**Sprawozdanie z realizacji projektu "Saper"**

**1. Podręcznik użytkownika programu**

**Opis zasad gry:** Gra polega na odkrywaniu pól na planszy, unikając min. Na każdym odkrytym polu znajduje się liczba wskazująca ilość min w sąsiednich polach. Gracz może również oznaczać pole flagą, aby oznaczyć że według niego znajduje się tam mina. Gra kończy się przegraną, gdy gracz odkryje pole z miną, lub wygraną, gdy odkryje wszystkie pola, które nie zawierają min i zakryje wszystkie miny flagami.

**Sposób uruchomienia programu:** Aby uruchomić grę, należy wywołać program w terminalu. Program poprosi o wybór poziomu trudności:

* 1: Łatwy (9x9, 10 min)
* 2: Średni (16x16, 40 min)
* 3: Trudny (16x30, 99 min)
* 4: Własny poziom, gdzie gracz definiuje wymiary planszy i liczbę min (aczkolwiek ten tryb jest nie rankingowy i nie będzie wliczał wyniku do listy najlepszych graczy).

Program obsługuje również tryb wczytywania z pliku, w którym gracze mogą przechodzić przez wcześniej zapisane ruchy i stan gry.

**Opis poszczególnych opcji:**

* r x y – Otwiera pole na współrzędnych x, y.
* f x y – Ustawia lub usuwa flagę na współrzędnych x, y.

Po zakończeniu gry, program wyświetla 5 najlepszych wyników graczy i wynik właśnie ukończonej rozgrywki, prosi o podanie pseudonimu gracza i jeżeli uzyskał odpowiedni wynik zapisuje go wraz z wynikiem w rankingu.

**2. Szczegóły implementacji programu**

**Podział programu na moduły:**

* **menu.c / menu.h** – Obsługuje menu wyboru poziomu trudności, wczytywanie poziomu własnego oraz pozycji startowej sapera.
* **pola.c / pola.h** – Zajmuje się obliczaniem liczby min wokół każdego pola.
* **generuj.c / generuj.h** – Odpowiada za losowanie rozmieszczenia min na planszy.
* **gra.c / gra.h** – Zawiera logikę rozgrywki, w tym odkrywanie pól, obsługę flag oraz obliczanie wyników.
* **main.c** – Główna funkcja programu, która uruchamia grę lub wczytuje dane z pliku, inicjalizuje planszę i zarządza interakcjami z użytkownikiem.

**Interfejs kluczowych funkcji:**

* **generuj()** – Losuje rozmieszczenie min.
* **pola()** – Oblicza liczbę min wokół każdego pola.
* **wypisz()** – Wyświetla aktualny stan planszy.
* **gra()** – Obsługuje główną pętlę gry, sprawdza zakończenie gry oraz liczy punkty.
* **koniec()** – Sprawdza, czy gra zakończyła się wygraną lub przegraną.
* **To\_koniec()** – Inicjalizuje sekwencje końcową gry: porażkę lub interfejs wygranej

**Opis ważniejszych struktur:**

* **Tablica T** – Reprezentuje planszę, na której wartości -1 oznaczają miny, a inne liczby to liczba min wokół danego pola.
* **Tablica O** – Przechowuje stan odkrycia pól (0 – zakryte, 1 – odkryte, -1 – flaga).

**Opis i wynik poszczególnych testów:** Wszystkie funkcje zostały przetestowane pod kątem poprawności logiki, szczególnie funkcje odpowiedzialne za generowanie min oraz obsługę odkrywania i flagowania pól. Testy zakończyły się pozytywnie.

**3. Podział pracy w zespole**

* Projekt wykonywałem w jednoosobowym zespole, całość pracy została wykonana przez Gniewka Wasilewskiego

**4. Podsumowanie**

**Czy wszystkie funkcjonalności udało się zaimplementować?**   
Tak, wszystkie wymagane funkcjonalności zostały zaimplementowane, w tym obsługa menu, losowanie planszy, flagowanie, odkrywanie pól, zapisywanie danych gracza.

**Co sprawiało trudności?**   
Największe wyzwanie stanowiło tworzenie pliku makefile, poprawianie drobnych problemów związanych z zakończeniem gry oraz stworzeniem dobrej funkcji rekurencyjnej odkrywającej puste pola.

**Problemy i rozwiązania:**   
Podczas testów wystąpiły problemy z zapisem oraz sortowaniem danych użytkownika do pliku z wynikami innych graczy, problem został rozwiązany poprzez dobranie odpowiedniego typu otwarcia pliku.

**Wnioski:**Gra jest w pełni funkcjonalna, a dzięki modularnej strukturze kodu jest łatwa do rozbudowy o dodatkowe funkcje.