Projekt rozpocząłem od napisania poglądowego algorytmu samej gry w pliku szkic.py – link:

https://github.com/niemazaco/saper/blob/master/szkic.py

Dobre jego napisanie umożliwiło lepsze rozeznanie, jakie pozostałe elementy będą potrzebne w programie, a także umożliwiło szybkie przerobienie zawartości tego pliku do działania w później napisanych klasach.

Następnie przystąpiłem do napisania interfejsu gry w stopniu umożliwiającym jego uruchomienie, rozpoczęcie gry i sprawdzanie poprawności działania dodawanych później elementów klas odpowiadających za logikę gry. Później dodawałem stopniowo kolejne elementy gry i na bieżąco je testowałem, dzięki czemu wiedziałem, gdzie szukać błędów i mogłem szybko je naprawiać.

Testowałem program pod różnymi kątami, które przychodziły mi do głowy, uwzględniając również wszystkie testy zadane w założeniach projektu. Z moich doświadczeń wynika, że program działa poprawnie.

Zgodnie z założeniami projekt posiada:

- po lewej stronie interfejsu: pola tekstowe do wprowadzania danych (wysokość, szerokość planszy oraz ilość bomb) i walidację tych danych w oparciu o mechanizm wyjątków. Są też liczniki informujące o ilości pytajników, wolnych pól (do odkrycia przez LPM do osiągnięcia wygranej), ilość „pozostałych bomb" oraz label informujący gracza o stanie programu, wskazujący co powinien robić.

-po prawej stronie interfejsu jest miejsce przeznaczone na planszę (siatka przycisków) o stałym rozmiarze przycisków.

- obsługa kodu xyzzy – w miejscach, w których znajdują się bomby interfejs podmienia tła jasne na odpowiednie tła ciemne, a stany tych pól używanych przez algorytm są odpowiednio zmieniane. Same przyciski zawierają tła wyświetlające pożądany obrazek oraz liczbę podającą ilość sąsiadujących pól z minami.

Licznik pozostałych dekrementuje się przy postawieniu flagi, niezależnie od tego, czy jest ona postawiona w dobrym miejscu. W momencie gdy partia zostaje przegrana, licznik pozostałych bomb informuje ile rzeczywiście bomb zostało nieoflagowanych. W przeciwnym razie pokazuje on ilość bomb zadaną w parametrach początkowych gry minus ilość postawionych flag, tak jak w saperze na Windowsie 7. Obowiązek stawiania flag w dobrych miejscach spoczywa na graczu. Rzeczywistą ilość pozostałych bomb oczywiście uwzględnia algorytm w klasie Plansza, a jedynie nie jest podawana do wiadomości gracza.

Wyjątek jest podnoszony również przy zerowej ilości bomb oraz ilości bomb równej ilości pól na planszy – uruchamianie gry przy takich ustawieniach jest pozbawione sensu.

Dodałem również kontrolę sekcji krytycznej. Zmienna _aktualizacja w klasie Logika jest ustawiana na True przed każdorazowym wejściem do obiektu klasy Plansza. Powinno to zmniejszyć ryzyko równoległych wejść do obiektu klasy Plansza w sytuacji, gdyby inne procesy znacznie spowolniłyby działanie sapera.

a. Lambda-wyrażenia

Akcja na przyciskach planszy (obsługa LPM i PPM):

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L190-L200

Bindowanie klawiszy do obsługi kodu xyzzy:

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L65-L71

Akcja na przycisku start (LPM):

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L113-L117

b. List comprehensions

Dwuwymiarowa macierz planszy przechowująca po dwa pola – nr stanu pola oraz obecność bomby (True/False):

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/logika.py#L41

Dwuwymiarowa lista przycisków w klasie Interfejs:

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L190-L194

c. Klasy,

e. Moduły

Realizacja podziału programu na dwie klasy:

Moduł interfejs.py zawiera klasę Interfejs odpowiadającą za wywoływanie akcji na przyciskach oraz za ustawianie elementów interfejsu graficznego. W ten sposób część odpowiadająca za interfejs została wydzielona z pozostałej części programu. Ponadto moduł zawiera walidację danych wejściowych i utworzone w tym celu klasy wyjątków.

Moduł logika.py i klasa Logika odpowiada za poprawność algorytmu sapera. Część klasy Logika została wydzielona do klasy Plansza, aby można ją było łatwo usuwać w całości i tworzyć na nowo.

d. Wyjątki

Moje klasy wyjątków:

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L3-L50

Zastosowanie w kodzie:

https://github.com/niemazaco/saper/blob/275f91938bb52c0bf32ffb6fd704656b5313bd8e/interfejs.py#L149-L176