COMP1038 Coursework 02 – Digital Mini Banking System

Introduction

This is the COMP1038 Coursework 02. It is worth **40% of the module mark**. The deadline for this exercise is **16:00 on Saturday 26th of December 2020**.

Read the entire document before beginning the exercise.

If you have any questions about this exercise, please ask in the Q&A forum on Moodle, after a lecture, in a lab, or during the advertised office hours. Do not post your program or parts of your program to Moodle as you are not allowed to share your coursework programs with other students. If any questions require this exercise to be clarified then this document will be updated and everyone will be notified via Moodle.

Version History

- Version 2.0 - 2020-12-04 – version 2.

Submission

You must submit a single C source code file containing all your code for this exercise and a text file containing city names, branch names, and their codes. The files must be called `DigitalMiniBank_<Your_Name>_<Your Numeric Id>.c` and `CityBranchCodes_<Your_Name>_<Your Numeric Id>.txt`. The program must not requires any other files outside of the standard C headers which are always available. The first line of the C source code file should be a comment which contains your student ID number, username, and full name, of the form:

```
// 6512345 zy12345 Joe Blogs
```

The file must compile without warnings or errors when I use the command

```
gcc -std=c99 -lm -Wall DigitalMiniBank_<Your_Name>_<Your Numeric Id>.c  -o MiniBankAccount
```

This command will be run on our Linux server `cslinux`. If it does not compile, for any reason, then you will lose all the marks for testing (common reasons in the past have been submitting a file with the wrong filename, or developing your solution on your personal computer without having tested it on our Linux server). If the file compiles but has warnings then you will lose some marks for not correcting the warnings.

The completed source code file should be uploaded to the Coursework Submission link on the COMP1038 Moodle page. You may submit as many times as you wish before the deadline (the last submission before the deadline will be used). After the deadline has passed, if you have already submitted your exercise then you will not be able to submit again. If you have not already submitted then you will be allowed to submit **once**.

Late submissions: Late submissions will lose 2 percentage points **per hour**, rounded up to the next whole hour. This is to better represent the large benefit a small amount of extra time can give at the end of a programming exercise. No late submissions will be accepted more than 50 hours after the exercise deadline. If you have extenuating circumstances you should file them before the deadline.

Plagiarism

You should complete this coursework on your own. Anyone suspected of plagiarism will be investigated and punished in accordance with the university policy on plagiarism (see your student handbook and the University Quality Manual). This may include a mark of zero for this coursework.

You should write the source code required for this assignment yourself. If you use code from other sources (books, web pages, etc), you should use comments to acknowledge this (and marks will be heavily adjusted down accordingly). *The only exception to this is the dynamic data-structures (linked lists and others) developed during the lectures and tutorials; you may use these, with or without modification, without penalty as long as you add a comment in your program saying you have taken them from the lectures or tutorials and saying how you have modified it (or not modified it). If you do not acknowledge their source in a comment then it will be regarded as potential plagiarism.*

You must not copy or share source code with other students. You must not work together on your solution. You can informally talk about higher-level ideas but not to a level of detail that would allow you all to create the same source code.

Remember, it is quite easy for experienced lecturers to spot plagiarism in source code. We also have automated tools that can help us identify shared code, even with modifications designed to hide copying. If you are having problems you should ask questions rather than plagiarize. If you are not able to complete the exercise then you should still submit your incomplete program as that will still get you some of the marks for the parts you have done (but make sure your incomplete solution compiles and partially runs!).

If I have concerns about a submission, I may ask you to come to my office and explain your work in your own words.

Marking

The marking scheme will be as follows:

- *Tests (60%):* Your program should correctly implement the task requirements. A number of tests will be run against your program with different input data designed to test if this is the case for each individual requirement. The tests themselves are secret but general examples of the tests might be:
    - Does the program work with the example I/O in the question?
    - Does the program work with typical valid input?
    - Does the program correctly deal with input around boundary values?
    - Does the program correctly deal with invalid input (both invalid files and values)?
    - Does the program handle errors with resources not being available (eg, malloc failing or a filename being wrong)?
    - Does the program output match the required format?
    - Does the program output an appropriate table when required?

    As noted in the submission section, **if your program does not compile then you will lose all testing marks**.

- *Appropriate use of language features (30%):* Your program should use the appropriate C language features in your solution. You can use any language features or techniques that you have seen in the course, or you have learned on your own, as long as they are appropriate for your solution. Examples of this might be:
    - If you have many similar values, are you using arrays (or equivalent) instead of many individual variables?
    - Have you broken your program down into separate functions?
    - Are all your function arguments being used?
    - If your functions return values, are they being used?
    - If you have complex data, are you using structures?
    - Are you using loops to avoid repeating many lines of code?
    - Are your if/switch statements making a difference, or are the conditions always true or false making the statement pointless?
    - Are you closing files when the file is no longer in use?
- *Source code formatting (10%):* Your program should be correctly formatted and easy to understand by a competent C programmer. This includes, but is not

limited to, indentation, bracketing, variable/function naming, and use of comments.

Task

Nowadays, bank accounts have become an important part of human life for saving money easy and safe way. Due to the recent growth of digitization, most of the banks are maintaining the accounts of their customers on a computerized system. As a banking software developer, your task is to develop a computerized banking system.

When the program runs, it should display the following main menu to the user and prompt them to enter a menu option:

```
1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:
```

If the user enters the number of an option, the program should perform that option then return to the main menu to let the user select another option. If the user enters something which is not a valid option then the program should print "`Unknown option.`" then print "`Option: `" again for the user to select another option. The user may enter any input at this, or any other prompt, in the program, terminated by pressing the return key (newline character). Your program must deal with this appropriately, accepting valid input and rejecting invalid input according to the particular prompt.

If the user selects "`Show city code and branch code`", then the program will show the list of city and city codes, and a prompt "`Enter the city code:`". When the user enters a city code from the list, it will display list of branches within that city and their codes. For doing this, the students need to prepare a text file containing city names and branch names and their codes. The file shoud contains at least 10 city names, 5 branch name from each city, and their respective codes.

If the user selects "`Open an account`", then the program should ask the user to input flowing data.

Description	Type	Size	Remarks
Name	String	100 characters	
Id Number	String	15 Characters	Id can be alphanumeric
Address	String	200 Characters	
Date of Birth	String	10 Characters	YYYY/MM/DD
Initial deposit	Integer		not less than 1000 RMB

After inputting these data of an account the account will be created and a numeric account number will be assigned by the program in the format of xxxxxxxxxx. The first xxx is the city code which is a number within 001-999 from the list of city codes, the second xx is the branch code which is a number within 01-99 from list of branch codes, and the third xxxxx is the account number which is a number within 00001-99999. Account number will be assigned in sequence starting with 00001. With the same city code and same branch code, two account numbers can't be the same. Today's date (system date) will be associated with the account by the program as the account opening date.

If the user selects "`Show account details`", then the program should ask the user to input the account number and the program will display all the information about the account.

If the user selects "`Show list of accounts`", then the program should ask the user to input the city code and the branch code. The program will display a table as follows.

Account Number	Name	Id Number	Balance

If the user selects "`Deposit in an account`", then the program should ask the user to input the account number and show the details of the account and prompt for input the amount in integer by the user. After input, the amount will be added with the corresponding accounts' balance and display the new account balance.

If the user selects "`Withdraw from an account`", then the program should ask the user to input the account number and show the details of the account and prompt for input the amount in integer by the user. After input, the amount will be deducted from the corresponding accounts' balance and display the new account balance. If the withdrawal amount is more than the balance in the account the program will show 'Withdrawal amount can't be more than the balance in the account' and ask for the withdrawal amount again.

If the user selects "`Transfer money`", then the program should ask the user to input the account number of the debiting account and display the name, Id, and balance of the

account. Thereafter, the program will ask the user to the input account number of the crediting account and display the name, Id, and balance of the account. Then the program will ask the user to input the transfer amount. If the transfer amount is more than the balance in the debiting account the program will display “Transfer amount can’t be more than the balance of the debiting account” and the program will ask the user to input a new transfer amount. Thereafter, if the transaction is successful, the transfer money will be deducted from the balance of the debiting account and will be added with the balance of the crediting account and show the message “Transaction successful”. Then the program will keep a record of this transaction by storing the Debiting account number, Crediting account number, transfer amount, and the date of the transaction.

If the user selects “`Transaction details`”, then the program should ask the user to input the account number, start date, and end date. The dates will be input in YYYY/MM/DD format. If the end date is more recent than the start date the program will display “End date can’t be later than the start date”. Else the program will display a list of all transactions between the start and end date in the form of a table as follows.

Date	Debiting account	Crediting account	Transfer amount

Records in the table will be arranged in ascending order based on date. i.e. the record for the most recent transaction will be on the top of the table.

If the user selects “`Close an account`”, then the program should ask the user to input an account number and display the account details. Thereafter, it will show a prompt for the confirmation from the user for closing the account. If the user confirms to close the account, the corresponding records of the account will be moved from the file holding the information of the list of the accounts and move to the list of the closing accounts file. After closing an account the corresponding account number will not be given to any other user.

If the user selects “`Quit`”, then the program should terminate. Before exiting, the program should explicitly free any main memory it has allocated that it hasn't yet freed. (Note that most operating systems will recover all unfreed memory from a program when it exits but this exercise requires you to free allocated memory yourself in order to demonstrate you can do it). The program also closes all the open files before existing.

After every operation, the program will ask the user if the user wants to do the same operation again or return to the main menu. The program will proceed with its execution according to the users' decision.

Example input/output

```
1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:1
```

City	code	City	code	City	Code
Ningbo	001	Beijing	002	Chongqing	003
Shanghai	004	Tianjin	005	Anqing	006
Bengbu	007	Bozhou	008	Chaohu	009

```
Enter the city code: 001
```

Branch name	code	Branch name	code	Branch name	Code
Yinzhou	01	Fenghua	02	Ninghai	03
Xiangshan	04	Beilun	05	Haishu	06
Jiangbei	07	Zhenhai	08	Yuyao	09
Cixi	10				

```
Do you want to see more city and branch codes (Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:2

Enter the name of the account holder: John Bravia
Enter the Id number of the account holder: E145326780009
Enter the address of the account holder: 1A Brandon Road, Guangzhou, China-
3210011
```

```
Enter the DoB of the account holder: 1980/05/11
Enter the initial balance of the account (not less than 1000 RMB): 800
Initial deposit can’t be less than 1000 RMB
Enter the initial balance of the account (not less than 1000 RMB): 1500
The account number is: 1011212345

Do you want to open more accounts?(Y/N): Y

Enter the name of the account holder: Hao Liu
Enter the Id number of the account holder: E175526740307
Enter the address of the account holder: 2B Hansui Road, Ningbo, China-
312101
Enter the DoB of the account holder: 1985/07/21
Enter the initial balance of the account (not less than 1000 RMB): 800
Initial deposit can’t be less than 1000 RMB.
Enter the initial balance of the account (not less than 1000 RMB): 1200
The account number is: 1231514327

Do you want to open more accounts?(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:3

Enter an account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Address of the account holder: 1A Brandon Road, Guangzhou, China- 3210011
DoB of the account holder: 1980/05/11
Balance of the account: 1500
Date of opening: 2020/11/19

Do you want to see the details of more accounts?(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:4
```

```
Enter the city code: 101
Enter the branch code: 12

Account Number          Name                Id Number                   Balance
1011212345              John Bravia         E145326780009               1500

Do you want to see more account lists(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit


Option:5

Enter the account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Balance of the account: 2000

Enter the deposit amount: 500

New account balance is: 2500

Do you want to deposit in another account(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit


Option:6

Enter the account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Balance of the account: 1500

Enter the withdrawal amount: 2500
The withdrawal amount can’t be more than the balance in the account.
```

```
Enter the withdrawal amount: 500
New account balance is: 1500

Do you want to withdraw in another account(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:7

Enter the debiting account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Balance of the account: 1500

Enter the crediting account number: 1231514327

Name of the account holder: Hao Liu
Id number of the account holder: E175526740307
Balance of the account: 1200

Enter the transfer amount: 1600

Transfer amount can’t be more than the balance of the debiting account.

Enter the transfer amount: 200

Transaction successful

Balance of the account 1011212345 is: 1300
Balance of the account 1231514327 is: 1400

Do you want to do more transfer(Y/N): Y

Enter the debiting account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Balance of the account: 1300

Enter the crediting account number: 1231514327

Name of the account holder: Hao Liu
Id number of the account holder: E175526740307
Balance of the account: 1400
```

```
Enter the transfer amount: 100

Transaction successful.

Balance of the account 1011212345 is: 1200
Balance of the account 1231514327 is: 1500

Do you want to do more transfer(Y/N): N


1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:8

Enter the account number: 1011212345
Enter the starting date: 2020/11/19
Enter the ending date: 2020/11/22

End date can't be later than the start date

Enter the ending date: 2020/11/15
```

Date	Debiting account	Crediting account	Transfer amount
2020/11/19	1011212345	1231514327	100
2020/11/19	1011212345	1231514327	200

```
Do you want to check more transaction details(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:9

Enter an account number: 1011212345

Name of the account holder: John Bravia
Id number of the account holder: E145326780009
Balance of the account: 1200
```

```
Do you want to close the account(Y/N): Y

Account successfully closed

Do you want to close more account(Y/N): N

1) Show city code and branch code
2) Open an account
3) Show account details
4) Show list of accounts
5) Deposit in an account
6) Withdraw from an account
7) Transfer money
8) Transaction details
9) Close an account
10)Quit

Option:10

Z2019024 $
```

Hints

- You have efficiently utilized files to store account-related information.
- You have to appropriately create, read from, write into, and close files to store data and process data.
- You should create as minimum files as possible. You need to close all the open files when they are no longer in use.
- Remember to free any main memory which you no longer need. Your program should not have any main memory leaks (dynamically allocated areas of memory that are no longer reachable). You will need to consider how the responsibility for allocated data transfers as your program runs.
- Your program should be able to handle some, if not all, invalid input entered by users.

END