

Laborator 4

Deadline: saptamana 9

Obiectiv:

- Intelegerea/aprofundarea sablonului “producator-consumator”
- Intelegerea/aprofundarea sincronizarii conditionale
- Intelegerea/aprofundarea excluderii mutuale (granularitatea sectiunilor scrice)

Se considera n polinoame reprezentate prin lista de monoame (reprezentare: lista inlantuita ordonata dupa exponentii monoamele).

Se cere adunarea polinoamelor folosind o implementare multithreading (p threaduri).

Polinoamele se citesc din fisiere – cate un fisier pentru fiecare polinom:

- un fisier contine informatii de tip (coeficient, exponent) pentru fiecare monom al unui polinom,

(**conditie: fisierele nu contine monoame cu coeficient egal cu 0**)

Metoda:

1. Se creeaza o lista inlantuita - L corespunzatoare unui polinom nul.
 2. Primul thread citeste cate un monom si il adauga intr-o structura de date de tip coada.
(**conditie – pentru structura de tip coada NU se admite folosirea unei structuri de date pentru care partea de sincronizare este deja implementata!!!**)
 3. Celelalte threaduri preiau cate un monom din coada si il aduna la polinomul reprezentat in lista L .
- ➔ Se continua operatiile 1., 2., 3. pana cand toate monoamele, din toate fisierele, sunt adunate la lista L .
4. Rezultatul obtinut in lista L se scrie intr-un fisier rezultat
(**conditie: fisierul nu contine monoame cu coefficient egal cu 0**)

Sincronizare la nivel de lista!!!

Limbaj: la alegere intre Java si C++

Analiza timpului de executie pentru urmatoarele cazuri:

- 1) Rezolvare secventiala
- 2) 10 polinoame fiecare cu gradul maxim 1000 si cu maxim 50 monoame
 - a. $p = 4, 6, 8$
 - b. secvential
- 3) 5 polinoame fiecare cu gradul maxim 10000 si cu maxim 100 monoame
 - a. $p = 4, 6, 8$

b. secvential

Fisierele input se creeaza prin generare de numere aleatoare!