILE.

The Back End

The Back End reads in the previous day's master bank accounts file and then applies all of the bank account transactions from a merged set of bank account transaction files to the old master bank accounts file to produce the new master bank accounts file. Because transactions may also create or delete bank accounts, it also produces a new current bank accounts file for tomorrow's Front End runs.



Informal Customer Requirements for the Back End

The Back End reads the Master Bank Accounts File (see below) and the Merged Bank Account Transaction File (see below) and applies all transactions to the Old Master Bank Accounts File to produce the New Master Bank Accounts File and the new Current Bank Accounts File.

The Back End also calculates the daily cost of transaction based on the account plan. If the account plan is a student plan the account is debited $0.05 for each transaction. If the account plan is a non-student plan, the account is debited $0.10 for each transaction.

The Back End enforces the following business constraints, and produces a failed constraint log on the terminal as it processes transactions from the merged back account transaction file.

Constraints:

- no bank account should ever have a negative balance
- a newly created bank account must have an account number different from all existing bank accounts

General Requirements for the Back End

The Back End should assume correct input format on all files, and need not check for bad input. However, if by chance it notices bad input, it should immediately stop and log a fatal error on the terminal.

Back End Error Recording

All recorded errors should be of the form: `ERROR: <msg>`

- For failed constraint errors, `<msg>` should contain the type and description of the error and the transaction that caused it to occur.
- For fatal errors, `<msg>` should contain the type and description and the file that caused the error.

The Merged Bank Account Transaction File

The Merged Bank Account Transaction File is the concatenation of any number of bank account transaction files output from Front Ends of the Banking System, ended with an empty one (one containing no real transactions, just a line with a 00 transaction code).

The Current Bank Accounts File

A file containing every current bank account (in the New Master Bank Accounts File) and the appropriate information as defined in the format described for the Front End.

The Master Bank Accounts File

The Master Bank Accounts File consists of fixed-length (42 characters) lines of the form:

```
NNNNN_AAAAAAAAAAAAAAAAAAAA_S_PPPPPPPP_TTTT
```

Where:

`NNNNN`
: is the bank account number

`AAAAAAAAAAAAAAAAAAAA`
: is the account holder's name

`S`
: is the bank account status – active (A) or disabled (D)

`PPPPPPPP`
: is the current balance of the account (in Canadian dollars)

`TTTT`
: is the total number of transactions

`_`
: is a space

Constraints:

- every line is exactly 42 characters (plus newline)
- numeric fields are right justified, filled with zeroes (e.g., 00023 for bank account 23)
- alphabetic fields are left justified, filled with spaces (e.g. John_Doe_____ for bank account holder John Doe)
- unused numeric fields are filled with zeros (e.g., 0000)
- In a numeric field that is used to represent a monetary value, ".00" is appended to the end of the value (e.g. 00110.00 for 110)
- unused alphabetic fields are filled with spaces (blanks) (e.g., _____)
- the Master Bank Accounts File must always be kept in ascending order by bank account number