Tema laborator 4

* Analiza Cerintelor:

Se considera n polinoame reprezentate prin lista de monoame.

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

Consideratii generale:

- reprezentarea unui polinom in memorie: lista inlantuita (1 nod=1monom) ordonata dupa exponentii monoamelor cu urmatorul INVARIANT (predicat adevarat la orice moment al executiei) de reprezentare:

-monoamele sunt ordonate dupa exponenti

-nu se pasteaza in lista monoame cu coeficient 0;

- nu exista doua noduri (monoame) cu acelasi exponent

- 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,

- fisierele input se creeaza prin generare de numere aleatoare.

Rezolvare:

Se porneste prin crearea unei liste inlantuita - L corespunzatoare unui polinom nul. In final aceasta lista va continue polinomul rezultat.

Metoda A) Implementare secventiala

· Se citeste pe rand din fiecare fisier cate un monom si se adauga in lista rezultat -L (atentie – invariantul trebuie sa ramana adevarat dupa fiecare adaugare de monom).

Metoda B) Implementare paralela – p threaduri

1. 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!!!)

2. Celelalte threaduri preiau cate un monom din coada si il aduna la polinomul reprezentat in lista L.

è Se continua operatiile 1., 2. pana cand toate monoamele, din toate fisierele, sunt adunate la lista L.

3. Primul thread scrie rezultatul obtinut in lista L intr-un fisier rezultat

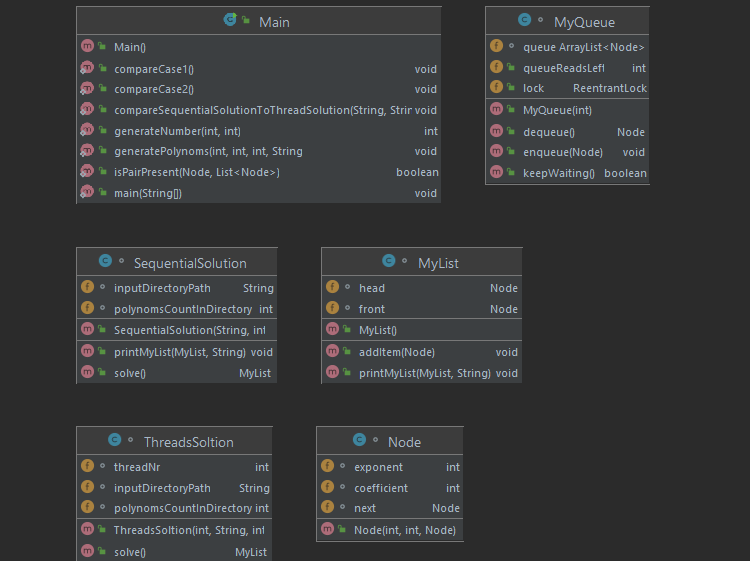
(conditie: fisierul nu contine monoame cu coefficient egal cu 0)

* Proiectare:

Pentru aceasta tema de laborator am ales sa lucrez in Java;

**Structurile de date folosite:**List(java) ,Linked List(implementata de mine cu varianta sincronizata), Queue(implementata de mine in varianta sincronizata)

**Clase:**

****

* **Detalii de implemenare:**

Pentru varianta **secventiala:**

Se citesc monoamele din fisierele in care s-au generat polinoamele.

In fisier un polinom este reprezentat astfe: fiecare linie contine exponentul si coeficientul exponentului curent.

In momentul citirii se si adauga direct in Lista Inlantuita implementata folosinf functia addItem(Node node) din clasa MyList

La final,dupa ce toate elementele au fost adaugate in lista se apeleaza metoda din clasa MyList printMyList(MyList ,String) pentru a afisa rezultatele din lista inlantuita.

Pentru varianta cu **thread-uri:**

Avem un producator si nrThreaduri-1 consumatori;

Producatorul citeste monamele din fisierele polinoamelor si le adauga in coada sincronizata implementata de mine,pentru operatiile efectuate asupra cozii am adnotat acele functii cu key-word-ul s***ynchronized***,thread-ului care este producator i se atribuie o prioritate mai mare decat celelalte folosinf functia setPriority().

Se calculeaza numarul de monoame care trebuie sa fie citite in total din fisiere pentru a sti cand sa se termine operatia consumatorilor care asteapta ca noi elemente sa fie adaugate in coada.

Consumatorii preiau elemente din coada si le adauga in Lista Inlantuita.

Cat timp nu s-a atins numarul de monoame care trebuia citit aceste thread-uri(consumatori) vor tot citi din coada.

La final se asteapta terminarea tuturor thread-urilor si se afiseaza rezultatul intr-un fisier.

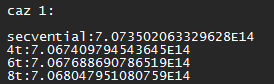
* Rezultate:

Caz 1:

10 polinoame fiecare cu gradul maxim 1000 si cu maxim 50 monoame

a. p = 4, 6, 8

b. secvential



Caz 2:

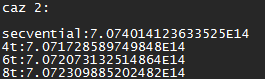
5 polinoame fiecare cu gradul maxim 10000 si cu maxim 100 monoame

secvential:7.074014123633525E14

4t:7.071728589749848E14

6t:7.072073132514864E14

8t:7.072309885202482E14



* Analiza : raport Tsecvential/Tparalel



