INF20013/70004 Business Systems Programming

Swinburne University of Technology
Faculty of Business & Enterprise

ASSIGNMENT 1

Assignment Value: 10% of your final mark
The assignment must be done in pairs

Due Date:
8:30am on Monday 25th of August 2014
Note: Late submissions will attract a penalty
(Late penalty: 10% per day)

Team Registration on ESP
You must register your team on ESP no later than Friday 15th of August 2014
(Students that have not registered by this date will be allocated to a team)

Submission Requirements
You must submit all source code, all executables and any other required files or folders in a single ZIP file via the ESP system.

https://esp.ict.swin.edu.au/

The name of the .ZIP file must use this naming convention:
1111111_222222_Ass1.ZIP
(where 111111 and 222222 are student ids of a group submitting the assignment)

DO NOT USE RAR FILES!!!

Tutor Demonstration
You must demonstrate your application to your tutor during your tutorial in Week 4.
Students that fail to demonstrate their application will receive a mark of 0.

(Note: You must have submitted your assignment via ESP prior to this)

Testing your software prior to submission and demonstration
Prior to submitting your assignment, please try to install your assignment on a different PC to where you have developed your system.

UnZip the assignment to a different path than where you have developed your system. Then test your system fully.

The purpose of this assignment is to demonstrate the knowledge and skills required to read and write various types of sequential files via a VB.NET (Visual Basic.NET) program.

Description of the Assignment:

Although there are several commercial programs available for personal financial management, they include so many bells and whistles that their original purpose - that is keeping track of transactions and reporting balances has been obscured. The program required in this assignment is aimed specifically to record financial transactions (referred as deposits and withdraws). Adding a reconciliation feature would be easy enough, although it is not in the scope of this assignment.
It is also important to keep in mind that the key objective of this assignment is to demonstrate the knowledge and skills required to read and write various types of sequential files via a VB.NET program. Developing a commercial level accounting system is not required here.

The overall assignment is decomposed into the following tasks:

Task 1 - Files Creation:

Create six files in a folder name C:\temp\A1-[your student id]
For instance, if your student id is 1234567; the name of the folder should be C:\temp\A1-1234567. If you are working in a pair, the student id that comes first in an ascending order should be used as a folder name.

Name of these files should be as follows:

1. CommaFile.txt
2. TabFile.txt
3. FixedFile.txt
4. RanFile.RAN
5. EncryptFile.txt
6. BinaryFile.txt

Task 2 – User Interface:

With VB .Net, create a windows form similar to the one given in Figure 1. It is suggested but not required to design exactly the same form as given in Figure 1. You can design a user interface according to your own choice. However, it is required that the interface should offer the following functionalities (see Figure 1 for reference).

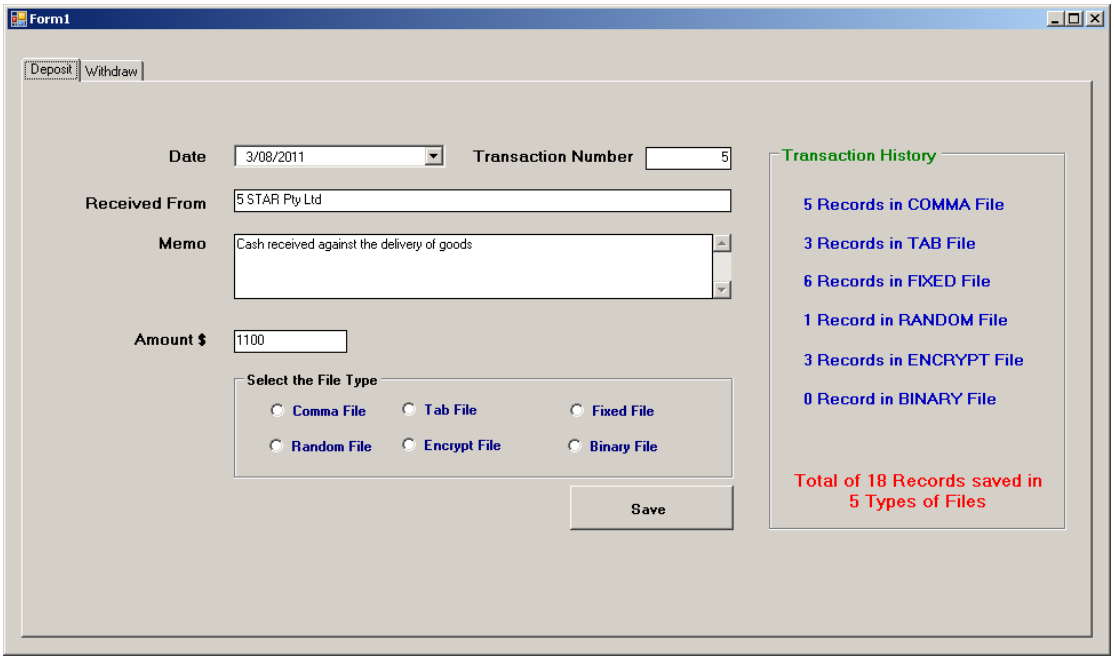

Figure 1: DEPOSIT Form

Task 2.1 DEPOSIT Form:

This (part of the) form is used to record the amounts deposited (stored) in a given file along with the following data:

- Transaction type: String value = "Deposit"
- Date: Date value (in DD/MM/YYYY format)
- Transaction number: Number value
- Received From: String value
- Memo: String value
- Amount: Number value

Task 3 Select the File Type

Once user enters this data, a type of file such as Comma, Binary, Tab etc would be selected from the available options.

Task 3.1 Save Data

When user clicks on Save button, the data should be saved in the following way:

- For Comma File, the values should be stored in the CommaFile and each piece of data in this file is delimited by comma (",")
- For Tab File, the values should be stored in TabFile and each piece of data in this file is delimited by a Tab.
- For Fixed File, the values should be stored in FixedFile and each piece of data in this file is stored in a fixed width format:
    - Transaction type: 8 characters
    - Date: 15 characters
    - Transaction number: 3 digits (no decimal)
    - Received From: 30 characters
    - Amount: 8 digits (no decimal)
    - Memo: 200 characters

- For Random file, the values should be stored in RanFile.RAN and each piece of data in this file is delimited by a mix of either a colon (":") or semicolon (";")
- For Encrypt file, the values should be stored in EncryptFile.txt and each piece of data in this file is delimited by comma - Similar to comma file. However, you need to encrypt the data.

  Tip:
  - Every odd numbered character (1,3,5,...) has had its ASCII value decreased by 1
    (So the letter 'd' has become 'c')
  - Every even numbered character (0,2,4,6,...) has had its ASCII value increased by 1. (So the letter 'h' has become 'i')
- For binary file, the values should be stored in BinFile.txt and should be saved in binary data format

Task 3.2 Update Transaction History

Once you have stored the data in the relevant file, update the transaction history.

Task 4 WITHDRAW Form

The purpose of this form is to display the values stored in the sequential files and to withdraw some amounts as shown in Figure 2.

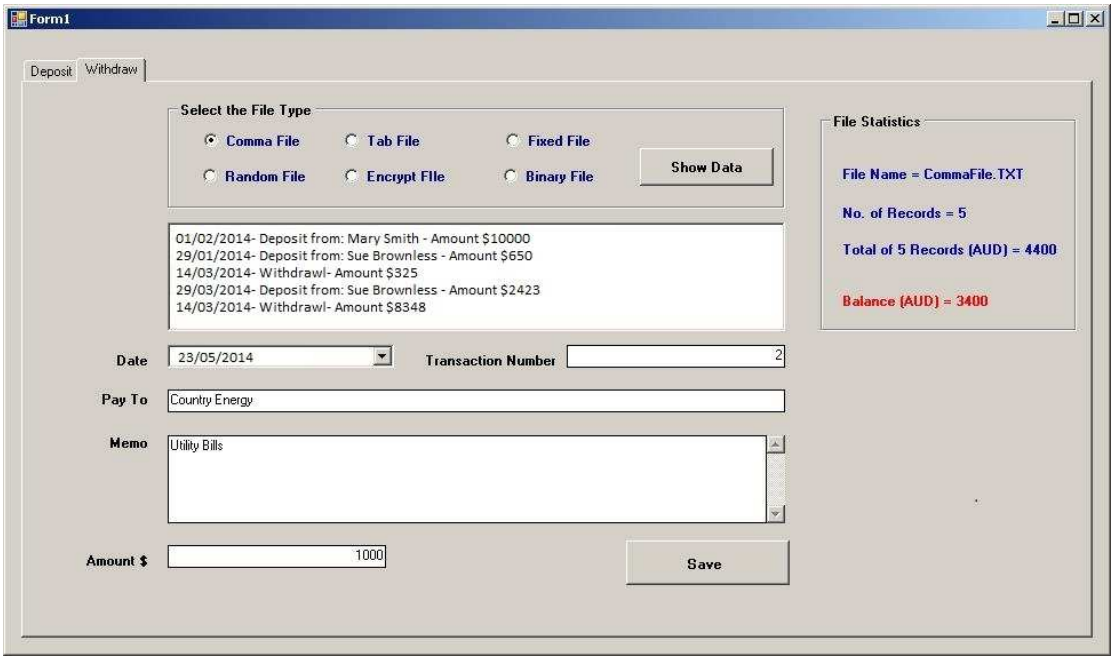

Figure 2: WITHDRAW Form

Task 2.2.1 Select the File Type

To start with, user needs to select a type of file that is required to be displayed; such as Comma File, Tab File etc.

Task 4.1

After selecting a file type, user clicks on Show Data button, the contents of the selected file should be displayed in the list box. Regardless to the type of file selected, the contents of all files should be displayed in the following format:

Date(in DD/MM/YYYY format) - [space][Transaction type] [space]{from: [received from][space]} - [space]Amount $[amount]

Each transaction/record in the file should be displayed on a separate line.
For instance:

```
01/02/2014- Deposit from: Mary Smith - Amount $10000
29/01/2014- Deposit from: Sue Brownless - Amount $650
14/03/2014- Withdrawl- Amount $325
29/03/2014- Deposit from: Sue Brownless - Amount $2423
14/03/2014- Withdrawl- Amount $8348
```

Task 4.2 Data Entry

User should enter data for Date, Transaction number, Pay To, Memo and Amount on Withdrawal form.

Task 4.2 Save & Update Statistics

Once user clicks on Save button, the program should save the data (same format as deposits) and display the statistics shown in Figure 2.

Assumptions and suggestions:

- On each fresh start of the program, all the data in the existing files should be deleted.
- It is assumed that all the required data will be entered in Deposit & Withdrawal Form – within their logical constraints
- Data validation is not required

Marking Guide

Total of 120 marks are allocated to this assignment. Marks will be deducted where portions of your submission

- have not been implemented
- do not work properly or do not meet specifications

The breakdown of marks allocated to various parts of the assignment is given in a table 1 below:

Table 1: Marking Guide for Assignment 1

Requirement, Function or Specification	Allocated Marks
Use of appropriate controls for data entry – Deposit and Withdraw Forms	5
Deposit	
Writing data into the CommaFile	5
Writing data into the TabFile	5
Writing data into the FixedFile	5
Writing data into the EncryptFile	5
Writing data into the RanFile	5
Writing data into the BinaryFile	5
Correct data for Transaction History	10
Withdrawal	
Reading data from the CommaFile	5
Reading data from the TabFile	5
Reading data from the FixedFile	5
Reading data from the EncryptFile	5
Reading data from the RanFile	5
Reading data from the BinaryFile	5
Display of file contents in a correct format	5
Correct Data for Transaction History	10
Withdrawal data saves correctly	15
Style	
Well named variables and/or procedures and comments used widely throughout code	15
Code Structure: High cohesion, low coupling:- reusable code	30
Total	**150**