NHF Terv - Nagy István – Aknakereső

Előszó

Ez a dokumentum a Programozás alapjai 2 tárgy követelményéhez tartozó Nagy házi feladat második, pontosítottabb dokumentuma. Számomra a kihirdetéstől számított rövid időközök miatt is, illetve az eddigi megszokásom, miszerint a függvényeket (fejléc szinten) úgy írom meg, ahogy szükségem lesz rájuk máshol, nem pedig előre az összeset, így lényegesen nagyobb fejfájást okozott, mint amennyire az a ZH-k közepette örömteli lett volna. Szép dokumentációkat láttam eddig a doxygen megnevezésével, de már második napja nem akar összeállni a gépemen a teljes történet, html-ben uml-t nem mutat, pdf-ig el se jut… ez még a jövő zenéje, hogy ezt a részét megértsem. Mindenesetre igyekeztem a fejléceket összegyűjteni, de nem zárom ki a későbbi kiegészítés, változás (pl. visszatérési érték) lehetőségét.

Program tervezett funkcionalitásai

A program induláskor bekéri a játékhoz szükséges adatokat (játékos neve, pálya mérete, nehézsége). Hiányuk esetén alapértelmezett (statikus) értékekkel indul el.

Létre jön a játék, ami egy aknakereső táblájaként fog megjelenni a felhasználó számára, plusz egy-két információ a játék állásáról (pl. játékidő, zászlók száma, stb.), ebben a felhasználó felfedezhet cellákat, megjelölhet még nem felfedezett cellákat zászlókkal, leveheti a zászlókat. Legalább egy akna szomszéddal, valamint ezek számával megegyező zászlós cella szomszéd esetén a már korábban felfedezett cellára is lehet kattintani, ami ilyenkor az összes szomszédos nem felfedett, nem zászlós szomszédot felfedezi, akár jó volt a zászlózás, akár nem.

A tábla tartalma, valamint a kurzor jelölésén felül statisztikai adatok jelennek meg a kijelzőn (pl. játékidő, zászlók száma, stb.)

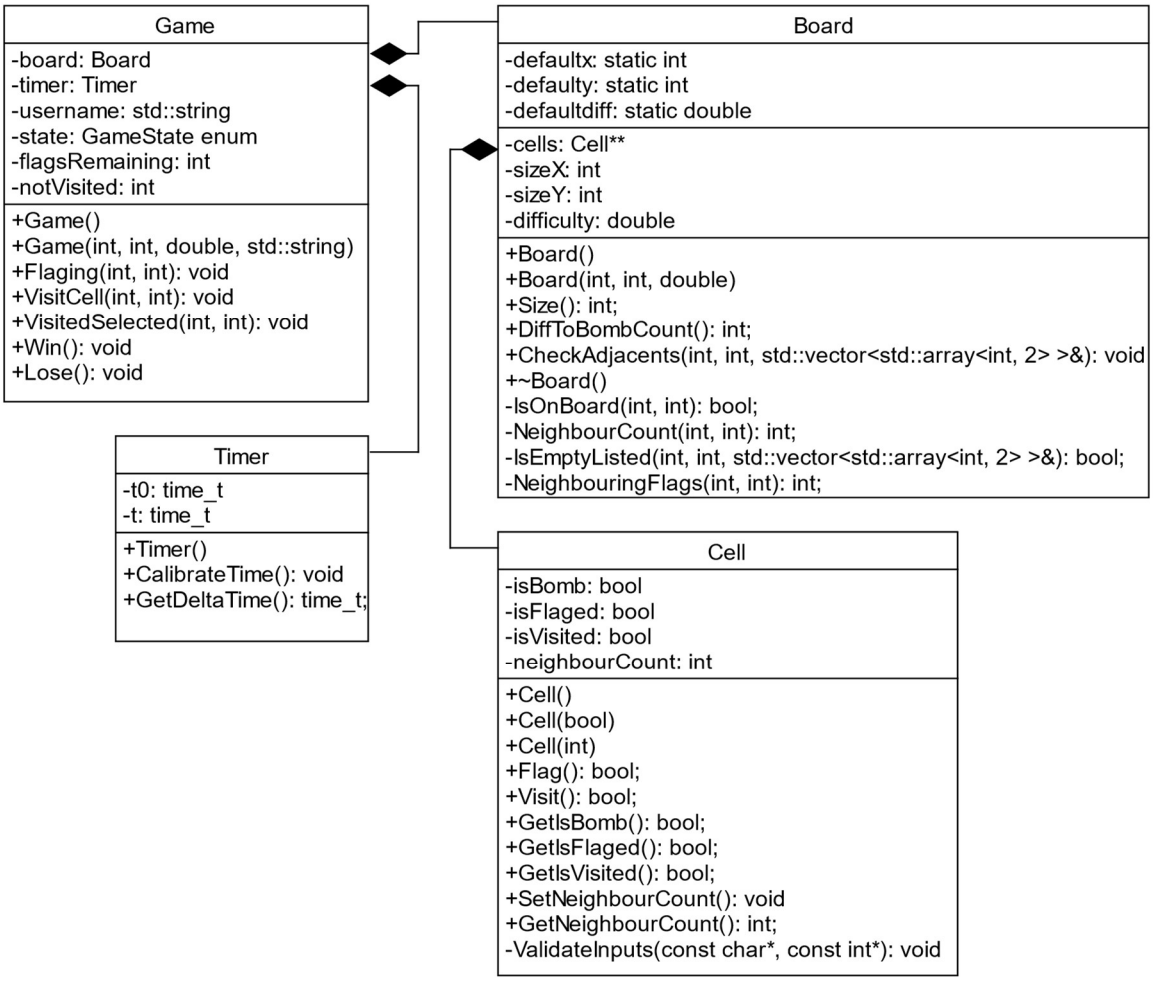
A programból való kilépéskor a játék állapota egy megfelelő fájlba kerül mentésre. Amennyiben ez egy befejezett játék volt, akkor egy, a már befejezett játékokat rögzítő fájlba fűződik a végéhez, ha viszont egy még folyamatban lévő játékról van szó, akkor azt egy külön fájlba írja ki. A következő elinduláskor, ha volt ilyen játékközi mentés, a program megkérdezi a játékos nevet, és ha a játékközi mentésekben van olyan, aminek ugyanaz a játékosa, akkor a program felajánlja, hogy az előzőt töltse be. Nemleges válasz esetén a régi játékközi mentés törlésre kerül.

Az aknakereső játékokban nem megszokott, de lehetőség lesz az események visszavonására. Ez valószínűleg egy ki-be kapcsolható funkció lesz, mivel a játék megnyerése, vagy elvesztése teljesen automatikusan befejezik a játékot, míg egy ilyen funkció esetén pont ez a lényeg, hogy egy tévesen felfedezett aknát ki tudjunk például kerülni. Nem egy jó példa, mivel ilyen funkció aknakeresőkben nem szokott előfordulni, jelenleg is csak programozás technikai kihívás, valamint játéktechnikai oktatásként látom létjogosultságát.

A kijelzőn tervezetten csak akkor frissül bármi is, ha a felhasználó valamilyen érvényes inputot ad. Érvénytelen input esetén a program a következő érvényes inputig nem csinál semmit.

Forráskód struktúra (Core)

Az eddig alkotott osztályok változói szinte[1] véglegesek, a tagfüggvények viszont folyamatosan változnak. Ez részben a rossz tervezés, részben a menetközben érkező újabb funkciók megteremtésének lehetővé tétele okozza. Már maga a kód is doxygen szintaxissal van dokumentálva, viszont ez a doxygen hiányos ismerete miatt még nem sikerült pdf-esített formátumban prezentálni határidőre. A jelenlegi állapotokat a következő UML ábrán látható.



A részletes függvényleírások a függvények deklarálása előtt találhatóak a forráskódban, melyekből a doxygen már jelenleg is tud dokumentációt generálni[2]. Ezeket a projekt github-oldalára csatolom. Egy rövid leírás az osztályok szintjén:

- Cell: A pályát alkotó cellák, azok alap tulajdonságai, illetve azok a függvények, amik nem igénylik más cellák ismeretét.
- Board: Maga a pálya. A pályát a befoglaló méretei, illetve a rajta található aknák aránya írja le, meg természetesen az azt kitöltő cellák.
- Timer: Egy egyszerüsített időmérő, ami eltárolja a kezdő időpontot. Amikor lekérdezés történik, egyszerűen csak kivonja a jelenlegi időből a kezdőt, ezzel „megúszva” a szálak használatát

[1] Az összes privát jelenleg, de még nem látom át, hogy melyikeket kéne publikussá tenni, melyikeket kéne csak get()/set() függvénnyel/függvényekkel ellátni.
[2] Az előszóban már említettek miatt csak a html alapú, a pdf-esítésen még dolgozok.

- Game: A játékot leíró paramétereket, adatokat összesíti, az előbb említett osztályokat összefogja.

Megjelenítés

A megjelenítés a játék mechanikához (core) képest külön működik, a teszteléshez eddig nem is kellett hasznosítani, így ennek implementálása még el sem kezdődött. Ez a saját paraméterein kívül csak a Game class egy objektumát fogja megkapni, és azt fogja valamilyen formázással megjeleníteni egy parancssoros környezetben.

Github

A projektet githubon tárolva szikronizálom több eszközöm között, ezen a most feltöltésre alkalmatlan állapotú dokumentáció is megtalálható.

Link: https://github.com/VasutasIsti/AknakeresoCpp