ASSIGNMENT COVER SHEET

A picture containing chart

Description automatically generated

STUDENT NAME: Stefanos Eleftheriadis

STUDENT NUMBER: 24839

SUBJECT CODE: EEI\_133

TITLE OF ASSIGNMENT: Credit Files

DUE DATE:30th of March 2022 DATE SUBMITTED: 9th of March 2022

LECTURER’S NAME: Christos Makarounas

I declare that:

a) this assignment is entirely my own work, except where I have included fully-documented references to the work of others;

b) the material contained in this assignment has not previously been submitted for any other subject at the University

Περιεχόμενο Σελίδες

Πρόγραμμα 3-15

-Interface 3

-Main 4-5

-Functions 6-16

Αποτελέσματα 17-21

-Input 17-19

-Binary File 19

-Δοκιμή1, Δοκιμή2 20

- Δοκιμή3, Text File 22

Πρόγραμμα:

***Interface:*** (interface.h)

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <cstring>

#include <windows.h>

#include <cmath>

#include <bits/stdc++.h>

#define ArraySize 50

using namespace std;

class files{

public:

files(FILE \*Cfile\_ptr);

bool EmptyFileYN();

void create\_bfile();

void customer();

void admi\_menu();

void create\_tfile();

int name\_size(int Sizes[ArraySize]);

void create\_record();

void update\_record();

void delete\_record();

void transaction();

private:

FILE \*file\_ptr;

};

struct clientData

{

int account\_number;

char customer\_name[100];

float current\_balance;

float credit\_limit;

};

***Main:*** (main.cpp)

#include "interface.h"

int main()

{

FILE \*file\_ptr;

int choice;

bool empty\_file=false;

files creator(file\_ptr);

do{

printf("Enter:\n\t1.Administrator\n\t2.Customer\n\t3.Exit\nChoice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:

creator.admi\_menu();

system("pause");

break;

case 2:

if((file\_ptr=fopen("credit.dat","r+"))!=NULL)

{

files client(file\_ptr);

empty\_file=client.EmptyFileYN();

if(!empty\_file)

{

client.customer();

}

else

{

printf("The binary file is empty!\n");

}

fclose(file\_ptr);

}

else

{

printf("The binary file doesn't exist!\n");

}

system("pause");

break;

case 3:

if((file\_ptr=fopen("credit.dat","r"))!=NULL)

{

files readFile(file\_ptr);

empty\_file=readFile.EmptyFileYN();

fclose(file\_ptr);

}

else

{

printf("The binary file doesn't exist!\n");

}

printf("Thanks for using our services!!!\n");

system("pause");

break;

default:

printf("Wrong Input!\n");

system("pause");

break;

}

system("CLS");

}while(choice!=3);

if(!empty\_file)

{

creator.create\_tfile();

printf("Text file was created!\n");

system("pause");

}

else

{

printf("There are no customers to be registerd in the text file!\n");

}

return 0;

}

***Functions:*** (functions.cpp)

#include "interface.h"

//Constructor

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

files::files(FILE \*Cfile\_ptr)

{

file\_ptr=Cfile\_ptr;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Administrator Menu

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void files::admi\_menu()

{

struct clientData Client={0,"",0.0,0.0};

bool empty\_file=false;

int choice;

system("CLS");

do{

printf("1.Create Account\n2.Delete Account\n3.Update Account\n4.Exit\nChoice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:

files::create\_bfile();

system("pause");

break;

case 2:

if((file\_ptr=fopen("credit.dat","r+"))!=NULL)

{

empty\_file=files::EmptyFileYN();

if(!empty\_file)

{

files::delete\_record();

}

else

{

printf("The binary file is empty!\n");

}

fclose(file\_ptr);

}

system("pause");

break;

case 3:

if((file\_ptr=fopen("credit.dat","r+"))!=NULL)

{

empty\_file=files::EmptyFileYN();

if(!empty\_file)

{

files::update\_record();

}

else

{

printf("The binary file is empty!\n");

}

fclose(file\_ptr);

}

system("pause");

break;

case 4:

cout<<"Exit from administrator!"<<endl;

break;

default:

printf("Wrong Input!\n");

system("pause");

break;

}

system("CLS");

}while(choice!=4);

}

//Update Account

void files::update\_record()

{

struct clientData Client={0,"",0.0,0.0};

fread(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

fflush(stdin);

printf("Enter Name: ");

cin.getline (Client.customer\_name,100);

fflush(stdin);

printf("Enter Credit Limit: ");

scanf("%f",&Client.credit\_limit);

fwrite(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

}

//Delete Account

void files::delete\_record()

{

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

fwrite(&blankClient,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(blankClient.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

printf("Successfully deleted!\n");

}

//Create Binary File

void files::create\_bfile()

{

struct clientData Clientlist={0,"",0.0,0.0};

int counter;

char exit;

if((file\_ptr=fopen("credit.dat","w"))!=NULL)

{

for(counter=1;counter<=200;counter++)

{

fwrite(&Clientlist,sizeof(struct clientData),1,file\_ptr);

}

printf("Now write the records!\n");

counter=1;

do{

fflush(stdin);

printf("Enter Name[%d]:",counter);

cin.getline (Clientlist.customer\_name,100);

do{

fflush(stdin);

printf("Enter account number[%d]: ",counter);

scanf("%d",&Clientlist.account\_number);

if(Clientlist.account\_number%50!=0 || Clientlist.account\_number>10000 || Clientlist.account\_number<50)

{

printf("Wrong Input!\n");

}

}while(Clientlist.account\_number%50!=0 || Clientlist.account\_number>10000 || Clientlist.account\_number<50);

Clientlist.current\_balance=0;

fflush(stdin);

printf("Enter Credit Limit[%d]: ",counter);

scanf("%f",&Clientlist.credit\_limit);

fseek(file\_ptr,(Clientlist.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

fwrite(&Clientlist,sizeof(struct clientData),1,file\_ptr);

fflush(stdin);

printf("If you want to stop entering clients, enter Y: ");

scanf("%c",&exit);

counter++;

}while(exit!='Y');

fclose(file\_ptr);

}

else

{

printf("The file couldn't be created!\n");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Check if the binary file is empty

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

bool files::EmptyFileYN()

{

struct clientData Client={0,"",0.0,0.0};

while(!feof(file\_ptr))

{

fread(&Client,sizeof(struct clientData),1,file\_ptr);

if(Client.account\_number!=0)

{

return false;

}

}

return true;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Customer Input

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void files::customer()

{

struct clientData Client={0,"",0.0,0.0};

int accountNum,value;

char new\_accountYN,exit;

char customerName[20];

float creditLimit;

do{

system("CLS");

do{

printf("Enter account number(50-20000): ");

scanf("%d",&accountNum);

if(accountNum%50!=0 || accountNum>10000 || accountNum<50)

{

printf("Wrong Input!\n");

}

}while(accountNum%50!=0 || accountNum>10000 || accountNum<50);

fseek(file\_ptr,(accountNum/50-1)\*sizeof(struct clientData),SEEK\_SET);

fread(&Client,sizeof(struct clientData),1,file\_ptr);

if(Client.account\_number!=0)

{

fseek(file\_ptr,(accountNum/50-1)\*sizeof(struct clientData),SEEK\_SET);

system("CLS");

files::transaction();

}

else

{

do{

fflush(stdin);

printf("Is this a new account number?(Y or N): ");

scanf("%c",&new\_accountYN);

if(new\_accountYN!='Y' && new\_accountYN!='N')

{

printf("Wrong Input!\n");

}

}while(new\_accountYN!='Y' && new\_accountYN!='N');

if(new\_accountYN=='N')

{

printf("Invalid account number-re-enter data\n");

}

else

{

system("CLS");

fseek(file\_ptr,(accountNum/50-1)\*sizeof(struct clientData),SEEK\_SET);

fread(&Client,sizeof(struct clientData),1,file\_ptr);

Client.account\_number=accountNum;

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

fwrite(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

files::create\_record();

}

}

fflush(stdin);

printf("If you want to exit from the customr checking enter Y: ");

scanf("%s",&exit);

}while(exit!='Y');

}

//Transaction

void files::transaction()

{

SetConsoleOutputCP(1253);

struct clientData Client={0,"",0.0,0.0};

float money=0.0;

bool wrong\_money=false;

fread(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

do

{

wrong\_money=false;

printf("Enter account number(50-20000): %d\n",Client.account\_number);

printf("Enter amount of purchase: ");

scanf("%f",&money);

if(Client.current\_balance+money<=Client.credit\_limit)

{

Client.current\_balance+=money;

printf("Dear %s thank you for the business!\n",Client.customer\_name);

system("pause");

}

else

{

wrong\_money=true;

printf("Dear %s we cannot authorise this purchase at this time as it would put your current balance %c%.2f above your credit limit.\n",Client.customer\_name,128,fabs(Client.credit\_limit-money-Client.current\_balance));

system("pause");

}

system("CLS");

}while(wrong\_money);

fwrite(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

}

//New\_record

void files::create\_record()

{

struct clientData Client={0,"",0.0,0.0};

char temp\_char;

fread(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

fflush(stdin);

printf("Enter Name: ");

cin.getline (Client.customer\_name,100);

Client.current\_balance=0;

fflush(stdin);

printf("Enter Credit Limit: ");

scanf("%f",&Client.credit\_limit);

fwrite(&Client,sizeof(struct clientData),1,file\_ptr);

fseek(file\_ptr,(Client.account\_number/50-1)\*sizeof(struct clientData),SEEK\_SET);

printf("Successfully created!\n");

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Find the size of names

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int files::name\_size(int Sizes[ArraySize])

{

struct clientData Client={0,"",0.0,0.0};

int counter,char\_count,max\_size, NSize\_customer\_count=0;

char name[20];

//Calculate the size of every name

while(!feof(file\_ptr))

{

char\_count=0;

fread(&Client,sizeof(struct clientData),1,file\_ptr);

strcpy(name,Client.customer\_name);

while(name[char\_count]!='\0')

{

Sizes[NSize\_customer\_count]++;

char\_count++;

}

if(Client.account\_number!=0)

{

Sizes[NSize\_customer\_count]++;

NSize\_customer\_count++;

}

}

//Find the maximum size of address

max\_size=Sizes[0];

for(counter=1;counter<=NSize\_customer\_count;counter++)

{

if(max\_size<Sizes[counter])

{

max\_size=Sizes[counter];

}

}

return max\_size;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Text File

void files::create\_tfile()

{

FILE \*textfile\_ptr;

struct clientData Client={0,"",0.0,0.0};

int Names\_size[ArraySize]={0},Max\_Size=0,icount,counter=0;

if((file\_ptr=fopen("credit.dat","r"))!=NULL)

{

if((textfile\_ptr=fopen("creditTextFile.txt","w"))!=NULL)

{

Max\_Size=name\_size(Names\_size);

rewind(file\_ptr);

fprintf(textfile\_ptr,"%s","Customer Name");

if(Max\_Size-13>0)

{

for(icount=0;icount<Max\_Size-8;icount++)

{

fprintf(textfile\_ptr," ");

}

}

else

{

Max\_Size=13;

fprintf(textfile\_ptr," ");

}

fprintf(textfile\_ptr,"%s\t\t%s\t\t%s\n","Account Number","Current Balance","Credit Limit");

while(!feof(file\_ptr))

{

fread(&Client,sizeof(struct clientData),1,file\_ptr);

if(Client.account\_number!=0)

{

fprintf(textfile\_ptr,"%s",Client.customer\_name);

for(icount=0;icount<Max\_Size-Names\_size[counter]+5;icount++)

{

fprintf(textfile\_ptr," ");

}

fprintf(textfile\_ptr,"%14d\t\t%15.2f\t\t%12.2f\n",Client.account\_number,Client.current\_balance,Client.credit\_limit);

counter++;

}

}

fclose(textfile\_ptr);

}

else

{

printf("The file couldn't be created!\n");

}

fclose(file\_ptr);

}

else

{

printf("The binary file doesn't exist!\n");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Αποτελέσματα:

***Input:***

Enter:

1.Administrator

2.Customer

3.Exit

Choice: 1

1.Create Account

2.Delete Account

3.Update Account

4.Exit

Choice: 1

Now write the records!

Enter Name[1]:TURBO FERRARI

Enter account number[1]: 450

Enter Credit Limit[1]: 900

If you want to stop entering clients, enter Y: N

Enter Name[2]:MARY JANE SMITH

Enter account number[2]: 400

Enter Credit Limit[2]: 500

If you want to stop entering clients, enter Y: N

Enter Name[3]:JIM JAY JONES

Enter account number[3]: 500

Enter Credit Limit[3]: 750

If you want to stop entering clients, enter Y: N

Enter Name[4]:ELIZABETH CROWLEY

Enter account number[4]: 100

Enter Credit Limit[4]: 1000

If you want to stop entering clients, enter Y: N

Enter Name[5]:FELICITY BOWSER

Enter account number[5]: 50

Enter Credit Limit[5]: 300

If you want to stop entering clients, enter Y: N

Enter Name[6]:SAM PENNYWHISTLE

Enter account number[6]: 250

Enter Credit Limit[6]: 250

If you want to stop entering clients, enter Y: N

Enter Name[7]:ALICE JOHNSON

Enter account number[7]: 350

Enter Credit Limit[7]: 500

If you want to stop entering clients, enter Y: N

Enter Name[8]:FRANK GRICHLEY

Enter account number[8]: 550

Enter Credit Limit[8]: 300

If you want to stop entering clients, enter Y: N

Enter Name[9]:DAISY DANDELION

Enter account number[9]: 150

Enter Credit Limit[9]: 500

If you want to stop entering clients, enter Y: N

Enter Name[10]:BILL COSBY

Enter account number[10]: 200

Enter Credit Limit[10]: 500

If you want to stop entering clients, enter Y: K

Enter Name[11]:LOUIS LACKLUSTER

Enter account number[11]: 650

Enter Credit Limit[11]: 100

If you want to stop entering clients, enter Y: N

Enter Name[12]:JANE SEYMOUR

Enter account number[12]: 600

Enter Credit Limit[12]: 200

If you want to stop entering clients, enter Y: Y

***Binary File:***

Graphical user interface, application

Description automatically generated

***Δοκιμή 1:***

Enter account number(50-20000): 700

Is this a new account number?(Y or N): N

Invalid account number-re-enter data

Enter account number(50-20000): 750

Is this a new account number?(Y or N): Y

Enter Name: CHRIS BIGBOTTON

Enter Credit Limit: 200

Successfully created!

Enter account number(50-20000): 750

Enter amount of purchase: 500

Dear CHRIS BIGBOTTON we cannot authorise this purchase at this time as it would put your current balance €300.00 above your credit limit.

Enter amount of purchase: 100

Dear CHRIS BIGBOTTON thank you for the business!

***Δοκιμή 2:***

Enter account number(50-20000): 50

Enter amount of purchase: 65.99

Dear FELICITY BOWSER thank you for the business!

***Δοκιμή 3:***

Enter account number(50-20000): 880

Wrong Input!

Enter account number(50-20000): 850

Is this a new account number?(Y or N): N

Invalid account number-re-enter data

If you want to exit from the customr checking enter Y: l

Enter account number(50-20000): 750

Enter amount of purchase: 90

Dear CHRIS BIGBOTTON thank you for the business!

***Text File:***

Graphical user interface, application

Description automatically generated