Programozói dokumentáció: Aknakereső

A program 3 modulból épül fel. Az egyik almodul (jatekallas.c) tartalmazza a játék menetéért felelős függvényeket, ezek változtatják meg a játékhoz tartozó pályát, ennek a modulnak a .h fájljában található a Mezo struktúra és a Kep enum. A másik almodul (kirajzol.c) a képek és ablakok kirajzolásáért felelős. A harmadik főmodulban pedig a main függvény kap helyet a 3 ablak meghívásával és a többi almodul függvényeinek meghívásával.

A programot a Code:Blocks nevű fejlesztőkörnyezetben hoztam létre, és SDL multimédiás könyvtár segítségével jelenítettem meg grafikusan.

Jatekmenet modul:

Mezo struktúra:

Ez a struktúra tartalmazza egy mezőnek a tulajdonságait, ezekből épül fel a pálya. 4 értéke van:

1)int korul: megmutatja, hogy a mellette lévő szomszédos mező közül hány akna

2)bool akna: ha igaz, az adott mezőn akna van

3)bool fedett: a mezőnél megmutatja, kattintottak-e már rá (fel akarták-e fedni)

4) bool zaszlo: a mezőn van-e zászló elhelyezve

Kep enum:

Ebben határoztam meg, hogy egy játék során milyen mezők lehetnek, akik egy kép alapján nyilvánulnak meg, ezzel segítettem a kirajzolást is egy másik függvényben.

sdl_init:

Ez a függvény az ablakok megnyitásáért felel és a lehetséges SDL hibaüzenetek találhatóak benne.

palyamaker:

Inputként meg kell adni egy kétdimenziós tömbre mutató pointert, egy szélességet, magasságot és az aknák kívánt darabszámát. Feltölti a tömböt üres mező értékekkel, majd egy véletlenszerűen két koordinátát kisorsolva generálja az aknák helyét a megadott „aknadarab”-szor. Ezeknek a környezetében megnöveli a „korul” értéket és visszatér az újonnan feltöltött és játszható kétdimenziós tömbbel.

szabadit:

A kétdimenziós tömb felszabadításáért felel. A játék bezárásával vagy győzelem és vesztés esetén ezt a függvényt hívom meg. Meg kell adni a tömbre mutató pointert és a méretét.

ertekadas:

Ezt a menüben használom, az egyes módokhoz tartozó értékeket veszi fel a szélesség, magasság és aknadarab változóm. Meg kell adni a tömbre mutató pointert a szélességet, magasságot és akna darabot tartalmazó változókat.

balklikk:

Itt a bemenet a tömb pointere, egy x és y koordináta, ami egy mezőt fog jelölni ezen belül, és mivel a pályák négyzetesek, elég a magasságot megadni. Ez egy rekurzív függvény, ennek segítségével lehet felfedni a mezőket. A függvény visszatér, ha:

1) a pályán kívüli mezőhöz jutnánk

2) ha már fel van fedve a mező

3) ha zászló van a mezőn

Ezután, ha továbbmegy, felfedi az adott mezőt és akkor tér vissza, ha:

1) aknát talál

2) ha a *korul* értéke a mezőn nagyobb nullánál

3) ha zászló van a mezőn

Amíg ezek nem teljesülnek, vagyis üres mezőn áll, addig meghívja önmagát a szomszédos mezőkre és így felfedi az egybefüggő üres mezőkből álló „szigeteket".

jobbklikk:

Itt a bemenet a tömb pointere, és szintén egy x és y koordináta, ami egy mezőt fog jelölni ezen belül. A függvény a zászlók lerakásáért fele. Ha az adott mezőn már van zászló, akkor azt „felveszi", a zaszlo értékét false-ra változtatja. Ha az adott mezőn viszont nincs zászló és a mező le van fedve, akkor lerak egyet, a zaslo értékét true-ra írja át.

ellenoriz:

Itt a bemenet a tömb pointere, a tömb magassága és szélessége, és az aknadarab. Ez vizsgálja meg minden lépés után, hogy a játékos megnyerte-e vagy vesztett-e a játékban. Megszámolja, hogy hány zászlót helyeztek le jól (vagyis amin tényleg akna van) és ha ennek és a jól felfedett mezők számának összege megegyezik a játék méretével, akkor nyert a játékos és 2-vel tér vissza. Ha felfedett aknás mezőt talál, akkor azonnal visszatér 1-gyel. Ha pedig semelyik állás nem igaz ezek közül, akkor 0-val tér vissza, ilyenkor a játék folytatódik.

Kirajzol modul:

menurajzol:

Az első előugró ablakban kirajzolja a menüt. Rajta a 3 módhoz tartozó kiírással és hozzájuk egy-egy

kep_rajzol:

Ez egy segédfüggvény, melyben a palyarajzol függvényben egyszerűsítettem le a képek beszúrását. Inputja a renderer, egy SDL_Texture, ami nálam a mezőket tartalmazó kép, egy x és egy y koordináta, hogy az ablakban hova rajzolja ki a képet, és egy méret, hogy ez mekkora legyen.

palyarajzol:

Ennek az inputja az előző kep_rajzol függvény inputjai mellett a kétdimenziós tömbre mutató pointer is. Minden kattintásnál végigmegy a tömb minden elemén és kirajzolja az ahhoz tartozó megfelelő Kep enumot.

nyertel:

Ez a harmadik ablak kinézetéért felel és ebben szerepel a file kezelés. Input a tömb pointere, a pálya magassága, és egy int érték, ami a játék ideje volt. A magasság alapján megvizsgálja melyik módban játszott a játékos, és megnyitja a megfelelő filet. Kiolvassa belőle az eddigi rekordokat (3 értéket), berakja egy tömbbe és ha a most lejátszott játék hamarabb véget ért, akkor azt írja a file-ba a megfelelő helyre. Ezután kiírja az aktuális rekordokat („TOP1:", „TOP2:", „TOP3:") és az aktuális játék idejét („YOURS:").

vesztettel:

Ez vesztés esetén jelenik meg, ilyekor nem vizsgálja, hogy rekord-e az idő, nincs filekezelés se.

Főmodul:

Ebben a modulban van a main függvény és itt definiálom a változókat, melyeket a függvényeknél használni fogok:

- a magasságot (mag)
- a szélességet (szel)
- az akna darabot (aknadb)
- egy allas integert, ami a játék aktuális állását jelzi nekünk
- 2 time_t értéket, amivel a játék idejét mérem
- egy jatszik bool-t, mellyel azt nézem kilép-e az ablakokból módválasztás és győzelem/vesztés nélkül
- és még 3 bool, amivel az egyes ablakok bezárását jelzem

Ezután az egymás után megnyitott 3 ablakban meghívom a megfelelő függvényeket, és a felszabadításokat.

Készítette Szalka Panka