Țogoe Radu-Mihai Vladu Marian Dumitru

Bank Management System

<documentație>

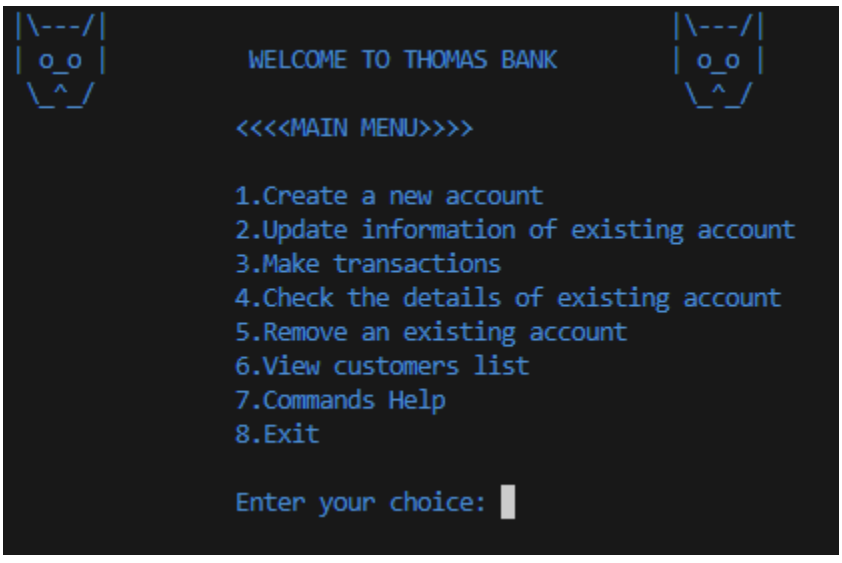
1. Descrierea proiectului și a opțiunilor din meniu

Proiectul propus reprezintă un sistem de evidență bancară cu ajutorul căruia putem crea, șterge și edita conturi bancare. Acesta a fost realizat în C++, folosind conceptul de programare orientată obiect. Operațiile implementate pot fi accesate din menin sau direct din comenzi.

Opțiunile meniului:

Pentru salvarea progresului și funcționarea corectă a programului, trebuie folosită opțiunea 8.Exit

Pentru a selecta o opțiune din meniu, se scrie numărul corespunzător opțiunii dorite [1-8] și se apasă ENTER



1. Create a new account

Crearea unui cont nou – utilizatorul trebuie să introducă pe rând datele cerute (first name/prenume, last name/prenume, phone number/numar de telefon, adress/adresă). După crearea contului, se va afișa un ID specific acestuia, ce trebuie memorat de utilizator pentru a modifica contul în viitor.

În final, utilizatorul are opțiunea de a crea un cont nou sau de a se întoarce în meniu

2.Update information of existing account

Actualizarea informatiilor unui cont existent – utilizatorul trebuie să întroducă ID-ul specific contului pe care vrea să îl actualizeze.

Dacă ID-ul este corect, utilizatorul trebuie să selecteze datele ce trebuie modificate (prenume/nume/adresă/număr telefon), iar apoi să introducă valoarea nouă care va fi atribuită. După ce modificarea este finalizată, utilizatorul va avea opțiunea de a modifica alt cont sau de a se întoarce în meniu.

Dacă ID-ul este incorect (nu corespunde niciunui cont), programul va specifica acest lucru, oferind utilizatorului opțiunea de a încerca din nou sau de a se întoarce în meniu.

3.Make transactions

Efectuarea de tranzacții - utilizatorul trebuie să întroducă ID-ul specific contului cu care vrea să facă tranzacții

Dacă ID-ul este corect, utilizatorul trebuie sa aleagă din a adăuga bani în cont (deposit) sau de a retrage bani (withdraw), după care acesta va trebui să introducă suma dorită. Daca din întamplare utilizatorul dorește să retragă o sumă mai mare decât balanța contului, programul va specifica acest lucru, oferindu-i opțiunea de a încerca din nou sau de a se întoarce în meniu.

Dacă ID-ul este incorect (nu corespunde niciunui cont), programul va specifica acest lucru, oferind utilizatorului opțiunea de a încerca din nou sau de a se întoarce în meniu.

4.Check the details of existing account

Verificarea detaliilor unui cont existent – utlizatorul trebuie să introducă ID-ul specific contului ce va fi afișat

Dacă ID-ul este corect, se va afișa contul, după care utilizatorul va avea opțiunea de a verifica alt cont sau de a se întoare în meniu

Dacă ID-ul este incorect (nu corespunde niciunui cont), programul va specifica acest lucru, oferind utilizatorului opțiunea de a încerca din nou sau de a se întoarce în meniu.

5.Remove an existing account

Ștergerea unui cont existent – utilizatorul trebuie să introducă ID-ul specific contului pe care dorește să îl șteargă

Dacă ID-ul este corect, contul va fi șters, după care utilizatorul va avea opțiunea de a șterge alt cont sau de a se întoarce în meniu

Dacă ID-ul este incorect (nu corespunde niciunui cont), programul va specifica acest lucru, oferind utilizatorului opțiunea de a încerca din nou sau de a se întoarce în meniu.

6. View customers list

Afișează lista conturilor create, fiind disponibilă opțiunea de întoarcere la meniul principal

7. Commands Help

Se afișează un ghid pentru utilizarea programului direct din comenzi, fiind disponibilă opțiunea de întoarcere la meniul principal

8. Exit

Se salvează modificările și se iese din program

2. Compilarea și execuția programului

Doar pentru a utiliza meniul manual:

Compilare:

g++ meniu.cpp

Execuție:

./a.exe

Pentru a utiliza meniul fie manual, fie pentru a efectua operațiuni direct din comenzi:

g++ meniu.cpp -o meniu

Executare pentru a utiliza meniul manual:

./meniu

Executare pentru a efectual operații direct din linia de comandă:

1. Crearea unui cont nou:

./meniu 1 “prenume” “nume” “adresa” numar_telefon

După execuția comenzii, se va afișa ID-ului noului cont creat

2. Actualizarea informațiilor unui cont existent:

./meniu 2 id_cont nr “Valoare_noua” unde nr = {1, 2, 3, 4} <=> {1 = prenume, 2 = nume, 3 =număr telefon, 4 = adresă}

3.Efectuare tranzacții:

./meniu 3 id_cont 1=depozitare/2=retragere suma_dorita

4.Verificarea detaliilor unui cont existent:

./meniu 4 id_cont

5.Ștergerea unui cont existent:

./meniu 5 id_cont

6.Afișarea tuturor conturilor:

./meniu 6

7.Afișarea ghidului comenzilor pentru efectuarea operațiunilor direct din linia de comandă:

./meniu 7

3. Explicarea programului / Documentarea codului sursă

Codul este realizat special pentru a fi executat folosind sistemul de operare Windows

Fișiere:

meniu.cpp - fișierul principal, conține implementarea meniului și a funcționalităților acestuia

data_management.h - fișier ce conține definirea structurilor de date, preluarea și actualizarea datelor în fișier

data.csv - fișier ce conține conturile bancare

Structura fișierului **data.csv** este următoarea:

prenume,nume,adresa,numar telefon,ID,balanță

Librării folosite:

#include <iostream> - pentru funcțiilor de citire din consolă

#include <fstream> - pentru stream-urile și funcțiile de citire din fișuer

#include <windows.h> - pentru editarea afișării în consolă (text colorat/golire ecran)

#include <time.h> - pentru randomizarea ID-ului

#include <vector> - pentru implementarea vectorilor din STL

#include <string.h> - pentru utilizarea funcțiilor pentru șiruri de caractere

#include <string: - pentru implementarea string-urilor din STL (rar folosite)

Clasele, atributele și metodele acestora:

Clasa ContBancar:

Reprezintă conturile bancare, acestea având următoarele atribute:

- nume, prenume, adresă, număr telefon – șiruri de caractere ce vor fi alocate în memorie (char*)
- id, balanță - variabile de tip întreg (int)

Constructori:

ContBancar(char *, char *, char *, char *, int, int) - constructor ce preia argumente și creează un ContBancar nou atribuind valorile argumentelor atributelor obiectului (atributele ce reprezinta siruri de caracter sunt alocate în memorie cu ajutorul funcției **new**)

ContBancar(const ContBancar &) - constuctor de copiere, ce copiază datele din obiectul dat ca argument în obiectul nou creat

ContBancar() { balanta = 0; }; - constructor default, ce setează balanța 0

Alte metode:

Pentru respectarea conceptului de encapsulare, am folosit setteri si getteri pentru a edita atributele:

setteri: setNume(), setPrenume(), setAdresa(), setTelefon(), setBalanta()

getteri: getID(), getBalanta()

creareLinie() - se preiau datele din obiect și se crează o un șir de caractere formatat astfel încât să respecte formatul fișierului CSV

Clasa Banca:

Clasă folosită pentru a implementa relația de asociere, aceasta fiind asociată cu clasa ContBancar prin vectorul: `vector <ContBancar*> vectorul;`

Pentru a modifica vectorul în timpul utilizării programului, am folosit getteri **getVector()** și setteri **setVector().**

Citirea din fișier:

-am golit vectorul de conturi bancare pentru a evita anumite erori/bug uri

-am deschis fișierul prin **ifstream**

-am citit fiecare linie din fișier prin intermediul funcției getline() și am separat datele de virgule prin intermediul funcției strtok()

-după separare, am creat un ContBancar nou cu ajutorul constructorului și l-am adăugat în vector

Afișarea în fișier:

-se deschide fișierul prin **ofstream**

-se parcurge vectorul de conturi bancare

-se afișează contul în fișier (acesta va repreznta o linie formatată prin intermediul metodei createLinie()

Alte funcții folosite pentru formatarea datelor:

char *nrToStr(int nr) - transformă un număr într-un șir de caractere

int strToNr(char *s) - transformă un șir de caractere într-un număr

int oglindit(int n) - returnează oglinditul numărului n

Implementarea funcționalităților din meniu:

Pentru lucrarea cu argumentele din comenzi:

-am folosit un pointer global care reține adresa listei de argumente (char** argumente) și numărul de argumente (int nr_argumente)

- dacă numărul de argumente este egal cu 1 (singurul argument este numele executabilului - “meniu”), vom executa programul normal

-dacă acesta este diferit de 1, vom verifica valoarea al doilea argument (specific opțiunii selectate), iar în opțiunea apelată implementarea va depinde de numărul de argumente (dacă este 1, atunci am apelat-o manual din meniu, altfel, am apelat-o din argument, deci vom modifica implementarea)

Funcționalitățile propriu-zise: (descrise și la începutul documentului)

-menu() - funcția ce afișează meniul și în care se execută opțiunile alese

-create() - se preiau date de la utilizator și se crează un cont nou, după care se afișeaza id ul acestuia și se oferă opțiunea de a crea un alt cont (se reapelează funcția create) sau de întoarcere în meniu (se apelează funcția menu)

-update_account() - se preiau date de la utilizator și se actualizeză datele contului specificat, contul fiind găsit prin intermediul id-ului

-transactions() - se caută contul prin intermediul ID-ului și se fac tranzacții

-view_customer() - se caută contul după ID și se afișează datele acestuia

-stergere_client() - se caută contul după ID și se șterge

-view_customers_list() - se afișează datele tuturor utilizatorilor

-commands_help() - se afișează un ghid pentru executarea operațiunilor din linia de comandă

Pentru afișarea textului colorat:

HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE) - se face referință la consolă

SetConsoleTextAttribute(hConsole, FOREGROUND_BLUE) - se modifică culoarea consolei

Pentru golirea ecranului: system("cls")