Aknakereső programozói dokumentáció

# Környezet

Maga a játék egy külső grafikus könyvtárat használ, aminek raylib a neve (egyébként nem értem miért sdl-t használunk az egyetemen általánosságban és nem ezt). Erről a könyvtárról [itt](https://www.raylib.com/) található információ, a telepítése pedig [itt](https://github.com/raysan5/raylib/wiki/Working-with-CMake). A program [CLion](https://www.jetbrains.com/clion/)-ban lett megírva és [WSL](https://docs.microsoft.com/en-us/windows/wsl/install)-el volt forítva.

# Fájlok

* main.c – az előre elkészített függvények vannak meghívva benne, ez adja a program keretét
* field.c, field.h – a játéktábla mezőinek a függvényei, struktúrája található bennük
* table.c, table.h – a játéktábla függvényei és struktúrája található benne
* startscreen.c, startscreen.h – a játék megkezdése előtti kezdőképernyő függvényei találhatóak meg benne
* debugmalloc.h – ellenőrzi a memóriaszivárgást (de valamiért az sprint-nél a startscreen.c-ben nem működik, seg faultot ad vissza, ezért ki van kommentelve)

# Adatszerkezetek

Az adatszerkezetek neve úgy lett választva, hogy teljesen el lehessen igazodni azzal kapcsolatban, hogy minden struktúra és azokon belüli elemek mire használandók.

Field struktúra

* x, y – a táblában elhelyezkedése
* neighbourMineCount – a szomszédos mezőkön lévő aknák összege (ez arra kell, hogy ki tudjuk írni körülötte mennyi akna található)
* neighbours – ebben tárolja a szomszédokat (azokra mutató pointereket) (ez a rekurzív bejáráshoz kell, hogy ha egy mező körül nincs akna, akkor a körülötte lévőket is felfedje)
* neighbourCount – a szomszédos mezők száma (ez a neighbours tömb mérete)
* revealed – már felfedtük-e
* isMine – akna található-e rajta
* flagged – megkelöltük-e

Table struktúra

* colums, rows – a tábla sorainak és oszlopainak száma
* fieldSize – egy mezőnek a szélessége és magassága pixelben
* bombs- a táblán lévő bombák száma
* flags – a táblán lévő megjelölt mezők száma
* fieldsLeft – a fel nem fedett és aknát nem tartalmazó mezők száma (ahhoz kell, hogy tudjuk a játékos mikor nyer)
* state – a játék éppen milyen státuszban van (inGame, lost, won)
* currentTime –a tábla kirajzolásának ideje (az aktuális időpillanat számolásához kell)
* timeText, flagText – ebben van az aktuális kiírandó szövegek a játék közben
* fields – két dimenziós tömb, amiben a mezőkre mutató pointerek vannak

StartScreen struktúra

* tableWidth, tableHeight – a tábla magassága és szélessége, amit a kezdőképernyőn a felhasználó ad meg
* mineCount – a tábla mennyi aknát tartalmaz, amit szintén a felhasználó ad meg
* tableWidthText, tableHeightText, mineCountText – a felhasználó által megadott adatok kiírásához szolgáló karaktertömbök

# Függvények

field.c

* NewField – létrehoz egy új mezőt az alapértékekkel
* DrawField – kirajzolj egy mezőt az aktuális állapotának megfelelően
* ResetField – egy mező alapértékeit állítja be
* RevealField – egy mezőt fed fel, ha akna, akkor a játéknak vége, ha megjelölt akkor nem fedi fel, ha pedig már minden mezőt felfedtünk, akkor a játékstátuszt won állapotra változtatja, meghívja a függvényt a szomszédos mezőkre is ha szükséges
* FlagField – megjelöli a mezőt, ha nincs felfedve
* UnFlagField – törli a mező megjelölését

table.c

* NewTable – létrehoz egy új táblát és feltölti az alapértékekkel
* UpdateTable – frissíti a tábla állapotát, ezután van a tábla kirajzolása (a mezők felfedését és megjelölését figyeli, emellett azt is, hogy a játék véget ért-e)
* DrawTable – kirajzolja a tábla mezőit, amiket az UpdateTable függvény frissít le, kiírja a megmaradó megjelölések számát és az időt, emellet ha a játék véget ért, akkor annak eredményét
* CountFieldNeighbourMines– ezt a függvényt a ResetTable függvény használja ami az összes mezőre meghívja ezt, megszámolja a szomszédos mezőkön az aknák számát
* ResetTable – alaphelyzetbe hozza a táblát
* FreeTable – felszabadítja a lefoglalt memóriaterületet, amit a tábla és a mezői foglaltak le
* FormatTime – az időt a megfelelő formátumba hozza a kiíráshoz

startscreen.c

* NewStartScreen – létrehozza az új kezdőképernyőt, aminek az alapértékeit fájlból olvassa be (ha a fájl beolvasása nem sikeres, vagy nem létezik a fájl, akkor alapértékekkel), és kiírja, hogy a játék milyen tulajdonságokkal fog létrejönni (a tábla magassága, szélessége, aknák száma)
* UpdateStartScreen – frissíti a kezdőképernyőt, ennek segítségével tudja a felhasználó megadni a tábla adatait
* DrawStartScreen – kirajzolja a kezdőképernyőt
* FreeStartScreen – felszabadítja a kezdőképernyő által foglalt memóriát