ÜTEMEZŐ C++ NYELVBEN

Patka Zsolt-András
-2018-

TARTALOM

1. Feladat

```
Folyamat ütemező algoritmusok megvalósítása objektum orientált személetben.
Algoritmusok: FCFS (First come, first served), RR (Round Robin).
Bővítési lehetőség: SJF (Shortest Job First), SRTF (Shortest Remaining Time
First).

Példa bemenet: Állományból
RR 2                Ütemező algoritmus neve, time slice (RR ütemező esetén)
4                                        Ütemezésre váró folyamatok száma
0 5          Első folyamat: 0 időpillanatban kezdődik, 5 CPU löketet igényel
4 5       Második folyamat: 4 időpillanatban kezdődik, 5 CPU löketet igényel
5 3      Harmadik folyamat: 5 időpillanatban kezdődik, 3 CPU löketet igényel
6 1      Negyedik folyamat: 6 időpillanatban kezdődik, 1 CPU löketet igényel

Példa kimenet: Gantt diagram formában a standard kimenetre.
```

Folyamatok	Kezdet, CPU löket														
A	0, 5	x	x			x	x						x		
B	4, 5			x	x			x	x					x	
C	5, 3									x	x				x
D	6, 1											x			
Lépés		1	2	3	4	5	6	7	8	9	10	11	12	13	14

2. Pontosított feladatspecifikáció

A feladat az ütemező algoritmusok implementálása. A felhasználó megadhat a bemeneti állományban akár több ütemezést is. Ezekre a kimenetet a program a szabványos kimenetre fogja egymás után kiírni.
A program ellenőrzi a bemenetet, ha ez nem megfelelő, akkor hibaüzenettel leáll.
A folyamatok maximális száma és a maximálisan megadható CPU-löket is meg lesz határozva. Ha ezeknek nem megfelelő bemenetet kap, hibaüzenettel leáll.
Tesztelés: különböző bemenetekre fogom tesztelni a programot, amelyek között lesz helytelen bemenet is. Az algoritmusok helyességét külön fogom tesztelni.

3. Terv

A feladat több osztály és egy tesztprogram megtervezését igényeli

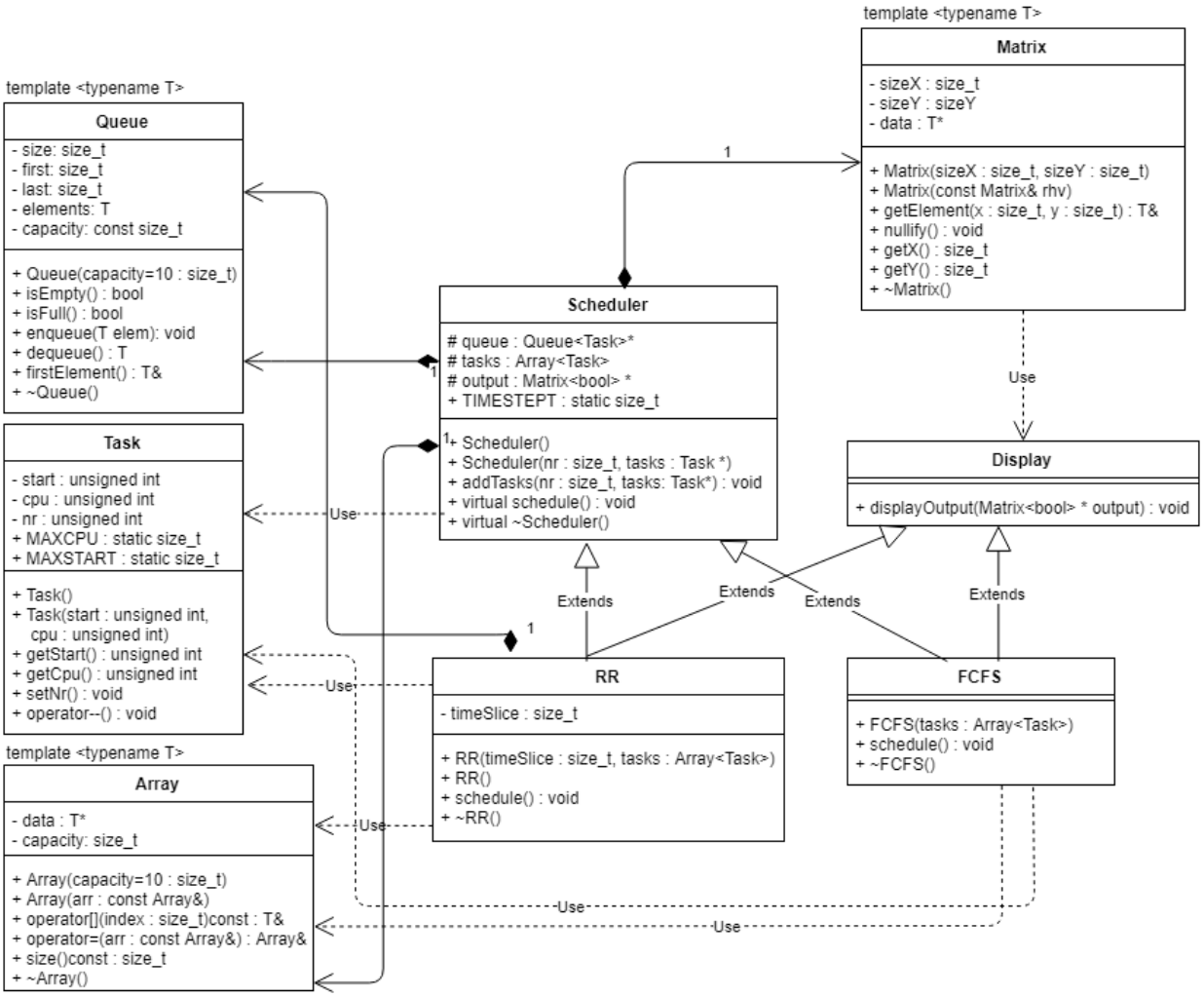
3.1. Objektum terv (UML)

A feladat megvalósításához három generikus tárolóra van szükségem:

- Queue: FIFO adatszerkezet, az ütemező algoritmusokhoz szükséges
- Array: Egyszerű generikus tömb, Task objektumok tárolásához használom
- Matrix: Generikus kétdimenziós tömb sorfolytonosan ábrázolva a memóriában.

A Task osztályban tárolom egy folyamat kezdeti idejét és cpu-löketét.

A Scheduler osztály minden ütemező osztálynak az őse. Itt állítom be az ütemezésre váró folyamatokat
A Display osztály szintén minden ütemező osztálynak az őse, ez jeleníti meg a kimenetet. Azért döntöttem külön osztály mellett, mert szükség esetén ezt könnyen le lehet cserélni más osztályra (amely szintén egy Matrix<bool> objektummal dolgozik) és ez által a kimenet megjelenítésén könnyen lehet változtatni.
Az FCFS osztály a First Come First Served algoritmust implementálja.
Az RR osztály a Round Robin algoritmust implementálja. Ennek az osztálynak van két külön adattagja: time slice és egy Queue típusú. Ezek szükségesek az adott algoritmushoz.



3.2. Algoritmusok

3.2.1. FCFS (First Come First Served)

A folyamatokat érkezésük sorrendjében ütemezi. Amikor megérkezik egy folyamat, bekerül egy várakozási sorba. Ebből a várakozási sorból szerre kerülnek ütemezésre a folyamatok.

3.2.2. RR (Round Robin)

A folyamatokat csak egy bizonyos ideig hagyja futni, utána várakoztatja őket és a következő folyamatot futtatja.

3.2.3. Queue (Várakozási sor)

Körkörös tömbbel implementálva. Ha a elérünk a tömb végéhez, de a várakozási sorban még van hely, akkor a tömb elejére rakunk be elemeket.

3.2.4. Matrix

Kétdimenziós tömböt egydimenziós tömbbel implementálva. Kívülről ugyanúgy lehet indexelni, mint egy hagyományos kétdimenziós tömböt.
Indexek konverziója: t[i][j] -> t[i*oszlopok_száma + j]

3.2.5. Tesztprogram algoritmusai

A tesztprogram egy állományból beolvassa az ütemező algoritmus típusát és az ütemezésre váró folyamatokat. Lefuttatja a megfelelő algoritmust az adott folyamatokkal és az eredményt kiírja a standard kimenetre.