Programozás alapjai 2 - házifeladat

készítette: Villei Bálint (X0TQK4)

Mealy modell

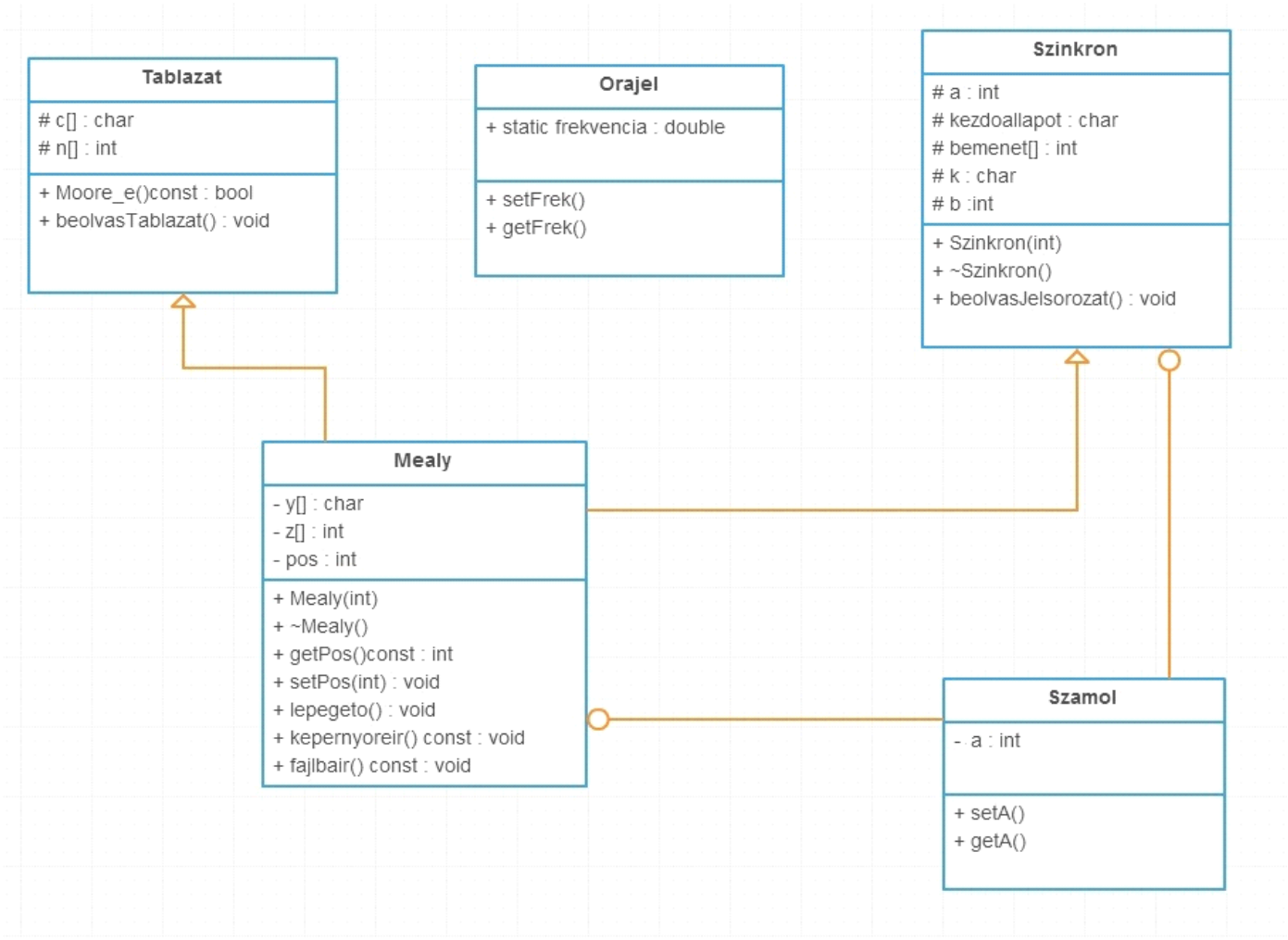
specifikáció

Program feladata

A program lekezel egy txt formátumú fájlban megadott állapottáblát és hozzá tartozó kezdeti állapotot és jelsorozatot. A program leteszteli hogy megfelelő lett minden megadva (nem tartalmaz funkcionális hazárdot, nem Moore esetről van szó...) és ha igen, akkor jelsorozat alapján megfelelően végig lépked a táblázaton és a kimeneti állapotokat és értékeket kiírja a képernyőre, továbbá egy "kimenet.txt" nevű fájlt is létrehoz, amibe szintén beírja az eredményt.

A hálózat tartalmaz két jól működő tesztfájlt és a hibákra külön tesztfájlokat.

UML



Kifejtés

Orajel: Órajel frekvenciáját tároló osztály (végül nem került használatra de hát egy szinkron hálózat nem szinkron hálózat órajel nélkül :))

tagváltozók:

- frekvencia: Órajel frekvencia

tagfüggvények:

- setFrek: Frekvencia beállításhoz
- getFrek: Frekvencia lekérdezéshez

Szamol: Megszámolja a bemeneti jelek számát és eltárolja egy 'a' változóban

tagváltozók:

- a: bemeneti jelek száma

tagfüggvények:

- setA: 'a' beállításához
- getA: 'a' lekérdezéséhez

Tablazat: txt fájlban megadott állapottáblát beolvassa két tömbbe

tagváltozók:

- c[]: táblázat állapotait tároló tömb
- n[]: táblázat értékeit tartalmazó tömb

tagfüggvények:

- Moore_e: Moore működést tesztelő függvény (Moore a hálózat ha soraiban azonos értékek találhatók)
- beolvasTablazat: táblázatot beolvasó függvény

Szinkron: txt fájlban megadott kezdőállapotot és jelsorozatot beolvassa két tömbbe

tagváltozók:

- a: bemeneti jelek száma
- kezdoallapot: kiindulási állapot a táblázatban (fájlba meg kell adni)
- bemenet[]: bemeneti jelek tömbje
- k: beolvasáshoz szükséges változó, későbbi értéke nem szükséges számunkra
- b: beolvasáshoz szükséges változó, későbbi értéke nem szükséges számunkra

tagfüggvények:

- Szinkron: konstruktor
- ~Szinkron: destruktor
- beolvasJelsorozat: kezdőallapotot beolvassa a kezdoallapot nevű változóba és a bemeneti jelsorozatot beolvassa a bemenet nevű táblázatba

Mealy:

tagváltozók:

- y[]: állapotokat tároló tömb, ezt fogjuk végül kiírni a képernyőre és fájlba is
- z[]: kimemeneti értékeket tároló tömb, ezt fogjuk végül kiírni a képernyőre és fájlba is
- pos: táblázaton belüli pozíciót tároló változó

tagfüggvények:

- Mealy: konstruktor, ami dinamikusan foglal területet y és z tömbnek a bemeneti jelek számának függvényében
- ~Mealy: destruktor
- getPos: get függvény pozíció lekérdezéséhez
- setPos: set függvény pozíció beállításához
- feltolt: kezdeti pozíció beállítása és kezdeti állapot és érték beírása y és z tömbökbe
- lepegeto: végig lépked a táblázaton(tömbön) és eltárolja az állapotokat és értékeket az y és z tömbökben
- kepernyoreir: eredményt (y és z tömbök tartalmát) képernyőre kiíró függvény
- fajlbair: "kimenet.txt"-be kiírja az eredményt (y és z tömbök tartalmát)

Kitételek:

Mivel a feladat szinkron Mealy hálózat állapottáblájának végig járásáról szólt egy beadott bemeneti jelsorozat alapján, a program nem oldja meg a Moore eseteket.

Ha a hálózatban található funkcionális hazárd, vagyis egyszerre két bemenet változik, akkor szintén hibát dobunk.

Továbbá ha a kezdeti állapotot rosszul adtuk meg, tehát nem A,B,C,D egyike, akkor hibát dobunk és a program leáll.

Végül hogyha nemlétező fájlt adtunk meg vagy például rossz kiterjesztéssel adtuk meg, szintén hibát dob és szól nekünk hogy nem sikerült megnyitni a fájlt

Használati utasítás:

(A hálózat tartalmaz két jól működő tesztfájlt és a hibákra külön tesztfájlokat.)

A bevitel szöveges(txt) fájlból történik. Először az állapottáblát adja meg a következő formában:

X y X y X y X y

X y X y X y X y

X y X y X y X y

X y X y X y X y

Z aa aa aa…

Ahol X A,B,C,D egyike, ez jelöli az állapotokat és y a bináris számrendszer 0, vagy 1 értéke, ez jelöli az állapotokhoz tartozó értékeket. Z szintén A,B,C,D egyike és aa szintén a 0 és 1-es számokból álló egy kombináció (00,01,10,11).

Az X és y segítségével adhatja meg az állapottáblát, ami alapján a feladat elvégzése megtörténik.

A Z és az első aa pedig a kezdőállapotot adhatja meg. A további aa-k pedig a bejövő jelsorozatot adják meg, amikből tetszőleges számú lehet.

Példa:

A 0 C 0 D 0 B 1

D 0 B 1 C 1 B 0

C 1 B 0 C 0 D 0

D 1 B 1 A 1 B 0

A 00 01 11 10 11 10 00 01

(Az első négy sor a táblázat, és az utolsó sor pedig kezdőállapot és a jelsorozat)