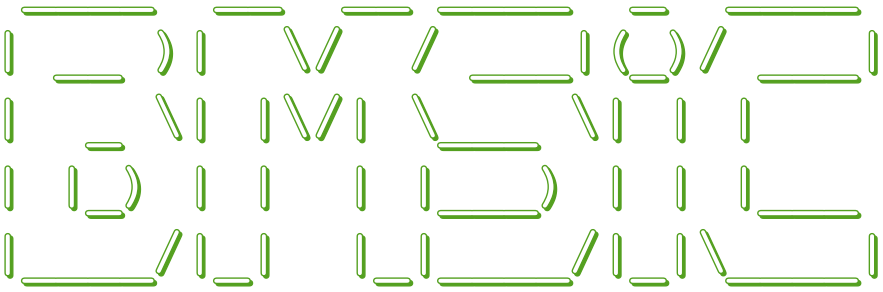


Bank Management System in C

DOKUMENTACJA PROJEKTU

AUTOR

Sebastian Fudalej

KIERUNEK STUDIÓW

Informatyka

GRUPA LABORATORYJNA

3

DATA

25.01.2020r.

Spis treści

1. Wstęp 3
1.1 Podstawowe Informacje 3
1.2 Założenia 3
1. Opis środowiska programistycznego oraz zastosowanych bibliotek 3
2. Opis funkcjonalności 4
3.1 Sterowanie i CUI 4
3.2 Założenia 4
3.2.1 Zakładanie konta w banku 4
3.2.2 Logowanie 4
3.2.3 Konto pracownika banku 5
3.2.4 Wykonanie przelewu 5
3.2.5 Wpłata/wypłata 5
3.2.6 Naliczanie odsetek 5
3.2.7 Baza danych w pliku 5
3.2.8 Wiele instancji 6
3. Grafiki koncepcyjne dla wybranych funkcjonalności 7
4. Podsumowanie 8
5.1 Problemy i przemyślenia 8
6. Raport z przebiegu realizacji projektu 9

1. Wstęp

1.1 Podstawowe Informacje

BMSiC – Bank Management System in C (z ang. “System Zarządzania Bankiem w C”) jest symulacją systemu bankowego. Podczas tworzenia konta zostaje wygenerowany unikalny numer konta na który można będzie wykonywać przelewy. Użytkownik może później zalogować się i wykonać podstawowe transakcje.

Projekt na GitHub jest dostępny pod adresem:

https://github.com/sebafudi/BMSiC

1.2 Założenia

- **Zakładanie konta w banku**
- **Generowanie numeru konta**
- **Logowanie się użytkowników**
- Podział na klientów i pracowników banku*
- **Wykonywanie przelewów pomiędzy kontami**
- **Wpłata/wypłata pieniędzy**
- **CUI pozwalające na łatwą obsługę aplikacji**
- Naliczanie odsetek na kontach w określonym interwale czasowym

Pogrubione – założenia, które udało się zrealizować

* - częściowo zrealizowane, konta mają odpowiednią zmienną (accout_type), która jest zapisywana w pliku, lecz nie jest aktualnie używana

1. Opis środowiska programistycznego oraz zastosowanych bibliotek

Środowisko programistyczne to Visual Studio 2019. Program ma docelowo działać w systemie operacyjnym Microsoft Windows 10 64 bit. Zostanie użyta kontrola wersji git na platformie GitHub. Wersja kompilatora to MSVC++14.28.

Biblioteki:

- stdio.h – podstawowe wyświetlanie informacji jak i wczytywanie danych od użytkownika, zapisywanie i odczytywanie plików
- PDCursesMod (v4.2.0) – aktywnie wspierana wersja biblioteki PDCurses (Public Domain Curses). Generowanie i wyświetlanie CUI dla użytkownika, wczytywanie wciśniętych klawiszy przez użytkownika i odpowiednie poruszanie się po CUI
- assert.h – funkcja assert() – używana podczas alokowania dynamicznej pamięci, aby sprawdzić, czy pamięć została poprawnie zaalokowana. W przeciwnym wypadku program kończy działanie
- locale.h – obsługa polskich znaków
- math.h – niektóre obliczenia takie jak wartość bezwzględna
- stdlib.h – wysyłanie komend do systemu (PAUSE), funkcje konwertowania tablicy znaków do typów liczbowych, losowe liczby, niektóre typy (div_t czy size_t)
- string.h – funkcje do obsługi tablicy znaków tj. strlen(), strstr()
- time.h – uzyskiwanie aktualnego czasu

2. Opis funkcjonalności

3.1 Sterowanie i CUI

Sterowanie w menu wyboru możliwe jest za pomocą strzałek, wyboru klawiszem ENTER jak i cofnięcia się klawiszem ESC. W menu wprowadzania tekstu użytkownik może poruszać się klawiszami strzałek (góra i dół), klawiszami TAB lub ENTER, aby przejść do następnej linijki, SHIFT+TAB – przechodzenie do poprzedniej linijki. Jeżeli podświetlona jest opcja CONFIRM, klawisz ENTER zatwierdza menu. Klawisz ESC jest dostępny w każdym momencie i zawsze cofa menu o 1, lub w menu głównym, wyłącza program.

Funkcja DisplayVerticalMenu() wyświetla odpowiednio podane w argumentach opcje i odpowiednio je środkuje wertykalnie. Funkcja DisplayHorizontalMenu() liczy długość wszystkich znaków we wszystkich opcjach, wynik odejmowania szerokości okna i wcześniejszego wyniku dzieli na ilość elementów + 1, aby automatycznie równomiernie rozstawić elementy w całej konsoli.

3.2 Założenia

3.2.1 Zakładanie konta w banku

Użytkownik podaje informacje o sobie. Ustala hasło do konta, do którego generowany jest indywidualny numer.

Funkcja CreateUser() tworzy użytkownika z danych wprowadzonych przez użytkownika w menu w funkcji TextInputMenu(), po czym przekazuje dane o użytkowniku do funkcji SaveUser(), która zapisuje dane do pliku. Podczas całej operacji sprawdzane jest to, czy nazwa użytkownika nie powtarza się, czy wszystkie pola są uzupełnione, a także czy wygenerowany losowo numer konta nie powtarza się. Dodatkowo hasło zostaje zaszyfrowane algorytmem djb2, przez co nie da się odczytać już wprowadzonych haseł w pliku. Dodatkowym zabezpieczeniem jest minimalna długość hasła – 8 znaków.

3.2.2 Logowanie

Użytkownik podaje utworzoną wcześniej nazwę użytkownika jak i hasło. Sprawdzane one są z danymi w pliku. Jeżeli się zgadzają wyświetla się stan konta i zapytanie co użytkownik chce robić dalej.

Wykonywana jest funkcja TextInputMenu() w której użytkownik musi wpisać swój login i hasło. Po kliknięciu przycisku CONFIRM, następuje wyszukiwanie użytkownika w pliku funkcją FindByLogin(), która korzysta z funkcji FindLineContainingText() znajdującej odpowiednią linię w pliku z bazą danych. Funkcja dodatkowo sprawdza czy znaleziony tekst znajduje się w odpowiednim polu (np. login czy numer konta). Jeżeli wszystko się powiedzie, wyświetla się menu z opcjami sprawdzania stanu konta, przelewu, wpłacanie lub wypłacenia pieniędzy. Jeżeli coś pójdzie nie tak (wprowadzone hasło będzie niepoprawne,

konto nie będzie istnieć) – wyświetli się komunikat o złym haśle. Uniemożliwia to potencjalnemu przestępcy sprawdzenia czy dane konto istnieje.

3.2.3 Konto pracownika banku

Pracownik banku zobaczyć listę użytkowników. Nie są wyświetlane stany kont. Ma możliwość zmiany hasła każdemu klientowi.

Założenie niezrealizowane.

3.2.4 Wykonanie przelewu

Użytkownik wpisuje numer konta i kwotę. Program sprawdza czy użytkownik ma tyle pieniędzy na koncie, a później odejmuje odpowiednią kwotę z konta użytkownika jak i dodaje ją do konta odbiorcy.

Funkcja DisplayTransferMoney() wywołuje funkcję TextInputMenu(), w którym należy wprowadzić login lub numer konta, na który chcemy wykonać przelew. Sprawdzane jest czy dane konto istnieje funkcją FindByLogin() dla wyszukiwania po loginie lub FindByAccNo() dla wyszukiwania po numerze konta. Następna wykonywana funkcja to FloatInputMenu(). Funkcja ta wyświetla menu, w którym możliwe jest wprowadzanie tylko cyfr, kropki i dokładnie dwóch cyfr po kropce. Liczba potem mnożona jest przez 100 (0.01 staje się 1, a 1 staje się 100) i przechowywana jako long long int zamiast np. float czy double, aby uniknąć niedokładności związanych z liczbami zmiennoprzecinkowymi. Później sprawdzane jest czy użytkownik ma odpowiednią kwotę na koncie i wykonywany jest przelew (kwota jest odejmowana z konta robiącego przelew i dodawana do konta adresata). Dodatkowo, jeżeli użytkownik wprowadzi login i numer konta, sprawdzana jest poprawność obu identyfikatorów i jeżeli są niezgodne, wyświetla się odpowiednie powiadomienie.

3.2.5 Wpłata/wypłata

W obu przypadkach wyświetla się podobne menu z wyborem kwoty wpłaty/wypłaty.

Funkcje DisplayDepositMoney() i DisplayWithdrawMoney() działają bardzo podobnie, lecz wyświetlają inny tekst. Obie docelowo uruchamiają funkcję FloatInputMenu(), w której użytkownik musi wprowadzić kwotę, którą chce wpłacić/wypłacić. Podczas wypłaty sprawdzane jest dodatkowo czy stan konta pozwala na taką wypłatę. Jeżeli nie – wyświetlany jest odpowiedni komunikat.

3.2.6 Naliczanie odsetek

Co określony czas (np. 10 minut) zostanie dodana do konta odpowiednia kwota (ustalony procent). Jeżeli program nie był uruchomiony przez jakiś czas – zostaje wykonana kalkulacja odsetek. Ostatnie uruchomienie programu będzie przechowywane w pliku.

Założenie niezrealizowane.

3.2.7 Baza danych w pliku

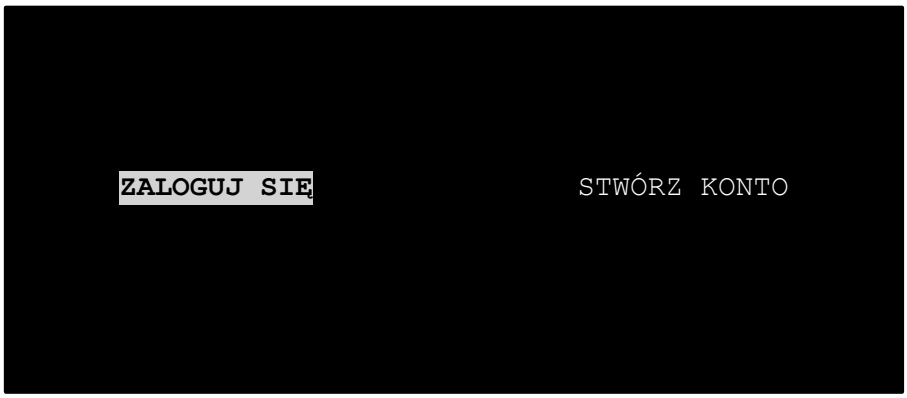
Użytkownik z odpowiadającym mu numerem konta, hasłem i stanem konta będą przechowywane w pliku tekstowym.

Wszyscy użytkownicy, którzy istnieją w systemie są przechowywani w bazie danych w pliku tekstowym. Funkcje ModifyUserInFile(), GetLastId(), FindLineContainingText() i SaveUser() odpowiadają odpowiednio za modyfikację odpowiedniego użytkownika w pliku, pozyskanie ostatniego numeru ID w pliku, wyszukiwanie linii zawierającej użytkownika i zapisywanie nowego użytkownika do pliku. Dodatkowo funkcja InitializeFiles() jest wywoływana, gdy nie znaleziony zostanie odpowiedni plik. Plik przechowywany jest w tej samej lokalizacji co plik wykonywalny.

3.2.8 Wiele instancji

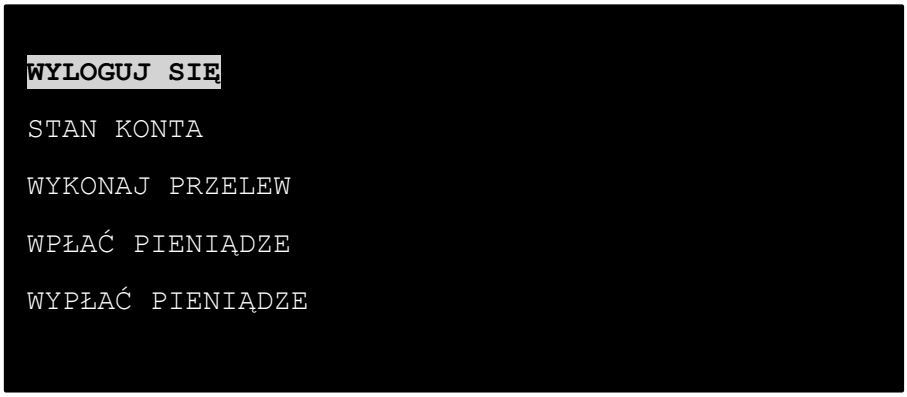
Istnieje możliwość uruchomienia wielu instancji programu, przez co może zalogować się wielu użytkowników na raz. Używany jest wtedy jeden plik (plik .exe i .txt muszą znajdywać się w tej samej lokalizacji). Program jest odpowiednio zabezpieczony przed tym, aby stan konta nie spadł poniżej 0 (choć wyświetlanie takiego stanu zostało uwzględnione).

3. Grafiki koncepcyjne dla wybranych funkcjonalności



Rysunek 1 Ekran startowy

Rysunek 2 Ekran logowania



Rysunek 3 Podstawowe menu

4. Podsumowanie

5.1 Problemy i przemyślenia

- Mimo kompilacji PDCursesMOD ze wsparciem dla UTF-8, nie udało mi się uzyskać polskich znaków
- Obsługa bazy danych w pliku .txt jest raczej trudna (w porównaniu do dedykowanej bazy danych) jak i bardzo niebezpieczna. Użytkownik może po prostu edytować plik i zmienić każdą wartość.
- Taki system powinien być podzielony na (co najmniej) 2 części – frontend do którego ma dostęp użytkownik i backend – który wykonuje wszystkie operacje na serwerze, do którego bezpośrednio nie ma dostępu użytkownik.
- Zmienne zmiennoprzecinkowe nie są najlepszą metodą do przechowywania stanu konta poprzez ich niedokładność. W kodzie użyłem long long int podzielonego przez 100, aby przechowywać stan konta, lecz nie jest to też najlepsze rozwiązanie. Jest w stanie przechować majątek aktualnie najbogatszego człowieka na ziemi w PLN, ale gdyby waluta była dużo mniej warta, mógłby wystąpić overflow. Dużo lepiej byłoby przechowywać kwotę w innej formie, na przykład tablicy.

6. Raport z przebiegu realizacji projektu

Historia zmian jest także opisana poprzez wiadomości podczas commitów na GitHubie. Jest tam dostępny także kod, który został zmieniony podczas każdej zmiany. Repozytorium git jest również załączone w pliku .zip

13.12.2020 – pierwsza pomyślna kompilacja PDCursesMOD, stworzenie plików i repozytorium na GitHubie

25.12.2020 – stworzenie podstawowego menu

26.12.2020 – stworzenie menu, które automatycznie horyzontalnie środkuje elementy

27.12.2020 – poprawienie funkcji horyzontalnego wyświetlania elementów, ustawienie stylu automatycznego formatowania kodu na takie odpowiadające standardom firmy Google, dodanie podświetlenie aktualnie wybranego elementu w menu

29.12.2020 – stworzenie funkcji wertykalnego menu

30.12.2020 – tworzenie pliku bazy danych

02.01.2021 – stworzenie menu do tworzenie konta

03.01.2021 – dodano możliwość wprowadzania i usuwania tekstu podczas tworzenia konta

04.01.2021 – zmiana funkcji tworzenia konta na ogólną funkcję wprowadzania tekstu, dodano podstawową weryfikację danych w formularzu, dodanie ukrywania hasła podczas jego wpisywania, dodano wsparcie dla znaków specjalnych (~!@#$%^&*()_+{}|:”<>?-=[]\;’,./), dodanie funkcji zapisywania użytkownika do pliku, dodano funkcję wyszukiwania ostatniego id

05.01.2021 – poprawiono funkcję tworzenia użytkownika, ponieważ zawsze tworzyła użytkownika z takimi samymi danymi

06.01.2021 – stworzono funkcję do generowania numeru konta

07.01.2021 – naprawiono ostrzeżenia dotyczące typów, dodanie funkcji znajdującej i wczytującej użytkownika z pliku do pamięci

08.01.2021 – dodanie warunku podczas tworzenia konta, aby nie można było utworzyć konta z takim samym loginem, dodanie funkcji wyświetlania stanu konta

09.01.2021 – dodanie funkcji umożliwiającej modyfikację użytkownika w bazie danych

10.01.2021 – naprawienie funkcji do modyfikacji użytkownika w bazie danych, ponieważ modyfikowała niepoprawnego użytkownika, dodanie funkcji wyświetlającej menu do wpisywania liczb z przecinkiem, dodanie funkcji wpłaty/wypłaty, znormalizowanie funkcji (-1 mówi, że użytkownik wcisnął ESC lub że coś poszło nie tak, więc cofa się menu o 1), dodanie instrukcji podczas wprowadzania liczby z przecinkiem

12.01.2021 – zmiana typu zmiennej przechowywującej stan konta z float na long long int, aby zapobiec niedokładności przy liczbach zmiennoprzecinkowych – reszta kodu została odpowiednio zmodyfikowana, żeby to wspierać, naprawiono klawisz ESC podczas

wpłaty/wypłaty – za każdym razem odejmowana lub dodawana została liczba -1 ze stanu konta

13.01.2021 – zapobiegnięcie memory leak, dodanie powiadomienie o niewystarczającym stanie konta podczas wypłaty, dodanie możliwości transferu gotówki, zapobiegnięcie powtarzaniu się numerów konta, zdiagnozowanie memory leak (poprzez narzędzie „Memory Usage" i „Take Snapshot") i naprawienie go, dodanie szyfrowania haseł, przebudowa kodu, aby funkcje znajdowały się pod funkcją main(), dodanie wyświetlania numeru konta podczas sprawdzania stanu konta

14.01.2021 – naprawiono polskie znaki (gdy kompilowano w anglojęzycznej wersji Visual Studio) poprzez dodanie „/execution-charset:windows-1250" do dodatkowych opcji podczas kompilowania projektu, dodano odświeżanie użytkownika z pliku przed każdą podjętą akcją, aby program wspierał wiele instancji pracujących na tej samej bazie danych, naprawiono problem z zaznaczeniem podczas TextInputMenu(), ponieważ możliwe było zaznaczenie nieistniejących opcji

23.01.2021 – dodanie dodatkowego zabezpieczenia przed wysłaniem przelewu do niewłaściwego użytkownika, poprzez sprawdzanie loginu z numerem konta, dodanie informacji o obsłudze programu podczas wyświetlania menu wprowadzania tekstu

25.01.2021 – dodanie przeładowanej funkcji FloatInputMenu(), która pozwala na wyświetlenie tego menu nie znając wymiarów konsoli (tekst wyświetla się w lewym górnym rogu, zamiast na środku), zmiana rozszerzenia pliku z .c na .cpp i naprawienie błędów z tym związanych