AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

Sudoku

z przedmiotu

Programowanie obiektowe

Elektronika, II rok

Jakub Sędłak

Grupa: poniedziałek 18:30

prowadzący: Rafał Frączek

13.01.2025

1. Opis projektu

Projekt implementuję popularną grę „Sudoku” w języku programowania C++. Celem projektu jest stworzenie aplikacji umożliwiającej stworzenie planszy, rozwiązanie i sprawdzanie poprawności wypełnianego Sudoku.

2. Project description

The project implements a popular game ‘Sudoku’ in C++ programming language. The main objective of the project is to make an application that allows users to solve Sudoku puzzles and automatically validates whether the solution is correct.

3. Instrukcja użytkownika

Sudoku polega na uzupełnieniu każdego pola cyframi 1,2,3,…,9. Aby Sudoku było poprawnie rozwiązane:

- Nie może powtórzyć się cyfra w danym wierszu
- Nie może powtórzyć się cyfra w danej kolumnie
- Nie może powtórzyć się cyfra w danym bloku 3 x 3

Jeżeli, którykolwiek z tych warunków nie jest spełniony Sudoku należy poprawić.

Program umożliwia:

- Rozpoczęcie nowej gry Sudoku na jednym z czterech poziomów trudności
- Rozwiązanie danego Sudoku poprzez wybór koordynatów X i Y oraz wartości
- Sprawdzenie kolejno poprawności:
    - Wierszy
    - Kolumn
    - Kwadratów 3x3
- Automatyczny zapis oraz możliwość (ale nie konieczność) wczytania ostatniego zapisu.

Uruchomienie gry następuje przez skompilowanie i uruchomienie pliku wykonywalnego. Interfejs działa w trybie konsolowym.

4. Kompilacja

Projekt może być skompilowany standardowo. Jednym wartym zaznaczenia warunkiem poprawnej kompilacji jest kompilowanie w standardzie minimum ISO C++17 (optymalnie ISO C++20). Testy kompilacji projektu zostały przeprowadzone z użyciem programu Visual Studio o standardzie ISO C++20 na systemie operacyjnym Windows 11 Home.

Polecenie do kompilacji przy użyciu GCC: „ g++ -o sudoku main.cpp ”

5. Pliki źródłowe

Projekt składa się z następujących plików:

- ‘main.cpp’ : Zawiera główną logikę programu, w tym interfejs użytkownika i funkcję gry.

6. Zależności

Brak

7. Opis klas

W programie dostępne są poniższe klasy:

- ‘ICellWrite’ : Interfejs służący do obsługi zapisywania wartości z danej komórki macierzy planszy Sudoku
    - *void virtual Write(int input) = 0;* jest to metoda interfejsu służąca do zapisu danych do komórki. Metoda ta zostanie nadpisana poprzez metodę klasy Cell.
- ‘ICellRead’ : Interfejs służący do obsługi odczytywania wartości z danej komórki macierzy planszy Sudoku
    - *int virtual Read() = 0;* jest to metoda interfejsu służąca do odczytu danych z komórki. Metoda ta zostanie nadpisana poprzez metodę klasy Cell.
- ‘Cell’ : Klasa dziedzicząca od powyższych interfejsów reprezentująca pojedynczą komórkę macierzy planszy Sudoku.
    - int content; zawartość komórki macierzy planszy Sudoku (przetrzymuję wpisaną cyfrę)
    - Cell(int def = 0) konstruktor parametryczny z przypisaną wartością domyślną. Inicjalizuje wartość domyślną zawartości komórki.
    - void Write(int input) override metoda nadpisująca interfejs. Służy do zapisu do int content.
    - int Read() override metoda nadpisująca interfejs. Służy do odczytu z int content.
- ‘SudokuDatabase’ : Klasa dziedzicząca od klasy Cell zarządzająca planszą Sudoku i operacjami na niej
    - vector<vector<Cell>> matrix; Wektor wektorów klasy Cell służy do stworzenia macierzy i przetrzymywania i organizacji danych w macierzy.
    - SudokuDatabase(int def = 0) : Cell(def) Konstruktor parametryczne z wartością domyślną 0 służący do inicjalizuji planszy Sudoku jako macierz 9 x 9 zawierającą w każdej komórce klasę Cell.
    - vector<vector<Cell>> GetMatrix() Getter dla planszy Sudoku
    - void WriteCell(int PosX, int PosY, int input) Metoda służąca do zapisu wartości Input do komórki (X,Y)
    - void SaveToFile() Metoda służąca do zapisu planszy Sudoku do pliku „Save.txt” w celu ponownego wczytania po ponownym uruchomieniu aplikacji.
    - void LoadFromFile() Metoda służąca do odczytu z pliku „Save.txt” i wczytania danych do planszy Sudoku. Metoda ta jest używana przy wczytaniu ostatnio zapisanej planszy.

W projekcie występują również:

- ostream& operator<<(ostream& out, SudokuDatabase& sdb) Przeciążenie operatora wyjścia w celu, łatwego oraz estetycznego przedstawienia planszy Sudoku na terminalu
- void DaRules() Funkcja służąca do wypisania na terminalu zasad Sudoku
- bool counter(vector<Cell> v) Funkcja pomocnicza służąca do sprawdzenia czy w komórkach wektora v powtarza się liczba. Wartość true – jakaś cyfra się powtarza. Wartość false – brak powtórzeń
- bool ERRORRow(SudokuDatabase& sdb) Funkcja służąca do sprawdzenie poprawności wierszy Sudoku
- bool ERRORCol(SudokuDatabase& sdb) Funkcja służąca do sprawdzenia poprawności kolumn Sudoku
- bool ERRORSqu(SudokuDatabase& sdb) Funkcja służąca do sprawdzenia poprawności kwadratów 3x3
- bool TestALL(SudokuDatabase& sdb) Funkcja za pomocą powyższych trzech funkcji sprawdza czy istnieje błąd w Sudoku
- bool Win(SudokuDatabase& sdb) Funkcja sprawdza czy jest nieuzupełnione miejsce oraz czy nie występuje błąd. Jeżeli plansza jest uzupełniona oraz nie ma błędu to wyświetla komunikat oraz kończy grę.

8. Zasoby

W projekcie wykorzystywane są następujące pliki zasobów:

- `Save.txt` – plik zawierający ostatni zapis z rozgrywki Sudoku. Struktura pliku:
  - 9 linii tekstu w każdej linii po 9 znaków
  - Zapis ten symuluje macierz 9 x 9, która jest reprezentacją macierzy Sudoku. Na przykład 4 znak z 3 linii to jest odpowiednik wartości z 4 kolumny, 3 wiersza.

9. Dalszy rozwój i ulepszenia

Możliwe drogi rozwoju projektu:

- Rozwinięcie algorytmu odpowiedzialnego za stworzenia początkowej planszy Sudoku. Obecnie generowana jest na podstawie uzupełnionej planszy Sudoku. Można rozwinąć ten algorytm stosując większą ilość gotowych planszy oraz kod odpowiedzialny za wybranie jednej. Potencjalne plansze mogły by być przechowywane w pliku tekstowym i wczytywane za pomocą funkcji podobnej do tej do co jest odpowiedzialna za wczytanie gry z zapisu.
- Dodanie możliwości zapisu wyników. Na przykład tak zwany leaderboard, na którym zawarty byłby zapisany na przykład poziom trudności oraz czas rozwiązania.
- Dodanie możliwości wyboru widoczności wiadomości z podpowiedzią związaną z tym, czy istnieje błąd w obecnym Sudoku

10. Inne

Brak