Programozó Dokumentáció

A program jelenlegi változata félkész jelenlegi formájában már javarészt játszható, de egyelőre még nem képes érzékelni az egér kattintásokat, csak billentyűzettel működik(WASD vagy irány billentyűk, SPACE, R és ESC). Ezenkívül hiányzik belőle a játékosok nyerési rátáját számontartó rendszer.

Futtatás és fordítás

A program a „**gcc -Wall -Werror Solitaire.c cards.c gamestate.c input.c logging.c rendering.c -o Solitaire.exe -lncurses -DNCURSES\_STATIC**” paranccsal fordítható le. Fontos a hogy a projecthez használt ncurses könyvtár a MINGW-w64 csomagot használja, ennek a telepítésének a leírása az alábbi linken érhető el: <https://stackoverflow.com/a/30071634>  
A könyvtár útvonala a MINGW install esetében az ncurses/ncurses.h lesz, ahol a program a könyvtárat használja ezen az útvonalon éri el. Ez fontos mivel a program más install esetében nem biztos, hogy képes lesz megtalálni a könyvtárat.

Modulok

A programban használt modulokat az alábbi gráfon lehet megjeleníteni, a nyilak jelzik, hogy melyik modul include-ol mely másik modul(oka)t. Az egyes modulokat és az azokban használt függvényeket részletesebben is kifejtem használt függvényeket.

A network of white lines and dots

Description automatically generated

Layout.h:

A layout.h egy konfiguráció fájl, benne találhatóak a játéktér és a kártyák méretei és a kártyák távolsága egymástól

Utils.h

A utils.h egy fájl mely segédfüggvényeket szolgáltat a többi modulnak. Jelenleg kettő makró található benne a MAX(a, b) ÉS A MIN(a, b)

Cards.c és cards.h

A cards modul implementálja a játék alapjául szolgáló kártyákhoz köthető struktúrákat és függvényeket.

struct Card: egy darab kártyalap adatait tárolja

Adatai:

* suit: Szín(treff, szív, pikk, káró)
* rank: a kártya rangja (ász, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jumbó, Dáma, Király)

Függvényei:

* CardSuit(Card card): visszaadja a kártya színét jelző karaktert
* CardRank(Card card): visszadja a kártya rangját jelző karaktert(A 10-es esetében 0-át vissza mivel a 10 nem fér bele egy karakterbe, a 0 tekintendő tizesnek a vizuális megjelenítésnél)
* CardColor(Card card): visszaadja a kártya színéhez tartozó indexet, ha nem a valid tartományon belül van akkor az alapértelmezett szövegszínhez tartozó indexxel tér vissza
* ShuffleCards(Card\* cards, int length): véletlenszerűen megkever egy length hosszúságú kártyákból álló listát, ezt a függvényt a gamestate használja hogy a kártyák osztásakor megkeverje a kártyákat

struct CardPile: egy kártyakupac az asztalon, melyhez kártyákat lehet hozzáadni és belőle eltávolítani, a kártyáknak foglalt memória dinamikusan kezelt, ha megtelik a program akkor újra foglal kétszer annyi memóriát és ha kapacitás csak fele van megtöltve akkor felére csökkenti a terület méretét

Adatai:

* cards: a kártyákat tároló dinamikusan foglalt listára mutató pointer
* size: a pakliban lévő kártyák száma(NEM FELTÉTLEN A LISTA MÉRETE)
* capacity: a lista mérete, ha size kisebb, mint a fele vagy nagyobb mint a kapacitás akkor újra kell foglalni
* maxDisplay: a kupac megjelenítésére vonatkozó tulajdonság, mindig a maximum megjeleníthető számú felső kártyája lesz kirajzolva, ha kevesebb kártya van akkor kevesebb
* uncovered: egy pakliban a uncovered indexű kártyáig lesznek letakarva kirajzolva a kártyák(index 0-tól kezdődik az uncovered még nincs letakarva)

Függvényei:

* CardPile CreateCardPile(): léterhoz Egy új kupacot, foglal neki egy kártyára elengedő terület és a tulajdonságait az alapértelemezettre állítja
* ResizePile(CardPile\* pile, int newCapacity): Vesz egy meglévő paklira mutató pointert és egy newcapacity méretű területet lefoglal ahova a kupac tartalmát átmásolja, ha az új kapacitás kisebb mint a régi akkor csak annyit másol amennyi ráfér
* AddCard(CardPile\* pile, Card): Egy létező kupachoz ad hozzá egy újabb kártyát, amennyiben nem fér rá elegendő kártya, újraméretezi a paklit