**Kasa zapomogowo-pożyczkowa**

Autorzy:

/hidden/

Spis treści:

# Krótki wstęp

# Szczegółowa analiza wymagań użytkownika

# Podstawowe obiekty i słownik danych globalnych

# Opis algorytmów działania programu wraz z wyszczególnieniem punktów krytycznych

# Ustalenie co należy przechowywać w bazie danych, ilustracja bazy poprzez tabele

# Projekt interfejsu użytkownika. Założenia ogólne, lista dialogów, ekranów, formatek

# Plan testowania

# Proponowany harmonogram implementacji projektu, podział zadań pomiędzy członków zespołu

1. **Krótki wstęp:**

Projekt utworzony dla Kasy Zapomogowo-Pożyczkowej będzie programem obsługującym system pożyczkowy rzeczonej Kasy. Klient wymaga, aby system Kasa Zapomogowo-Pożyczkowa realizował poniższe wymagania:

1. Możliwość łączności z bazami danych, dotyczących klientów banku i pożyczek danych klientów**.**
2. Interfejs użytkownika na być wykonany klarownie
3. Możliwość dodawania, usuwania, modyfikowania danych dotyczących klientów
4. Możliwość wyszukiwania w liście klientów zadłużenia na określony dzień
5. Sortowanie listy klientów po nazwisku i po wysokości pożyczki
6. Możliwość sprawdzenia, którzy klienci kończą spłacać pożyczkę w tym miesiącu

Wykorzystane zostaną takie narzędzia jak:

Struktura opisująca kolejne osoby z bazy danych oraz niektóre elementy powyższych struktur (np. numer dowodu).

Imię

Nazwisko

Adres

PESEL

Numer Dowodu

Numer umowy o pożyczkę (dla danego klienta)

Struktura opisująca poszczególne pożyczki:

Numer umowy o pożyczkę

Kwota Pożyczki

Data Pożyczki

Ilość miesięcy do spłaty

Stan na dzień bieżący (ile teraz osoba jest winna bankowi)

Oprocentowanie

W dokumentacji przedstawione zostaną przedstawione kolejne wymagania użytkowników, pomysły programistów, słownik globalnych danych (zmienne opisujące dane, nazwy funkcji), opis działania algorytmów, projekt interfejsu użytkownika, planowanie testów oraz plan implementacji projektu wraz z harmonogramem działania członków zespołu.

1. **Szczegółowa analiza wymagań użytkownika:**

Wymagane jest, aby:

1. Interfejs użytkownika zawierał komentarze opisujące poszczególne opcje (np. do opcji wyszukaj szczególny komentarz //wyszukuje osobę o podanym nazwisku spośród bazy danych……
2. Rozpoczynając prace użytkownik podawał hasło (dowód uprawniający do korzystania z systemu)
3. System obsługiwał bazę danych zawierającą do 1 miliona klientów
4. Opcjonalnie, realizacja dynamicznego alokowania pamięci (oszczędzanie pamięci operacyjnej)
5. **Podstawowe obiekty i słownik danych globalnych**

Pliki:

- Dane o pożyczkobiorcach zapisane w pliku osoba.txt

- Dane o pożyczkach w pliku p.txt

- Plik zmniejszający złożoność o nazwie index.txt

Zmienne globalne:

- int i,j,k,l – zmienne pomocnicze, ich ilość zależy od sposobu realizacji projektu

Struktury:

- klient – największa ze struktur która umożliwi stworzenie tablicy struktur

- data\_pozyczki - potrzebna do obliczenia odsetek, stanu na dzień bieżący itp.

- data\_aktualna – potrzebna do obliczenia odsetek stanu na dzień bieżący itp.

Funkcje:

- dodaj\_klienta //dodanie klienta do bazy danych

- usun\_klienta //usunięcie klienta z bazy danych

- mod\_klienta //modyfikacja danych klienta

- sort\_nazw //wyświetlanie listy klientów posortowanej po nazwisku

- sort\_kwota //wyświetlanie listy klientów posortowanej po wysokości pożyczki

- zadłużenie //sprawdzanie ile zadłużony jest dany klient

- splata\_teraz //wyświetlanie listy klientów którzy skończą spłacać pożyczkę w tym miesiącu

- menu //funkcja do obsługi interfejsu

- haslo //funkcja odpowiadająca za zabezpieczenie

# Opis algorytmów działania programu wraz z wyszczególnieniem punktów krytycznych

**Dodawanie klienta:**

Dopisanie danych klienta na końcu bazy klientów

**Usuwanie klienta:**

Odszukanie danego klienta w bazie

Usunięcie jego danych

Wklejenie w jego miejsce ostatniego klienta

**Modyfikowanie danych klienta:**

Dodanie klienta:

Dopisanie danych klienta na końcu bazy klientów

Usunięcie starych danych:

Odszukanie danego klienta w bazie

Usunięcie jego danych

Wklejenie w jego miejsce ostatniego klienta

**Sortowanie listy klientów po nazwisku:**

Quicksort dla kodów ASCII

**Sortowanie listy klientów po wysokości pożyczki:**

Quicksort

**Sprawdzanie zadłużenia dla osoby o danym nazwisku:**

Sortuj po nazwisku

Sprawdź stan zadłużenia dla danego nazwiska

Wyświetl stan zadłużenia

**Hasło:**

W pętli zaporowej porównujemy hasło użytkownika z hasłem systemowym

Jeśli hasło jest prawidłowe kontynuujemy wykonanie programu(funkcja menu)

# Ustalenie co należy przechowywać w bazie danych, ilustracja bazy poprzez tabele

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Imię | Nazwisko | PESEL | Nr dowodu | Kwota pożyczki | Data pożyczki | Ilość miesięcy | Stan na dzień bieżący | Odsetki | Nr pożyczki |
| char | char | string | string | float | struktura data | int | float | float | int |

1. **Projekt interfejsu użytkownika:**

Funkcja haslo() w pętli zaporowej powtarza żądanie o hasło

Funkcja menu() wypisuje poniższy interfejs:

1. Dodaj klienta
2. Usuń klienta
3. Modyfikuj dane klienta
4. Wyświetl listę klientów posortowaną po nazwisku
5. Wyświetl listę klientów posortowaną po wysokości pożyczki
6. Sprawdź, którzy spośród klientów kończą spłacać pożyczkę w tym miesiącu
7. Sprawdź zadłużenie danego klienta
8. Wybierz dodatkowe opcje:
9. Wyświetl posortowaną listę klientów od najbardziej zadłużonego
10. Wyświetl posortowaną listę klientów po roku urodzenia
11. Znajdź klientów po nazwisku
12. Znajdź klientów po dacie pożyczki
13. Wyświetl posortowaną listę po ilości pożyczek na klienta (od 5 do 1)
14. Znajdź klienta po identyfikatorze
15. Zakończ
16. **Plan testowania**

Testy w trakcie pisania kodu:

- Testy działania konkretnych funkcji np. sortowania po nazwisku, w trakcie implementacji i natychmiastowe poprawki ewentualnych błędów działania.

- Próbne wypisywanie tablicy struktur do pliku dla sprawdzenia spójności wprowadzonych/wyjściowych danych.

Testy po zaimplementowaniu interfejsu użytkownika:

- Sprawdzenie wszystkich opcji menu włącznie z opcjami dodatkowymi, poprawa ewentualnych błędów działania.

Działania dodatkowe (opcjonalne) :

- Korekta wyświetlania danych dla łatwiejszego użycia przez użytkownika

# Proponowany harmonogram implementacji projektu, podział zadań pomiędzy członków zespołu

|  |  |  |
| --- | --- | --- |
| /hidden/ | /hidden/ | /hidden/ |
| - dokumentacja  - interfejs  - testowanie  - opcje dodatkowe/spójność | -dokumentacja  - obsługa plików  - in/out danych  - funkcje/algorytmy | - dokumentacja  - sortowanie  - działania na bazie danych  - funkcje/algorytmy |