**Documentatie – P1**

Run:

* Conexiunea dintre un client si server se face prin Sockets.
* Serverul asteapta noi conexiuni pe socket. La conectarea unui client se creeaza un nou ClientHandler pentru a se ocupa de requesturile acelui client. Acesta este un task executat de un fixedThreadPool pentru a impiedica crearea unui numar foarte mare de threaduri.
* Dupa conectarea prin Socket, clientul poate face una din urmatoarele requesturi in mod asincron:
  + START – care include id-ul tarii pentru care clientul va trimite date. ClientHandlerul raspunde cu READY atunci cand este pregatit sa proceseze datele trimise de client.
  + PROCESS – in cadrul acestui request se trimit calupuri de cate 20 de entry-uri. Inainte de fiecare nou request se asteapta delta x secunde. La primirea unui astfel de request, handlerul va crea un future si va apela o metoda din cadrul serviciului de procesare a clasamentului in care se creeaza un task de procesare (se adauga id-ul tarii si se adauga in queue) executat de threadpool
  + PARTIAL – in momentul in care un client termina de procesat toate datele asociate unei probleme se trimite un astfel de request. La primirea acestui request, se calculeaza clasamentul cu datele procesate pana acum. In cazul in care acest clasament a fost deja calculat anterior intr-un interval de timp mai mic decat delta t, acesta nu se mai recalculeaza ci se intoarce ultimul rezultat
  + FINAL – dupa ce clientul a trimis toate datele corespunzatoare tarii lui, se trimite acest request catre server. In cazul in care acest clasamentele finale nu sunt inca disponibile, acesta intra intr-o stare de asteptare si va fi notificat de catre server cand datele finale vor fi calculate (await pe variabile conditionale in functie de cati clienti au terminat). In final, clientul va primi atat clasamentul pe tari, cat si cel pe concurenti.
  + STOP – dupa primirea rezultatelor finale, se notifica inchiderea conexiunii cu serverul.
* Requesturile se executa in mod asincron, iar raspunsurile se trimit in acelasi mod, in momentul in care Futureul creat se completeaza cu succes, astfel permitand procesarea altor requesturi in acel timp.
* Dupa conectarea numarului maxim de clienti, serverul va astepta sa primeasca requesturile de FINAL de la toti acestia, dupa care va fi notificat sa inceapa calcularea clasamentelor finale, trimiterea raspunsurilor pentru aceste requesturi tuturor in mod asincron si scrierea in fiesier a rezultatelor. Totodata, acesta reseteaza serviciul pentru a putea deservi noi clienti.
* Procesarea taskurilor de enqueue a entryurilor se face in exactitate cu laboratorul trecut (threadpool pt executia taskurilor, enqueue intr-un queue sincronizat si procesarea elementelor din queue de catre threadurile workeri).

**Testare:**

**Testare Java:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Delta T | P\_R | P\_W | Delta X | Time |
| 1 | 4 | 4 | 1 | 41373.321 |
| 2 | 4 | 4 | 41474.787 |
| 4 | 4 | 4 | 41411.081 |
| 1 | 2 | 2 | 49343.154 |
| 1 | 4 | 2 | 41382.223 |
| 1 | 4 | 8 | 41443.286 |
| 2 | 2 | 2 | 41644.593 |
| 2 | 4 | 2 | 41460.215 |
| 2 | 4 | 8 | 41384.077 |
| 4 | 2 | 2 | 41368.703 |
| 4 | 4 | 2 | 49098.892 |
| 4 | 4 | 8 | 41344.725 |
| 1 | 4 | 4 | 2 | 80332.622 |
| 2 | 4 | 4 | 80340.533 |
| 4 | 4 | 4 | 80320.536 |
| 1 | 2 | 2 | 80279.049 |
| 1 | 4 | 2 | 80511.309 |
| 1 | 4 | 8 | 80342.450 |
| 2 | 2 | 2 | 80325.463 |
| 2 | 4 | 2 | 80413.442 |
| 2 | 4 | 8 | 80324.221 |
| 4 | 2 | 2 | 72595.757 |
| 4 | 4 | 2 | 72566.868 |
| 4 | 4 | 8 | 72558.297 |

**Analiza:**

*Testarile s-au realizat pe un CPU Intel I3 12100F 4 cores 8 threads.*

* **Java**

In cazul in care intervalul de timp dintre trimiterea calupurilor este mai mare, se poate observa o imbunatatire a timpului pentru cazurile in care durata de viata a cacheului pentru clasamentul pe tari este unul mai mare. Deoarece calculul acestui clasament este unul suficient de costisitor dpdv al timpului, iar numarul de requesturi este unul mare, folosirea rezultatului calculat anterior pentru alt client duce la imbunatatire. De asemenea, numarul workerilor are un impact pozitiv, dar minor asupra timpului.

Aceasta performanta este observata si in cazul in care intervalul intre requesturile de procesare este unul mai redus. Totodata, acesti timpi sunt mai apropiati datorita utilizarii cacheului in aproape toate cazurile in contextul trimiterii aproape succesive a datelor. Timpul de viata al acestuia, insa, isi pierde din impact, deoarece procesarea acestor requesturi se realizeaza intr-un timp mai scurt.

