**University of Michigan – Dearborn**

**CIS 200 – Computer Science 2**

**Lab07**

Nahrin Sharna

[nsharna@umich.edu](mailto:nsharna@umich.edu)

Date: 03/11/2019

# Source Code:

/\*

Author: Nahrin Sharna

Creation Date: 03/02/2019

Modification Date: 03/011/2019

Purpose: To store input to a random access file

\*/

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

using namespace std;

struct clientData {

int accountNumber;

char lastName[16];

char firstName[11];

float balance;

};

int main() {

string filename = "credit.dat";

ofstream creditFile(filename.c\_str(), ios::out);

clientData blankClient = { 0, "", "", 0.0 };

const int size = sizeof(clientData);

//creating file with empty records

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "creating file " << filename << " with empty records" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 100; i++) {

creditFile.write(reinterpret\_cast<const char \*>(&blankClient), size);

}

creditFile.close();

//now fill some data for few clients

clientData client;

ofstream creditFile1(filename.c\_str(), ios::ate);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "filling file " << filename << " with user input records" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

while (true) {

do {

cout << "Enter account number in the range 1 - 100(type 0 to quit): ";

cin >> client.accountNumber;

} while (client.accountNumber < 0 || client.accountNumber > 100);

if (client.accountNumber == 0)//quit?

break;

cout << "Enter first name: ";

cin >> client.firstName;

cout << "Enter last name: ";

cin >> client.lastName;

cout << "Enter account balance: ";

cin >> client.balance;

creditFile1.seekp((client.accountNumber - 1) \* size);

creditFile1.write(reinterpret\_cast<char \*>(&client), size);

cout << "updated client details for account number " << client.accountNumber << " in file" << endl << endl;

}

creditFile1.close();

//now display the records whihc user requests

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Now reading specific user accounts" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

ifstream inCredit(filename.c\_str(), ios::in);

while (true) {

do {

cout << "Enter account number in the range 1 - 100(type 0 to quit): ";

cin >> client.accountNumber;

} while (client.accountNumber < 0 || client.accountNumber > 100);

if (client.accountNumber == 0)//quit?

break;

inCredit.seekg((client.accountNumber - 1) \* size);

inCredit.read(reinterpret\_cast<char \*>(&client), size);

cout << "Account number: " << client.accountNumber << endl;

cout << "Name: " << client.lastName << ", " << client.firstName << endl;

cout << "Balance: " << client.balance << endl << endl;

}

//now display all valid accounts

cout << "Displaying records" << endl;

inCredit.seekg(0);

cout << setw(10) << "A/C Num";

cout << setw(20) << "First Name";

cout << setw(20) << "Last Name";

cout << fixed << setprecision(2) << setw(10) << "Balance" << endl;

inCredit.read(reinterpret\_cast<char\*>(&client), size);

while (inCredit && !inCredit.eof()) {

if (client.accountNumber != 0) {

cout << setw(10) << client.accountNumber;

cout << setw(20) << client.firstName;

cout << setw(20) << client.lastName;

cout << fixed << setprecision(2) << setw(10) << client.balance << endl;

}

inCredit.read(reinterpret\_cast<char\*>(&client), size);

}

inCredit.close();

//update

ofstream creditFile2(filename.c\_str(), ios::ate);

while (true) {

do {

cout << "Enter your account number to update your account in the range 1 -100(type 0 to quit): ";

cin >> client.accountNumber;

} while (client.accountNumber < 0 || client.accountNumber > 100);

if (client.accountNumber == 0)//quit??

break;

cout << "Enter first name: ";

cin >> client.firstName;

cout << "Enter last name: ";

cin >> client.lastName;

cout << "Enter account balance: ";

cin >> client.balance;

creditFile2.seekp(size\*client.accountNumber, ios::beg);

creditFile2.write(reinterpret\_cast<char \*>(&client), size);

cout << "Account number: " << client.accountNumber << endl;

cout << "Name: " << client.lastName << ", " << client.firstName << endl;

cout << "Balance: " << client.balance << endl << endl;

}

creditFile2.close();

system("pause");

return 0;

}

# Initial Testing:

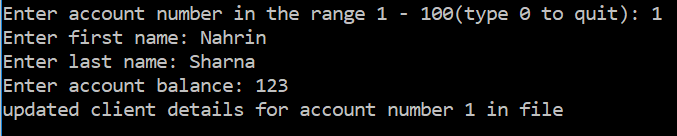
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | Storing values to an account number (1-100) | Account: 1  First name: Nahrin  Second name: Sharna  Balance: 123 | Updated client details for the account number 1 in the file |  |  |
| 2 | Valid | When user enter account = 0 while storing data in the file | Account: 0 | Option to store datas quits.  Reading option is given |  |  |
| 3 | Valid | Read the data field value of a valid account number | Account: 1 | Account number: 1  Name: Sharna, Nahrin  Balance: 123 |  |  |
| 4 | Valid | When user enter account = 0 while reading data in the file | Account: 0 | Option to read datas quits.  All the stored datas are displayed |  |  |
| 5 | Valid | Updating account 1 | Account:1  First: Nahrin  Last: Zarin  Balance: 234 | Account number: 1  Name: Zarin, Nahrin  Balance: 234 |  |  |
| 6 | Valid | When user enter account = 0 for updating | Account = 0 | The program quits |  |  |

# Final Testing:

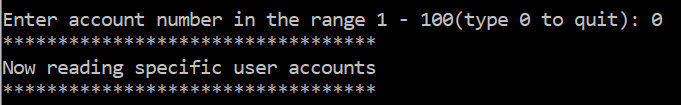
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | Storing values to an account number (1-100) | Account: 1  First name: Nahrin  Second name: Sharna  Balance: 123 | Updated client details for the account number 1 in the file | Updated client details for the account number 1 in the file | Pass |
| 2 | Valid | When user enter account = 0 while storing data in the file | Account: 0 | Option to store datas quits.  Reading option is given | Option to store datas quits.  Reading option is given | Pass |
| 3 | Valid | Read the data field value of a valid account number | Account: 1 | Account number: 1  Name: Sharna, Nahrin  Balance: 123 | Account number: 1  Name: Sharna, Nahrin  Balance: 123 | Pass |
| 4 | Valid | When user enter account = 0 while reading data in the file | Account: 0 | Option to read datas quits.  All the stored datas are displayed | Option to read datas quits.  All the stored datas are displayed | Pass |
| 5 | Valid | Updating account 1 | Account:1  First: Nahrin  Last: Zarin  Balance: 234 | Account number: 1  Name: Zarin, Nahrin  Balance: 234 | Account number: 1  Name: Zarin, Nahrin  Balance: 234 | Pass |
| 6 | Valid | When user enter account = 0 for updating | Account = 0 | The program quits | The program quits | Pass |

# Screenshots:

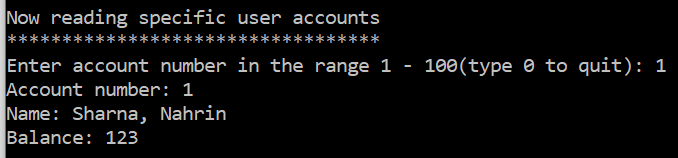
## Test Case 1:



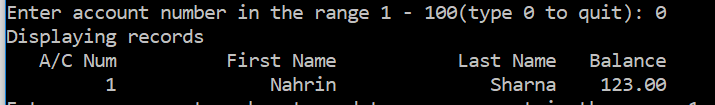
## Test Case 2:



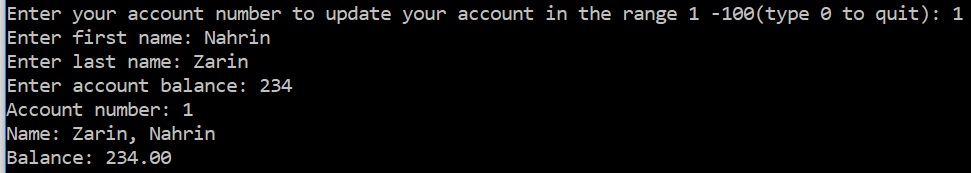
## Test Case 3:



## Test Case 4:



## Test Case 5:



## Test Case 6:



## Displaying All Stored Records(Total 16):

