REV	DATA	ZMIANY
0.1	07.02.2021	*Jakub Legutko (jlegutko@student.agh.edu.pl)*

BAZA SIMPLEDB

Autor: Jakub Legutko
Akademia Górniczo-Hutnicza

Kraków 2021

Spis treści

Lista oznaczeń

SFML	Simple and Fast Multimedia Library
FSM	Finite State Machine
GUI	Graphical User Interface

1. Wstęp

Dokument dotyczy programu działającego jako prosta baza danych z określoną liczbą pól w każdym rekordzie, program pozwala na tworzenie nowych pól, odczyt już istniejących, zapis rekordów do pliku oraz usuwanie stworzonych rekordów. Obsługa bazy jest umożliwiona z użyciem klawiatury oraz GUI zaprojektowanego z użyciem bibliotek SMFL.

This document describes a program, which acts as a simple database, with a set amount of fields per record, the program allows for creation of new records, reading of existing fields, saving the records to a text file and deletion of existing records. Access is enabled through the keyboard and a GUI created using SFML.

2. Funkcjonalność

Na funkcjonalność programu składa się:

1. Tworzenie rekordów zawierających 5 pól, uzupełnianych danymi z klawiatury, wymagane jest wypełnienie jedynie ostatniego pola w rekordzie – pozostałe mogą być puste.
2. Odczytywanie zawartości rekordu o wybranym przez użytkownika indeksie i wyświetlenie jego zawartości na ekranie.
3. Usuwanie najstarszego rekordu (tego z indeksem 1).
4. Zapisywanie wszystkich rekordów do pliku tekstowego.
5. Odczyt listy rekordów z pliku tekstowego przy uruchomieniu programu.
6. Odwracanie kolejności rekordów przy każdorazowym uruchomieniu programu.

3. Projekt techniczny

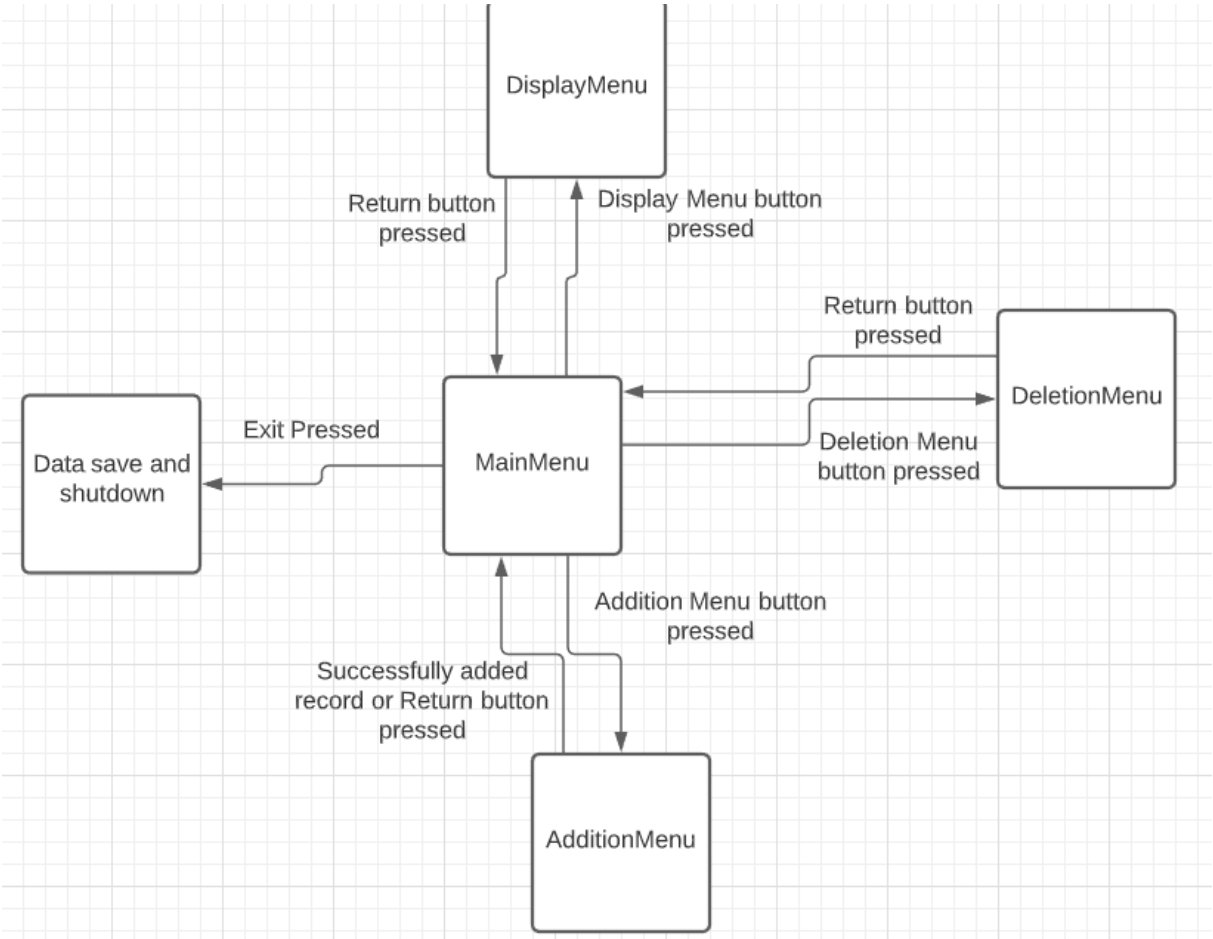


Figure 1 - The program is based on a FSM, its diagram is shown in the figure

Jeżeli program nie zostanie zamknięty poprawnie, to znaczy zostanie zamknięty inaczej niż poprzez przycisk Exit, wszystkie dane obecnie wczytane do bazy ulegną zniszczeniu.

Cały program oparty jest o FSM pokazany wyżej, każdy blok FSM ma swoją strukturę wewnętrzną opartą na eventach z SFML, w zależności od wybranego klawisza podejmowana jest akcja.

Poniżej zaprezentowana jest przykładowa struktura wewnętrzna submenu dodawania rekordu, najbardziej skomplikowana struktura w programie.

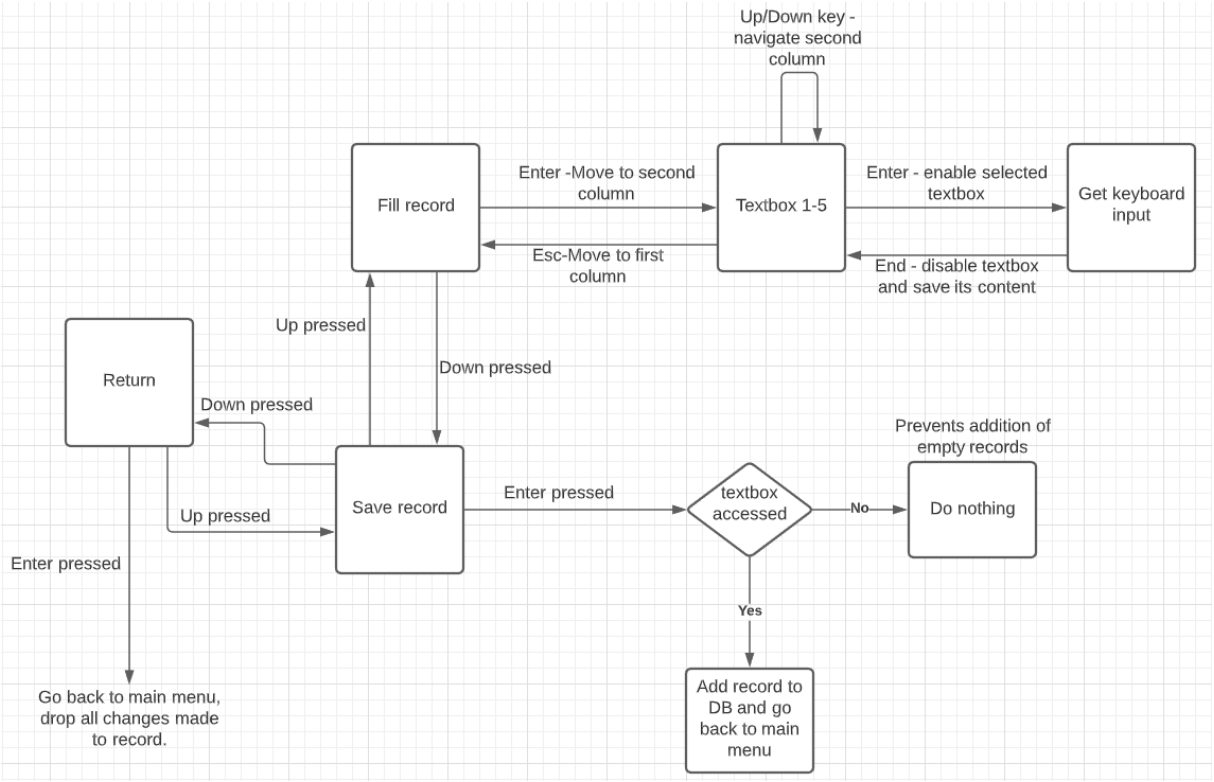


Figure 2 - example of addition submenu

Każde menu to osobna klasa, wszystkie są oparte na podobnym schemacie, jednak zawierają unikatowe dla siebie metody i zmienne poza elementami wspólnymi.

4. Opis realizacji

Program oparty jest na klasach odpowiadających za poszczególne menu , klasie Textbox odpowiadającej za pole obsługujące czytanie tekstu podawanego przez użytkownika, oraz klasie SimpleDB, która jest rdzeniem programu, jej metody odpowiadają za działanie bazy danych.

Poza tym w programie obecna jest jedna funkcja:

```
1.  bool is_digits(const std::string& str)
2.  {
3.      return std::all_of(str.begin(), str.end(), ::isdigit);
4.  }
```

Sprawdzająca, czy podana przez użytkownika wartość jest cyfrą, by uniknąć problemów przy konwersji. Mechanizm ten wykorzystany jest na przykład w jednej z metod klasy DisplayMenu, by zapewnić, że program otrzyma liczbę jako indeks rekordu do wyświetlenia.
Ponadto obecny jest wcześniej wspomniany FSM pozwalający na nawigację w programie.
FSM - typ enumeracyjny, zawiera 4 elementy, odpowiadające obecnie aktywnemu menu. Zmiana wartości następuje w momencie wciśnięcia Enter na wybranym przycisku w GUI, zobrazowane w **Figure 1.**

SimpleDB – główna klasa programu, składająca się przede wszystkim z wektora:

`vector<Dane_pola> DB_in_prog;`

przechowującego struktury typu Dane_pola, które działają jako rekordy:

```
1.  //This struct works as a record.
2.      struct Dane_pola {
3.          string wiek;
4.          string PESEL;
5.          string imie;
6.          string nazwisko;
7.          string liczba_500_plus;
8.
9.      };
```

Ta klasa zawiera ponadto 6 metod służących do obsługi programu :

- `void Init()`: Uruchamiane podczas startu programu, tylko jeden raz. Wykorzystuje strumienie do plików by odczytać zawartość ostatnio zapisanej bazy danych, jeżeli takowa istnieje. Odczytywanie realizujemy aż do momentu znalezienia końca pliku. W każdym cyklu odczytu zbierane są dane o tylko jednym rekordzie, jeżeli wartość ostatniego pola jest pusta, rekord nie zostanie wczytany do bazy, jest to mechanizm zabezpieczający przed powstawaniem pustych rekordów przy starcie programu, jak również działanie celowe, gdyż jest to pole najważniejsze w każdym rekordzie. Dane wprowadzane są do pomocniczej struktury i następnie umieszczane na końcu wektora przechowującego je.
- `int Get_amount_of_records()`: Ta metoda to prosty getter informujący nas o długości wektora przechowującego dane – wykorzystywane na przykład w funkcji usuwającej, by poinformować użytkownika ile rekordów pozostało do usunięcia.
- `void Read_DB(int input_choose_rec, string& imie, string& nazwisko, string& pesel, string& wiek, string& bomble)`: Metoda wyciąga wartości z rekordu o indeksie `input_choose_rec - 1` i umieszcza je w korespondujących stringach podanych poprzez referencję jako argumenty do metody, by mogły zostać wyświetlone na ekranie przez metodę z klasy DisplayMenu.

- `void Write_Data(string imie, string nazwisko, string PESEL, string wiek, string bomble)`: Wartości argumentów są przypisywane do tymczasowej struktury typu Dane_pola, następnie struktura jest umieszczana na końcu wektora służącego za pamięć bazy.
- `void Delete(int field_num)`: Usuwa z wektora pamięci element o indeksie podanym poprzez argument field_num. Jeżeli rekord o danym indeksie nie istnieje, wyświetli się informacja w konsoli.
- `void Save_Exit()`: Wywoływane tylko w momencie prawidłowego zamknięcia programu, to jest z poziomu MainMenu poprzez przyciśnięcie Enter przy wybranej pozycji Exit. Zapisuje pojedynczo każdy rekord do tymczasowej struktury używając na wektorze pamięci metody pop_back(), co jednocześnie usuwa dany rekord z pamięci, następnie zapisuje jej zawartość do pliku i powtarza aż do wyczerpania długości wektora pamięci. Po ukończeniu zapisu kończy działanie programu.

MainMenu – pierwszy element GUI, który widzi użytkownik zaraz po uruchomieniu programu, służy jako centrum nawigacyjne po programie. Metody tej klasy to:

- `void draw(sf::RenderWindow& window);` - przekazuje aktualny stan klasy do okna, tak by zmiany mogły zostać pokazane na ekranie.
- `void MoveUp();` - wywoływana gdy naciśnięta zostanie strzałka w górę, przesuwa zaznaczenie na element wyżej, o ile to możliwe. Działa w oparciu o ItemIndex. Dla użytkownika zmiana ta jest obrazowana poprzez zmianę koloru wyświetlanego elementu.
- `void MoveDown();` - działanie odwrotne do MoveUp();
- `int GetPressedItem();` - getter zwracający wartość ItemIndex, by móc określić w programie, który przycisk jest wybrany. Dla użytkownika wybór ten jest obrazowany poprzez podświetlenie wartości aktualnie wybranej na zielono.

Klasa ta posiada też prywatne zmienne:

- `int ItemIndex;` - używane do nawigowania po menu, patrz metody MoveUp/Down.
- `sf::Font font;` - określa czcionkę używaną w tym menu.
- `sf::Text text[MAX_MENU_ITEMS];` - generuje pola tekstowe wypełniane następnie tekstem pozwalającym użytkownikowi na wybranie akcji.

Textbox – klasa tworząca pole tekstowe z możliwością wpisania danych przez użytkownika, pozwala też na poprawę wpisanego tekstu, poprzez usuwanie klawiszem Backspace:

- `void inputLogic(int charTyped)` – Metoda ustalająca, czy wciśnięty klawisz powinien zostać dopisany na końcu pola tekstowego, czy jest to jeden z klawiszy funkcyjnych pola (Enter, Backspace, Escape).
- `void deleteLastChar()` – wywoływana przez klawisz funkcyjny Backspace, usuwa ostatni znak z pola tekstowego.
- `void typedOn(sf::Event input)` – Metoda odpowiedzialna za działanie programu, w zależności od wciśniętego przycisku wywołuje odpowiednią metodę klasy po uprzednim sprawdzeniu przycisku metodą inputLogic.
- `bool getSelection()` – informuje, czy pole tekstowe jest obecnie aktywne.
- `void drawTo(sf::RenderWindow& window)` – przesyła wartość pola tekstowego do okna, by umożliwić aktualizację.
- `std::string getText()` – zwraca zawartość pola tekstowego.
- `void setSelected(bool sel)` – ustawia pole tekstowe jako aktywne/nieaktywne w zależności od wartości sel.
- `void setLimit(bool ToF, int lim)` – ustawia/aktualizuje limit znaków w polu tekstowym.
- `void setPosition(sf::Vector2f point)` – ustawia pozycję pola tekstowego na oknie.

- `void setFont(sf::Font& fonts)` – ustawia czcionkę używaną w polu tekstowym.

AdditionMenu – Menu to pozwala na dodawanie rekordów przez użytkownika, realizowane poprzez instancje klasy Textbox. Klasa zawiera 5 takich instancji, po jednej na każde pole w rekordzie. Przy tworzeniu rekordu wymagane jest wypełnienie ostatniego – najważniejszego pola, gdyż jeżeli pozostanie ono puste, rekord nie zostanie zapisany przy zamknięciu programu. Jest to działanie zamierzone z dwóch powodów opisanych w **SimpleDB.Init().**

To menu poza metodami obecnymi w MainMenu zawiera dodatkowo:

- `void SwitchToInput();` - zmienia indeks obecnie używany na indeks drugiej kolumny, zawierającej pola Textbox. Wizualnie zmienia kolor tekstu opisu komórki w drugiej kolumnie.
- `void SwitchBack();` - działanie odwrotne do SwitchToInput.

Oraz:

- `void MoveUpSecondRow();`
- `void MoveDownSecondRow();`
- `int GetSecondRowIndex();`

Które działają analogicznie do wersji bez SecondRow.

DisplayMenu – Odpowiedzialne za prezentację zawartości rekordu pobranego metodą SimpleDB.Read_DB(), składa się z pól tekstowych oraz Textbox pozwalającego na wybór numeru rekordu do pokazania oraz metod obecnych w MainMenu. Dodatkową metodą jest:

- `void update_amount(std::string amount_of_records);` - zmienia wartość wyświetlaną w polu tekstowym, informuje użytkownika, ile rekordów znajduje się w bazie. Aktualizuje dane na podstawie odczytu długości wektora pamięci.

DeletionMenu – Składa się z interfejsu informującego na bieżąco (co każdą zmianę) o ilości dostępnych rekordów, przycisku usuwania, oraz przycisku powrotu. Wciśnięcie Enter przy wybranym przycisku usuwania uruchamia SimpleDB.Delete(1). Argument informujący który rekord zostanie usunięty, jest w tym momencie ustawiony na stałe jako 1, co spowoduje usunięcie najstarszego rekordu

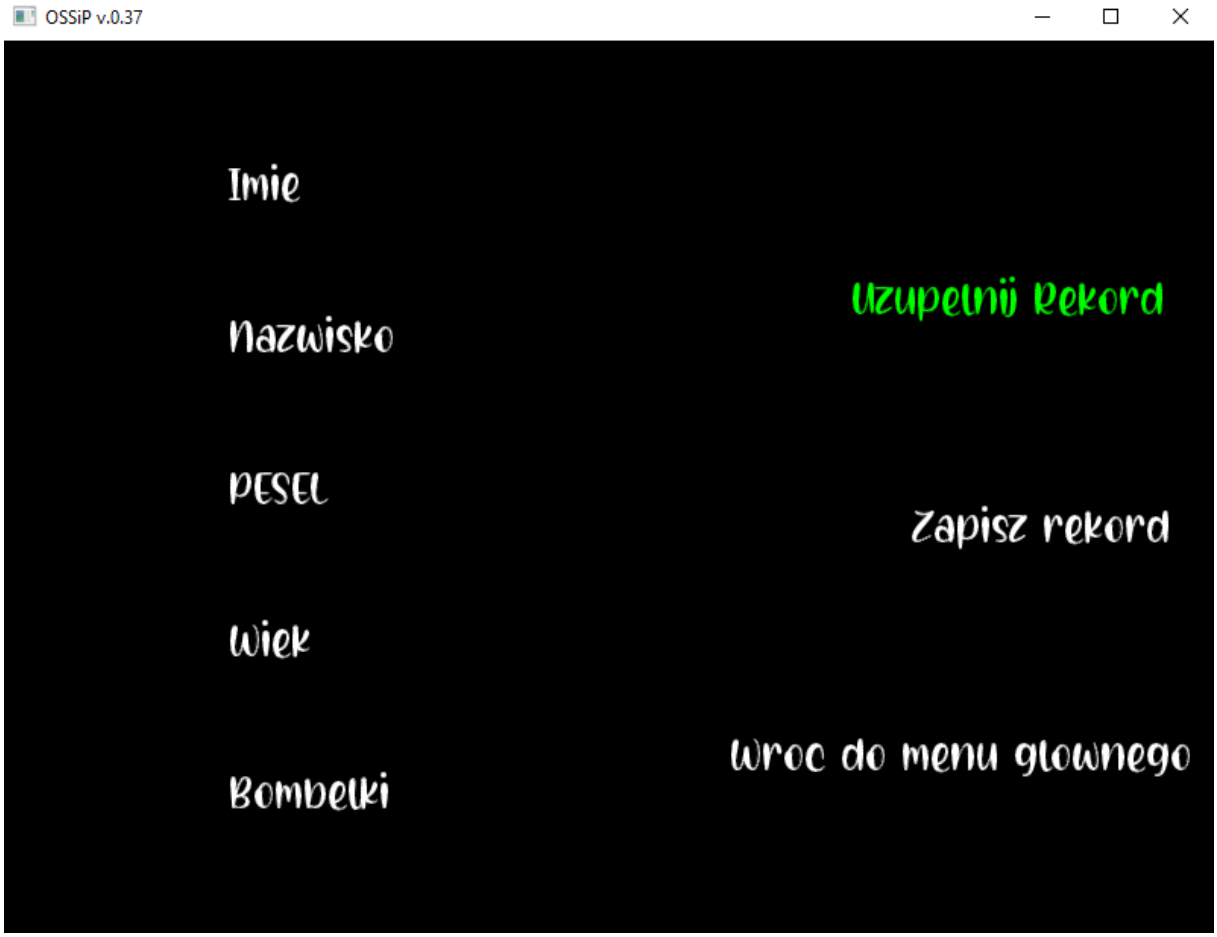
w tym uruchomieniu programu, czyli tego, który ma obecnie indeks 1. To menu zawiera te same metody co DisplayMenu, różni się jedynie tekstem wprowadzonym do poszczególnych pól tesktowych oraz ich liczbą.

5. Podręcznik użytkownika

Program przeznaczony jest do działania w systemach Windows, został skompilowany przy użyciu Visual Studio, zalecane jest korzystanie z wersji dostępnej w katalogu Release projektu, dostępnego tutaj:
https://mega.nz/file/8BkCAA7S#rNS62vduLjkpaeqMbw8G1Em6yY1GMxqmKsOGslabZqg

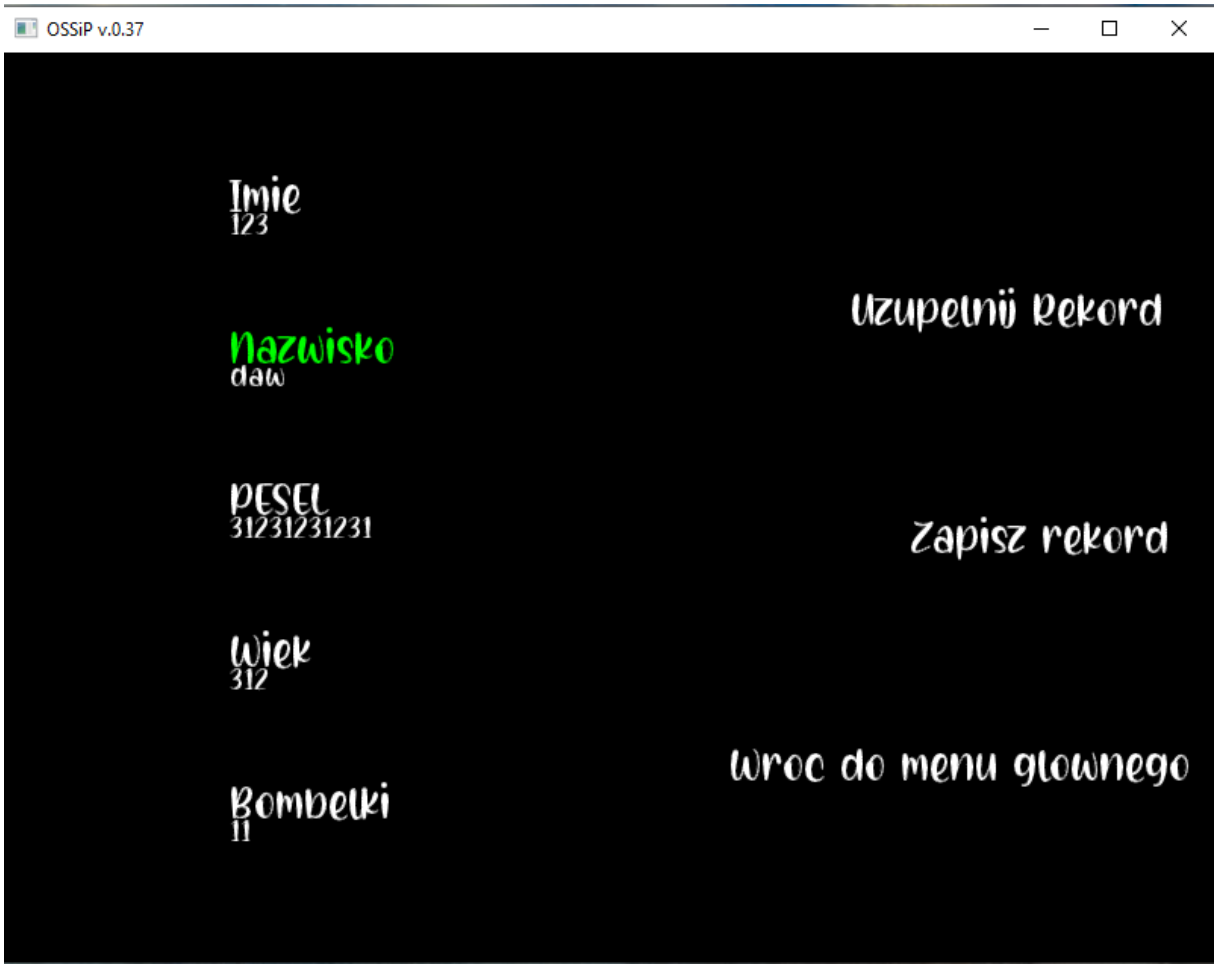
Kod źródłowy najnowszych, niekoniecznie stabilnych wersji znajdzie się tutaj:
https://github.com/JakubLegutko/SimpleDB

Powyższy screen obrazuje menu główne widziane przez użytkownika przy uruchomieniu programu. Do nawigacji służą klawisze strzałek góra/dół, wybrany element jest podświetlony na zielono, a wybór zatwierdzamy klawiszem Enter. Jako czcionka użyty został Creamy Sunset, ze względu na podobieństwo do popularnego i poważanego w środowisku programistycznym Comic Sans.

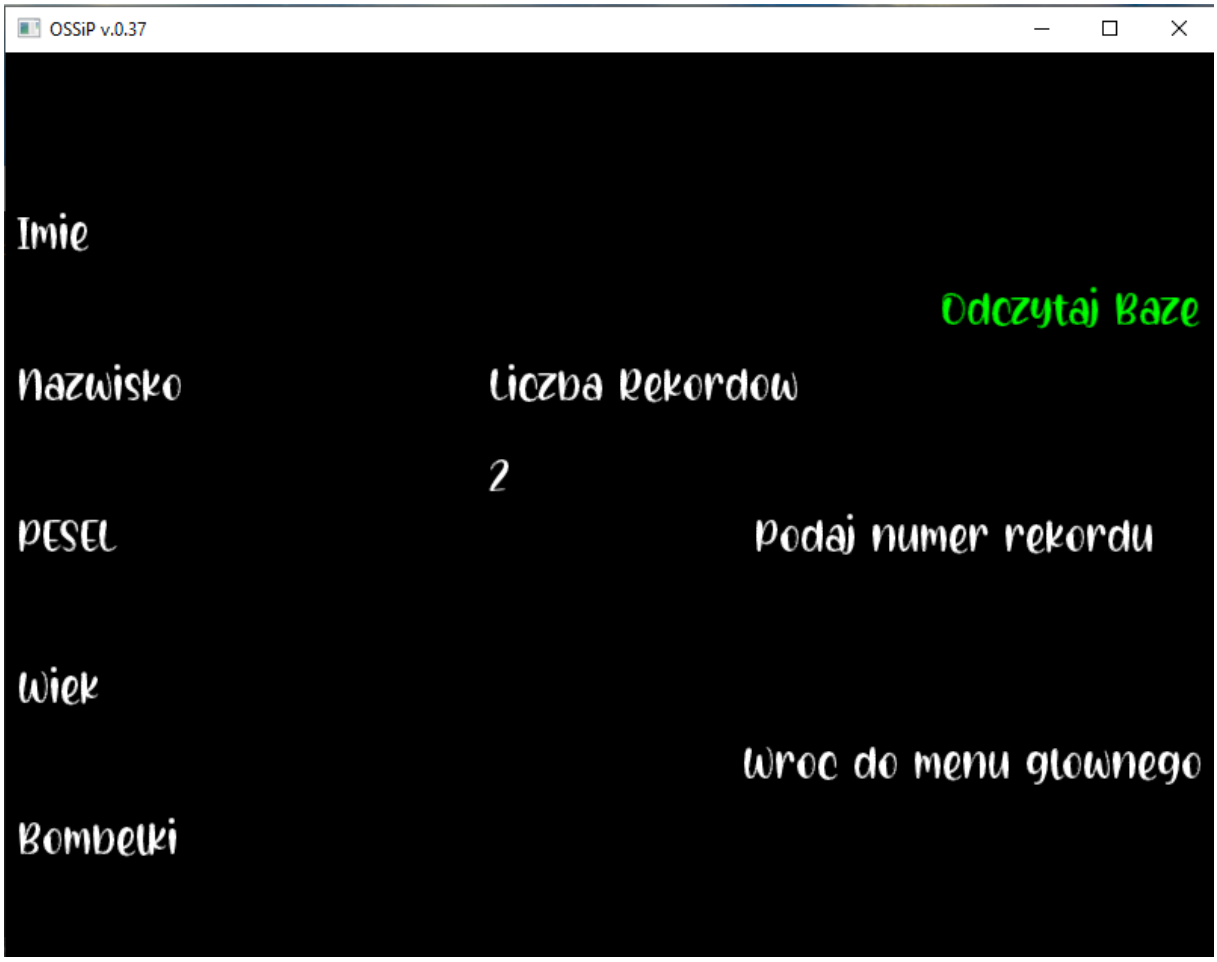


Menu zapisu składa się z dwóch kolumn, przycisk uzupełnij rekord pozwala na przejście do drugiej kolumny złożonej z pól tekstowych pozwalających na wpisywanie danych.

Nawigacja po pierwszej kolumnie działa podobnie jak w menu głównym, przycisk zapisz rekord aktywuje się dopiero w momencie wpisania danych do którejkolwiek z komórek, jednak najważniejsza jest ostatnia komórka, bez niej rekord zostanie wprawdzie utworzony, jednak nie zostanie zapisany przy wyłączeniu programu, gdyż takie rekordy są nam absolutnie zbędne.

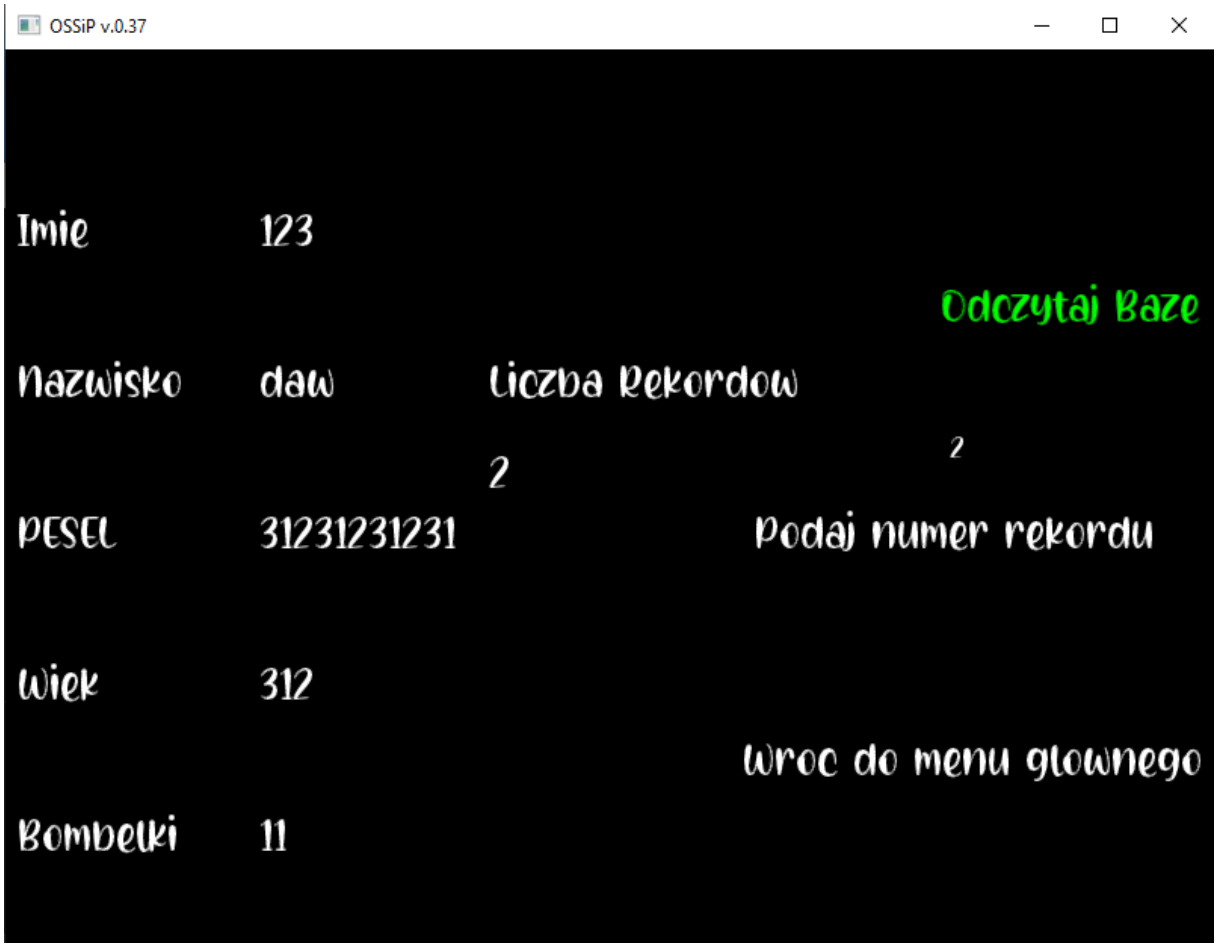


Po naciśnięciu Enter na przycisku uzupełnij rekord zostaniemy przeniesieni do drugiej kolumny, tutaj nawigacja odbywa się klawiszami strzałek, aby uaktywnić dane pole wciskamy klawisz Enter, aby opuścić pole tekstowe używamy klawisza End. Pola tekstowe imie, nazwisko oraz Bombelki nie są ograniczone, jednak PESEL przyjmuje tylko 11 znaków, a pole Wiek tylko 3, gdyż sugerując się opakowaniami LEGO, ludzie nie mogą być starsi niż 123 lata. Aby powrócić do pierwszej kolumny, by zapisać lub pominąć rekord, używamy klawisza Esc.
Po przyciśnięciu przycisku zapisu wrócimy do menu głównego, a rekord będzie dostępny w bazie.



Menu odczytu rekordów wyświetla na bieżąco liczbę rekordów dostępnych do odczytania, by odczytać rekord nawigujemy do przycisku podaj numer rekordu, naciskamy Enter i wpisujemy liczbę odpowiadającą numerowi indeksu żądanego rekordu. Warto pamiętać, iż przy każdorazowym uruchomieniu programu, kolejność rekordów zostaje odwrócona, jest to nawiązanie do słów biblijnych, gdzie głoszono, iż ostatni będą pierwszymi.

Przycisk odczytaj bazę jest niedostępny, aż do momentu podania dozwolonego indeksu rekordu, numerujemy od 1.



Przykład działania odczytu na podstawie wcześniej stworzonego rekordu.

Ostatnim dostępnym menu jest menu usuwania, gdzie można wyżywać się na stworzonych rekordach do woli, nawigujemy za pomocą strzałek oraz klawisza Enter, którego naciśnięcie na przycisku usuń rekord spowoduje anihilację rekordu o obecnym indeksie 1. Usuwanie nie zadziała jeżeli nie mamy już rekordów, należy pohamować wtedy swoje zapędy i wrócić do menu tworzenia.

```
daw
31231231231
312
11
123
```

Po naciśnięciu przycisku Wyjdź w menu głównym, rekordy dostępne w programie zostaną zapisane do pliku DB.txt, co następnie umożliwi wczytanie ich do programu przy kolejnym uruchomieniu.

Bibliografia

[1] Program powstał na podstawie informacji uzyskanych na wykładach autorstwa B. Cyganek w ramach przedmiotu Języki Programowania Obiektowego, kierunek Elektronika i Telekomunikacja, rok 3., WIEiT, AGH, 2020/2021.

[2] Wykorzystano dokumentację biblioteki SFML https://www.sfml-dev.org/