README

Dina Ștefania-Gabriela 325CC

Ianuarie 2021

1 Gradul de dificultate. Timpul alocat.

Nivelul de dificultate al task-urilor 1 si 3 nu a fost unul ridicat ca si tehnici de programare, insa ce consider dificil este volumul de cod ce trebuie gandit, pentru ca te poti pierde rapid. Timpul alocat pentru rezolvarea task-urilor 1 si 3 a fost de aproximativ 12 ore.

2 Detalii de implementare

Cum am specificat si mai sus, rezolvarea temei consta in implementarea claselor si a design pattern-urilor. Ce consider relevant sa mentionez si de explicat din implementare este:

2.1 Metoda getRecruiter(User user) din clasa Recruiter

Prima data am facut BFS pe intreaga retea sociala a user-ului. Pentru BFS am folosit un LinkedList prin care simulez comportamentul de coada si un HashMap care are rolul de a tine evidenta distantei de la radacina la userii din reteaua sociala. De fiecare data cand gasesc un user nou, il adaug in coada, respectiv in hashmap cu distanta corespunzatoare. Daca unul dintre cunoscutii userului ce a fost scos din coada are o intrare inregistrata in hashmap, atunci verific daca distanta prin parintele curent este mai mica decat distanta deja calculata, caz in care sterg vechea intrare, o adaug pe cea noua si adaug din nou userul in coada.
In continuare, creez un ArrayList in care retin angajatii care sunt cei mai indepartati social. Daca la final, in ArrayList, sunt mai multi recruiteri, il voi alege pe cel cu ratingul cel mai mare. Altfel, pe primul din lista.
Tin sa mentionez ca metoda getDegreeInFriendship(Consumer consumer) este similara cu cea explicata mai sus. Aceasta face un BFS, dar care foloseste un Hashtable.

2.2 Metoda process(Job job) din clasa Manager

Imi creez un ArrayList ce contine tupluri de user si score, in care retin toti candidatii pentru job impreuna cu scorurile asociate. Sortez lista in ordine

descrescatoare dupa scor si apoi o parcurg. Pentru fiecare element din lista, verific daca userul nu este angajat deja, caz in care il adaug in lista de angajati si il scot din lista de observeri ai companiei. Ma opresc cand s-au umplut toate locruile disponibile. In final, sterg restul de requesturi, setez jobul ca fiind inchis si notific toti userii care au ramas neangajati pe acest post de inchiderea sa.

2.3 Metoda evaluate() din clasa Recruiter

In primul rand, verific daca candidatul indeplineste conditiile postului. Daca indeplineste toate conditiile, i se calculeaza scorul si se creeaza requestul de job, care apoi este adaugat in lista de aplicanti ai managerului firmei unde candidatul doreste sa se angajeze.

2.4 Metoda getExperienceYears() din clasa User

Metoda calculeaza cu aproximatie numarul de ani de experienta pe care ii are un user. Pentru data de inceput si de final al unui stagiu/loc de munca am folosit tipul LocalDate. Ideea implementarii este urmatoarea: calculez numarul de ani lucrati si numarul de luni lucrate (daca se acumuleaza un numar de luni mai mare de 12, atunci incrementez numarul de ani cu 1).