Aknakereső Dokumentáció

> Ez az Aknakereső **macOS** és különböző **Linux** disztribúciók alatt fut.

A játék már játszható állapotban van:

- A főmenüből indíthatunk új játékot.
- Tud játékmenetet generálni, véletlenszerű akna elhelyezéssel, felhasználói bemenet alapján.
- A játékmenet közben minden lépés után ellenőrzi, hogy nyert-e a játékos.
- A játékos tud megjelölni vagy felderíteni mezőket.
- Felderítésnél minden környező nem akna mezőt felderít, és a felderített mezőknél írja hány akna van egy mező környezetében.
- Tud hibás bemenetet kezelni.

Hiányzó funkciók:

- A játék kezdete óta eltelt idő mérése és kiírása.

minesweeper.h

```
typedef struct gameTime {
    int min, sec;
} gameTime;

// Idő adattípus
```

```
typedef struct GameField {
    char **field, **visible, **opened;
    int size_X, size_Y, mine_C;
    gameTime timer;
} GameField;

/* A játékmenethez szükséges változókat és segédtömböket tartalmazó
adattípus.
 * Három tömböt tárol mely szükséges a játékmenethez:
 * A field tömbben tárjuk az aknák helyét, és a környező aknamezők számát.
 * A visible tömb az amit a játékos láthat, ezt módosítjuk csak a játék
során
 * Az opened tömbben rögzítjük a már felnyitott mezőket, ez a győzelemi
eset felismeréséhez kell.
 * A size_X és size_Y változók rögzítik a pálya méretét.
 * A mine_C változó a pályán található aknák mennyiségét rögzíti.
 * A gTime adattípusban tároljuk a játék kezdete óta eltelt időt.
*/
```

```
```

```c
typedef struct Coordinate {
    int x, y;
} Coordinate;

// Koordináta adattípus
```

```c
void gameLoop(GameField gf);

/* A game loop maga a játékmenet, a ciklusnak akkor van vége, amikor a
játékos
 * veszít, vagy nyer. Ameddig fut a ciklus, kirajzolja a pályát és várja
 * játékos lépépsét, amit a guesssing függvény dolgoz fel.
 * Paraméterek: GameField struktúra
 * Visszatérési érték: void
*/
```

```c
void guessing(GameField gf, Coordinate guess, int cmd);

/* A játékos lépését dolgozza fel a függvény.
 * A cmd változó lehet 1 vagy 2, ha 1, akkor felfedez, ha 2, akkor
megjelöl egy mezőt.
 * Ha aknamezőt próbál felfedni a játékos a gameloop leáll.
 * Ha nincs akna a mezőn, derítse fel a program a környező nem akna
mezőket.
 * Ha megjelölni szeretne egy mezőt a játékos, csak olyat tud ami még
nincs feldedve.
 * Paraméterek: GameField struktúra, Coordinate struktúra, Lépés tipusa
 * Viszzatérési érték: void
*/
```

```c
bool checkWin(GameField gf);

/* Ellenőrzi, hogy nyert-e a játékos.
 * Paraméterek: GameField gf
 * Visszatérési érték: bool
*/
```

```c
void endScreen(GameField gf, bool win);

/* A játék végén kiírt "Nyert" / "Veszített" szöveget írja ki, ha
veszített a játékos, felfedi az összes mezőt.
 * Leállítja a gameloop-ot.
 * Paraméterek: GameField struktúra, Egy bool érték (true ha nyert, false
ha veszített a játékos)
```

```
 * Visszatérési érték: void
*/
```

```
char adjacentMines(GameField gf, int x, int y);

/* Megszámolja, hogy egy mező környezetében hány akna található, és azt
rögzíti.
 * Paraméterek: GameField struktúra, X és Y koordináta
 * Visszatérési érték: char (Környező mezőkben lévő aknák száma)
*/
```

```
bool isMine(GameField gf, Coordinate c);

/* Ellenőrzi, hogy adott koordinátán található mezőn van-e akna.
 * Paraméterek: GameField struktúra, Coordináta struktúra
 * Visszatérési érték: bool
*/
```

```
void floodFill(GameField gf, int x, int y);

/* A "Flood Fill" rekurzív algoritmus megvalósítása, amely a környező nem-
akna mezők felfedéséért felel.
 * Paraméterek: GameField struktúra, X és Y koordináta
 * Visszatérési érték: void
*/
```

mainMenu.h

```
void mainMenu();

/* A főmenü működését vezérli.
 * Kiírja a menüpontokat, és a felhasználó választását feldolgozza.
 * Paraméterek: -
 * Visszatérési érték: void
*/
```

```
void newGame(GameField mf);

/* A függvény egy új játékmenetet indít el. Először bekéri a
felhasználótól, hogy
 * mekkora pályán és hány aknával szeretne játszani, majd legenerálja a
játékmenethez
```

```
 * szükséges 2D tömböket.
 * Paraméterek: GameField struktúra
 * Visszatérési érték: void
*/
```

```
char **allocateMemory(char **array, GameField gf);

/* Lefoglalja a kért dinamikus 2D tömböt, amely méretét a második
paraméterből kapja.
 * Paraméterek: Lefoglalandó tömb, GameField struktúra
 * Visszatérési érték: ** pointer a lefoglalt 2d tömbre
*/
```

```
void freeMemory(GameField gf);

/* Felszabadítja az összes dinamikusan foglalt tömböt, ami a játékmenethez
szükséges.
 * Paraméterek: GameField struktúra
 * Visszatérési érték: void
*/
```

render.h

```
void render(GameField gf, bool reveal);

/* Kirajzolja a pályát, amit a játékos láthat.
 * Ha a második paraméter true, feldedve rajzolja ki a pályát.
 * Paraméterek: GameField struktúra
 * Visszatérési érték: void
*/
```