Sprawozdanie z realizacji projektu "Saper"

1. Podręcznik użytkownika

Zasady gry

Gra polega na odkrywaniu pól na planszy tak, aby uniknąć bomb. Każde odkryte pole pokazuje liczbę bomb w sąsiednich 8 polach (0 oznacza brak bomb), gracz musi wybrać punkty które są w przedziale od 1 do końca przedziału(zależnie od poziomu który się wybrało). Gracz może:

- **Odkrywać pola** (komenda r X Y)
- **Stawiać flagi** (komenda f X Y) w miejscach podejrzewanych bomby
- Wygrywa po odkryciu wszystkich pól niebędących bombami
- Przegrywa po odkryciu bomby

Sposób uruchomienia

./saper [-p POZIOM] [-f PLIK]

Opcje:

- -p POZIOM: Wybór poziomu trudności:
    - łatwy: 9x9, 10 bomb
    - średni: 16x16, 40 bomb
    - trudny: 16x30, 99 bomb
    - niestandardowy ROWS COLS BOMBS: Własne parametry
- -f PLIK: Tryb plikowy – wczytuje planszę i komendy z pliku

Przykłady:

Tryb interaktywny - poziom średni

./saper -p średni

Tryb niestandardowy

./saper -p niestandardowy 10 15 10

Tryb plikowy

./saper -f plansza.txt

Format pliku w trybie plikowym

- Pierwsze ROWS linii zawiera planszę (liczby i B dla bomb)
- Kolejne linie zawierają komendy w formacie:

Przykładowy plik:
0 0 1 B 2 1 1 1 1
0 0 1 1 2 B 1 1 B
0 0 0 0 2 3 3 3 2
1 1 0 0 2 B B 2 B
B 1 0 0 2 B 3 2 1
1 1 1 1 2 1 1 0 0
1 1 2 B 1 0 1 1 1
1 B 2 1 1 0 1 B 1
1 1 1 0 0 0 1 1 1
r 3 3
r 2 6
r 1 3
r 3 6

2. Szczegóły implementacji

Podział na moduły

Moduł	**Opis funkcjonalności**
main.c	Główna logika programu, obsługa argumentów
plansza.c	Inicjalizacja planszy, generowanie bomb
wejsciewyjscie.c	Interakcja z użytkownikiem, zapis wyników
utils.c	Funkcje pomocnicze

Kluczowe funkcje

// plansza.c

void placeBombs(char board[][COLS], int n); // Losowe rozmieszczenie bomb

void bialePole(int x, int y, ...); // Rekurencyjne odkrywanie pustych pól

// utils.c

int sprawdzSasiadow(...); // Liczy bomby w sąsiedztwie

void wspolzedne(...); // Parsuje współrzędne z inputu

// wejsciewyjscie.c

void zapisGraczy(int score); // Zapis wyników do pliku

Struktury danych

// Struktura dla rankingu graczy

```
typedef struct {
   char name[MAX_NAME_LENGTH];
   int score;
} Player;
```

Testy i wyniki

Scenariusz testowy	**Rezultat**	**Uwagi**
Poziom łatwy – próba odkrycia bomby	Przegrana	Bomby poprawnie wykrywane
Pełne odkrycie planszy i położenie flag	Wygrana	Mechanizm wykrywania wygranej działa
Tryb plikowy z poprawnymi danymi	Sukces	Plansza wczytuje się prawidłowo
Niestandardowa plansza 3x4 z 1 bombą	Poprawna inicjalizacja	Walidacja parametrów działa
Zapis wyników	Top 5 w pliku	Sortowanie wyników poprawne

3. Podział pracy (projekt indywidualny)

Całość projektu została zaimplementowana przeze mnie Kacpra Tattarkiewicza.

4. Podsumowanie

Program poprawnie się uruchamia i jest w stanie obsłużyć zdecydowaną większość przypadków mimo tego nie zaimplementowałem resetowania się tablicy, ponieważ nie wiedziałem że trzeba to w ten sposób zrobić aż do dzisiaj, kolejną rzeczą jest nieponumerowanie górnego rzędu i lewej kolumny aby łatwiej było graczowi odczytać które pole wybiera, niestety trochę miałem mało czasu a to mi trochę sprawiało trudność. Po zrobieniu tego wszystkiego nauczyłem się że lepiej wszystko od razu powinno się robić w sposób jak ma być na końcu (pomyślałem że zrobię trochę inny input i zamiast r X Y po prostu na początku zrobiłem X Y i to spowodowało że przez dłuższy czas szukałem błędów, również na początku zamiast B na bomby miałem -1 i to też trochę problemów mi przysporzyło). Dodatkowo dla mnie trochę ciężki była praca nad trybem plikowym, długo mi zajęło szukanie funkcji rewind nawet nie wiedziałem że to istnieje i okazuje się że w niektórych sytuacjach to bywa bardzo przydatne. Na szczęście przezwyciężyłem większość z tych trudności i program dobrze funkcjonuje.