Programozói dokumentáció

A programot szabvány c nyelvben írtam a CodeBlocks fordítóprogramban.

A main kódban van leírva az egér kattintások szabályozása, amiket a modulokban levő függvények írják le.

A Bal kattintásra:

- Ha a menüben vagyunk (firstclick bool) akkor a három pályát és a raglistát lehet megnyitni a megfelelő gombra kattintva.
  Pálya gombra kattintva a megfelelő pályát legenetálja a:

void EASY_rajzol(SDL_Renderer *renderer, SDL_Texture *mezok)
(Intermediate vagy hard ha arra kattintottunk, de funkcióban ugyan azt csinálja ez a 3 csak más ablakmératekkel)
Ez a három függvény kirajzolja az ablak szélére a szürke keretet és a pályát a lefordított x * y mezővel.

Mezo** tomb_lefoglal (int x , int y)
Ez a függvény pedig a pálya kiválasztásnál lefoglalja azt az x * y méretű 2 dimenziós tömb helyét.

  A ranglistára kattintva betölt egy másik oldal a 3 pálya legjobb időivel és onnan visszatér a menübe egy másik bal kattintásra.
- Ha már az egyik nehézségi fokozat ki lett választva, akkor a teljesen lefordított pálya egyik mezőjére kattintva lesz legenerálva a tömb, hogy ne lehessen az első kattintásnál felrobbanni. (general bool) És csak ekkor kezdi el számolni az időt is.

void tomb_feltolt(sizex, sizey ,tomb)
A már lefoglalt 2 dimenziós tömböt most feltölti 0 számokkal és a megfelő bool értékekkel.

Az elso bool azt jelöli, hogy első kattintás melyik mezőn van,
A leforditott azt jelöli, hogy az adott mező még lefordított-e, ilyenkor fel is lehet fordítani vagy zászlót tenni rá,
A zaszlos_e pedig azt, hogy van-e zászló a lefordított mezőn, és akkor nem engedi felfordítani azt bal kattintásra.

void tomb_general(sizex, sizey, bomb, tomb, tombx, tomby)
Ez a függvény a tömb elemeibe a megfelelő számú bombát (bomb) legenerál -1 értékkel, persze az első kattintás és szomszédos mezőibe nem lesz bomba direkt, majd végigszalad az összes elemen és megnézi, hogy a szomszédos mezők közül mennyiben van bomba és ennyit az a mező értékének.

Amíg a játéknak nincs vége a következő összetett függvény mindent megold

void rajzol (SDL_Renderer* renderer, SDL_Texture *mezok, int szeles, int magas, int x, int y, int LEVEL, Mezo **tomb)
Az első teendője, hogy a kattintott mezőt felfordítja, majd megnézi, hogy 0 volt-e, ha nem akkor csak felfordított egy számot (vagy bombát), ha viszont 0 volt az a szám, akkor belép a függvény második felébe, ami rekurzívan meghívja ezt a függvényt a szomszédos mezőkre is, és ha azok is nullák, akkor addig, amíg minden nulla melletti szám fel nem fordul.

Azt, hogy melyik mezőre milyen képet rajzoljon azt a **milyen** függvény írja le, ami csak visszatér a megfelelő **Mezokep** struktúra elemével, és azt rajzolja ki az előző függvény.

A milyen függvényt még használom a mainben is feltétel meghatározásra, hogy bomba- e vagy nem a kattintott mező.

- Ha a játékosnak sikerül minden bombára zászlót helyezni vagy egy bombát felfordítani, akkor a képernyő közepére kirajzolja, hogy Nyert-e vagy vesztett, majd arra a gombra kattintva vissza lehet térni a menübe
  (ahol minden bool és változó alaphelyzetbe áll, és a tömböt is felszabadítja minden játék után)

A Jobb kattintásra:

- A kattintott mezőre rajzol egy zászlót, ha le van fordítva az (, különben semmit nem csinál) és elveszi a zászlót, ha zászlóra kattintottunk.

void mezo_rajzol(SDL_Renderer *renderer, SDL_Texture *mezok, Mezokep melyik, int x, int y, int LEVEL)
Ezt a függvényt használja a **rajzol** függvény is, ez a függvény a megfelelő x-y koordinátára rajzol egy képet, amit a melyik változó az meg ami ebben az esetben a Zaszlo vagy a Leforditott.

- Minden zászlót számol és nem engedi, hogy a bombaszámnál több szászlót tegyünk le, ezt a rajzolt_zaszlo változó tárolja fordított értékben, mert minden razolásra kivon egyet a bomba számból és addig enged rajzolni, amíg a rajzolt_zaszlo > 0.
- Emellett megnézi azt is, hogy a zászlót bombára tettük-e és, ha igen a zaszlok számlálót növeli, majd, ha ez eléri a bombák számát, akkor nyert is a játékos.
- Ha nyert a játékos, akkor menti el csak fájlba az időt, de azt is csak, akkor, ha jobb lett, mint az eddigi érték.