Readme – Detalii implementare tema 2 APD

Musat Andreea-Alexandra, 333CA

Evenimentele sunt implementate folosind o clasa abstracta Query care este mai apoi extinsa prin alte 4 clase, fiecare fiind specifica unui anumit tip de eveniment (PrimeQuery, FiboQuery, FactQuery, SqrtQuery). Clasa Query extinde Runnable, deci fiecare clasa-copil trebuie sa contina metoda run() in care sa se rezolve query-ul respectiv si sa se adauge rezultatul obtinut in vectorul corespunzator din clasa Main.

Clasa PrimeQuery contine o metoda statica isPrime care testeaza daca un numar este sau nu prim si foloseste aceasta metoda pentru a rezolva task-ul. Pentru a gasi cel mai mare numar mai mic ca N care este prim, se verifica, pe rand, daca numerele incepand cu N si continuand cu N-1, N-2, .. sunt prime. In momentul cand primul numar prim este gasit, acesta este adaugat in lista primeRes din Main si executia thread-ului este oprita.

Clasele FiboQuery si FactQuery prezinta o abordare diferita. Fiecare contine un arrayList static de intregi care stocheaza numerele din sirul respectiv (numerele fibonacci sau factorial). Aceste arrayList-uri vor fi generate o singura data (generandu-se numerele mai mici ca N-ul maxim gasit in fisiere), iar apoi, pe baza acestora, se va realiza o cautare liniara pentru a gasi elementul dorit. (Solutia query-ului este reprezentata de indicele elementului din array care are proprietatea ca elementul curent este mai mic sau egal cu N, iar elementul urmator, daca exista, este mai mare ca N). Dupa gasirea solutiei, aceasta este adaugata in lista de rezultate corespunzatoare (fiboRes sau factRes).

Metoda run() a clasei SqrtQuery doar adauga in lista sqrtRes, simplu, sqrt(N) rotunjit la cel mai mare intreg mai mic ca el.

Citirile din fisier se fac in paralel cu ajutorul unui ExecutorService. In clasa ReadFromFile ce extinde Runnable exista metoda run() care creeaza un fisier nou cu un nume dat, iar apoi cu ajutorul unui BufferedReader citeste linie cu linie fisierul, fiecare linie fiind procesata astfel: linia este despartita in tokens-i separati de virgula, cu semnificatia: primul token reprezinta numarul de milisecunde pe care thread-ul trebuie sa il astepte dupa citirea liniei, al doilea token este tipul evenimentului, iar al treilea token este N-ul specific query-ului. Pe baza acestora este creat un nou query avand tipul respectiv si este adaugat intr-o coada intermediara din Main.

Query-urile sunt apoi preluate din aceasta coada intermediara (cu dimensiune fixata) si adaugate intr-un alt ExecutorService responsabil cu rezolvarea evenimentelor. Cand ambele ExecutorService-uri termina, avem certitudinea ca listele de rezultate sunt complete si pot fi sortate, iar apoi afisate in fisierele de output.

Observatii: listele de rezultate sunt sincronizate, deoarece este posibil ca mai multe thread-uri sa incerce sa adauge elemente in acelasi timp, iar unele valori sa se piarda.