|  |  |  |
| --- | --- | --- |
| Nr Readers | Nr Workers | Timp executie |
| sec | sec | 66.1 |
| 4 | 2 | 65.9 |
| 4 | 4 | 66.5 |
| 4 | 12 | 66.6 |

***Timpi***

Comparativ cu laboratorul trecut, timpii cresc semnificativ datorita limitarii cozii si nevoii de a folosi 2 variabile conditionale. In acelasi timp, fiind o implementare care mentine lista simplu inlantuita sortata mereu, lucrul acesta se poate observa ma ales in cazul secvential, care se foloseste doar de lista, nu si de coada, datorita faptului ca se parcurge lista de 2 ori.

***Coada***

Folosim 2 variabile conditionale, una pentru operatia de push, ca sa ne asiguram ca nu crestem peste limita impusa, iar cealalta pentru operatie de pop, ca sa fim siguri ca avem elemente in coada. Pentru cazul in care se poate bloca coada din cauza faptului ca nu mai sunt elemente in lista si nu mai sunt readers, cand decrementam un reader notificam celelalte thread-uri ca sa reverifice conditia ca fie sa existe elemente, fie ca nu mai exista readers.

***Lista***

Implementarea este similara cu cea din laboratorul trecut, fiind adaugat un mutex pentru fiecare nod, iar atunci cand se parcurge lista pentru a adauag un nou nod, blocam 2 cate 2 noduri. Am adaugat si un mutex doar pentru cazul listei cu cei care au incercat sa triseze.

***Implementare***

Un worker este blocat intr-un while pana cand nu ma exista readers sau date. Acesta tot incearca sa dea pop, iar daca primeste o valoare invalida, nu va adauga in lista valoarea. In cazul in care sunt mai multi readers, facem o impartire a fisierelor catre numarul acestora, fiecare primind numberOfCountries \* numberOfProblems / numberOfReaders. Restul care ramane este distribuit uniform la primele threaduri care citesc. Este folosita o bariera pentru a ne asigura ca primul reader este cel care scrie in fisier.