**Tytuł projektu**: Gold fever

**Autor**: Magdalena Rzepka

**Fabuła gry**:

Nasz śmiałek, chcąc się szybko wzbogacić, ruszył na poszukiwanie złota do owianej złą sławą, starej kopalni. Nie spodziewał się jednak, że od wieków mieszkają tam wygłodniałe węże. Gdy poszukiwacz przemierzał kopalnie, niechcący zawalił sobie drogę powrotną. Jedyne co mu pozostało, to dalej szukać złota w nadziei, że uda mu się w międzyczasie odnaleźć wyjście z kopalni, której drogi są niczym labirynt.

**Zakończenie:**

W grze występują dwa alternatywne zakończenia.

1)Bohater stracił nadzieję na wyjście z kopalni i popełnił samobójstwo.

2)Bohater stanął oko w oko z wężem. Po pojedynku, z bohatera została tylko nerka, której wąż nie miał ochoty zjeść.

**Działanie i zasady gry:**

Po włączeniu gry ukazuje się nam plan kopalni, bohater, trzy węże oraz bryłka złota. Po prawej stronie wyświetlana jest ilość bryłek złota, które nasz bohater ma aktualnie przy sobie. Poniżej licznika, znajduje się napis, mówiący o tym, jak popełnić samobójstwo i zakończyć rozgrywkę.

Na planie kopalni znajdują się ściany, przez które ani bohater ani węże nie mogą przejść. Bryłka złota pojawia się na losowanej pozycji.

Celem gry jest zdobycie jak największej liczby bryłek złota.

Aby zebrać złoto bohater musi wejść na pole, na którym znajduje się bryłka.

Przeszkadzać będą mu w tym 3 węże. Spotkanie z wężem może nastąpić, kiedy wąż wejdzie na pole, na którym stoi bohater lub kiedy oboje udadzą się na to samo pole. Gdy wąż stoi na polu obok, można spróbować wystraszyć go idąc na pole, na którym się znajduje. Najczęściej węże są strachliwe i czując, że bohater się nie boi, mogą ustąpić, lecz nie dzieje się tak zawsze.

Po śmierci bohatera ukazuje się nam napis, świadczący o śmierci. Z dalszej części tekstu dowiadujemy się, że ktoś przemierzając te kopalnie, natknął się na martwe ciało i worek złota.

Następnie gra kończy się.

**Sterowanie**:

* w lub strzałka w górę, aby poruszyć postacią o jedno pole w górę
* s lub strzałka w dół, aby poruszyć postacią o jedno pole w dół
* a lub strzałka w lewo, aby poruszyć postacią o jedno pole w lewo
* d lub strzałka w prawo, aby poruszyć postacią o jedno pole w prawo

**Zawartość, która dla czytelności programu znajduje się w module funkcje.c składa się z:**

**Mapy:**

Mama 15 x 15 pól stworzona jest poprzez wpisanie do tablicy dwuwymiarowej cyrf 0, 1, 2 lub 3.

0 oznacza, że dane pole nie ma ściany ani u góry ani z prawej strony.

1 oznacza, że dane pole ma ścianę u góry, a nie ma jej z prawej strony.

2 oznacza, że dane pole ma ścianę z prawej strony, a nie ma jej na górze.

3 oznacza, że dane pole ma ścianę i u góry i z prawej strony.

**Struktury pole:**

składa się z dwóch zmiennych int x, int y. Struktura ta pozwala na zapisanie współrzędnych.

**Na\_(lewo/prawo/dole/gorze)\_ściana:**

sprawdza, czy na lewo/prawo/dole/górze znajduje się ściana i zwraca wartość true, jeśli znajduje się tam ściana, lub false w przeciwnym przypadku. Robi to za pomocą odwołania się do mapy na konkretnych, podanych do funkcji współrzędnych.

Lewy/prawy/gorny/dolny\_brzeg\_planszy:

sprawdza, czy konkretne współrzędne, podane do funkcji, znajdują się w obrębie mapy.

**Gdzie\_najbliżej:**

sprawdza dla konkretnych, podanych do funkcji współrzędnych przeciwnika (węża), po jakiej współrzędnej powinien się poruszyć, aby móc podążać jak najlepiej za przeciwnikiem.

Oblicza on wartości bezwględne po x i po y dla pola przeciwnika i pola bohatera. Jeśli znajdują się na tej samej współrzędnej po x lub po y, to zwraca tą drugą współrzędną. W przeciwnym przypadku zwraca współrzędną, po której jest najbliżej do bohatera.

**To\_samo\_pole:**

sprawdza, czy pole, na które chce udać się przeciwnik, jest tym samym polem, na którym był ruch wcześniej. Jeśli jest to to samo pole zwraca true lub false w przeciwnym przypadku.

**Zajete:**

sprawdza, czy pole, którego współrzędne są podanego do funkcji, jest już zajęte przez jakiegoś węża lub bryłkę złota.

**Zabojstwo:**

sprawdza, czy podane do funkcji współrzędne pola bohatera pokrywają się ze współrzędnymi jakiegoś węża. Zwraca true, jeśli jest to prawda lub false w przeciwnym przypadku.

**Plik naglowki.h:**

plik ten zawiera biblioteki, z których korzysta program oraz nagłówki funkcji opisanych w pliku funkcje.c.

Funkcje z pliku main.c:

* int init sprawdza, czy inicjacja SDL przebiegła pomyślnie
* int loadMedia wczytuje i sprawdza poprawność wczytanych bitmap
* void close zamyka grę

**Struktura kodu w main:**

* deklarujemy zmienne do liczenia punktów.
* ustawiamy bitmapy mapy, bohatera, węży, punktów oraz złota na konkretnych miejscach na ekranie
* za pomocą funkcji init i loadMedia sprawdzamy, czy nie wyrzuca błędów
* jeśli wszystko do tej pory poszło pomyślnie, następuje fragment odpowiedzialny za przypisywanie konkretnemu wciśniętemu klawiszowi, odpowiedniej funkcji
* w dalszej części wyznaczamy najlepszy możliwy ruch dla 3 węży
* następuje sprawdzenie, czy bohater nie stoi na tym samym polu co złoto. Jeśli tak, licznik punktów zwiększa się o 1, a współrzędne bryłki złota są na nowo losowane.
* sprawdzamy, czy nasza postać jeszcze żyje. Jeśli tak, ekran się aktualizuje i gra toczy się dalej. Jeśli nie w miejscu naszej postaci wyświetla się to, co z niej zostało (bitmapa nerki).
* Po śmierci naszej postaci widzimy napisa końcowy, opowiadający dalszą część historii oraz liczbę zebranych przez nas punktów.
* Gra kończy się,

**Biblioteki:**

* time.h
* SDL2/SDL.h
* SDL2/SDL\_image.h
* SDL2/SDL\_ttf.h

**Kompilacja dla CodeBlocks na systemie Windows:**

Doinstalowanie biblioteki SDL2:

* ściągnąć SDL2-devel-2.0.10-mingw.tar.gz.
* Zidentyfikować katalog, w którym znajduje się codeblocks.
* Zidentyfikować podkatalog MinGW w tym katalogu.
* z i686-w64-mingw32/bin skopiować sdl2.dll do MinGW/bin.
* z i686-w64-mingw32/lib skopiować wszystko do MinGW/lib.
* z i686-w64-mingw32/include skopiować katalog SDL2 do MinGW/include.
* w codeblocks stworzyć nowy projekt (C, console application).
* w Settings > Compiler > Linker settings w polu "Other liner options”

dopisać: -lmingw32 -lSDL2main -lSDL2 .

Doinstalowanie pozostałych bibliotek będzie odbywała się analogicznie, z wyjątkiem dopisywania w “Other liner options”.

* dla SDL\_image.h będzie to -lSDL2\_image
* dla SDL\_ttf.h będzie to -lSDL2\_ttf

Biblioteki time.h nie trzeba doinstalowywać.