**Laborator 5**

Aceeasi problema ca si la Laborator 4 dar cu urmatoarele modificari:

* Dintre cele p threaduri p1 sunt cititoare si p2 sunt de tip ‘worker’ (preiau din coada si adauga in lista rezultat) – p=p1+p2
* Coada in care se adauga monoamele citite are o capacitate maxima = MAX
* Implementarea pentru producator consumator (writer/reader) trebuie sa foloseasca mecanisme de tip wait notify (nu busy waiting)
* **Sincronizarea operatiile cu lista inlantuita se fac la nivel de nod nu pentru intreaga lista!**
  + Sincronizarile se vor face folosind variabile de tip Lock asociate cu fiecare nod.
  + Daca rezulta un nod cu coeficient nul acest nod se va sterge din lista!

Observatie: Este obligatoriu sa se foloseasca sablonul ‘producator-consumator’ pentru legatura dintre threadurile ‘reader’ si cele de tip ‘worker’ --- wait|notify.

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 si MAX=20
   1. p = 4, 6, 8 ; si p1=2 , p1=3
   2. secvential ; si p1=2 , p1=3
3. 5 polinoame fiecare cu gradul maxim 10000 si cu maxim 100 monoame si MAX=30
   1. p = 4, 6, 8 ; si p1=2 , p1=3
   2. secvential ; si p1=2 , p1=3

Analiza comparativa a performantei –Lab 4) versus Lab 5)