**DOCUMENTATIE TEMA5**

**LAMBDA EXPRESSION AND STREAM PROCESSING**

Nume si prenume:Pop Cristian

Grupa: 30223

**Cuprins**

1.Cerinta

2.Obiective

3.Analiza problemei

4.Proiectare

5.Implementare

6.Testare

7.Concluzii si dezvoltari ulterioare

**1.Cerinta**

Considerati cerinta de analizare a comportamentului unei mersoane monitorizate de un set de senzori.Istoricul activitatilor persoanei respective este stocat ca un triplet(inceputul activitatii,sfarsitul activitatii si numele activitatii),unde inceputul activitatii si sfarsitul activitatii reprezinta data si ora la care a inceput activitatea,respective s-a incheita activitatea,iar numele activitatii reprezinta activitatea desfasurata de respective persoana in intervalul de timp.Aceste activitati sunt:dormit,mic dejun,tv,pranz,cina,gustare,plecare,dus.

Activitatile aceste persoane sunt monitorizate pe mai multe zile si sunt stocate in fisierul “Activity.txt” localizat in acest folder.

Scrieti un program Java 1.8 folosing expresii lambda si stream processing pentru a rezolva cerintele din tabel.

1. Definiți o clasă MonitoredData cu 3 câmpuri: ora de începere, ora de încheiere și activitatea ca șir. Citiți datele din fișierul Activity.txt folosind fluxurile și împărțiți fiecare rând în 3 părți:start\_time, end\_time și eticheta de activitate și să creați o listă de obiecte de tip MonitoredData.
2. Aflați câte zile de date monitorizate apar în jurnal.
3. Determinați un map cu tipul <String, Integer> care se potrivește fiecărui tip de acțiune distinct numărul de apariții în jurnal. Scrieți map-ul rezultată într-un fișier text.
4. Aflati de cate ori a aparaut fiecare activitate pentru fiecare zi din periaoda monitorizata.
5. Pentru fiecare linie din file map pentru activitate aflati durata acesteia.
6. Pentru fiecare activitate calculate durata totala a acesteia pe parcursul monitorizarii.
7. Filktrati activitatile care au 90% din perioada de monitorizare cu durata mai mica de 5 minute.
8. Documentatie.

**2.Obiective**

Obiectivul acestei teme familiarizarea cu expresii lambda si stream processing si avantajul utilizarii acestora.

|  |  |  |
| --- | --- | --- |
| **Obiectiv Secundar** | **Descriere** | **Capitol** |
| Dezvoltarea de use case-uri si scenarii |  | 3 |
| Alegerea structurilor de date |  | 4 |
| Impartirea pe clase |  | 4 |
| Dezvoltarea algoritmilor |  | 4 |
| Implementarea solutiei |  | 5 |
| Testare |  | 6 |

**3.Analiza problemei**

Aceasta tema nu mai necesita crearea une interfete grafice,astefel userul va trebui doar sa ruleze programul pentru a vedea rezultatele.Citind cu atentie cerinta problemei observam ca vom avea nevoie de trei variabile,doua de tipul LocalDateFormat si una de tip String pentru inceputul/sfarsitul activitatii si pentru numele acesteia.Vom citi linie cu linie si vom stoca elementele in aceste trei valori(aceste variabile isi vor modifica valorile primate la fiecare linie).De asemenea vom avea nevoie dupa cum zice si prima cerinta de o Lista de tipul MonitoredData,o lista care contine elementele de tipul enumerate mai sus si in care vom stoca fiecare linie.In aceasta lista vom avea practic tot ce se afla in fisierul Activity.txt,doar ca datele vor fi impartite in cele trei categorii de mai sus.Dupa aceasta va trebui sa rezolvam cerintele din tabel.

Pnetru rezolvarea cerintelor vom folosi anumite functii lambda:streamuri(transforma datele tale in streamuri pentru a putea fi folosite in functii lambda),map(selecteaza o anumita primitive din obiect),filter(selecteaza anumite obiecte care respecta o conditie).distinct(returneaza doar elemente unice),counting(Numara elemente din lista).

Nefiind nevoie de o interfata grafica utilizatorul trebuie doar sa verifice corectitudinea operatiilor implementate.

Ultilizatorul ruleaza programul.

Sistemul efectueaza toate operatiile implementate.

Sistemul afiseaza rezultatele operatiilor in ordinea in care acestea au fost apelate.

Utilizatorul verifica corectitudinea datelor furnizate de system.

**4.Proiectare**

Pentru proiectarea acestei teme am folosit trei clase:MonitoredData,Operatii si Main.In clasa MonitoredData am folosit doar trei variabile:start\_time,end\_time si activity.In clasa Operatii implementez o rezolvarea a cerintei.In clasa Main creez un obiect de tip Operatii care va afisa rezultatul cerintelor rezolvate.

**5.Implementare**

In clasa MonitoredData am declarant trei variabile.Primele doua start\_time si end\_time sunt de tip LocalDateTime,ele fiind chivalente cu primele doua campuri din fisierul “Activity.txt” si respective activity de tip String pentru al treilea camp reprezentand numele activitatii.

De asemenea in aceasta clasa mai exista metode de get si set pentru fiecare din aceste variabile.

In clasa Operatii ne cream functiile care rezolva cerintele.Functiile implementate sunt crearea unei liste de tip MonitoredData in care sunt stocate informatiile din Activity.txt,acestea fiind si afisate pe ecran,o functie pentru a returna nr de zile pe parcursul carora se desfasoara monitorizarea,si o functie care returneaza de cate ori a aparut fiecare activitate pe perioada monitorizata sub forma unui map de timp <String,Long>.

In prima functie de create folosim un String pentru a memora locatie fisierului Activity.txt,iar apoi vom folosi un stream pentru a putea citi datele din fisier.Ne mai declaram inca doua variabile de tip LocalDateTime si una de tip String,date1,date2,activity,si un vector de string care separa elementele din fisisier cu ajutorul functie line.split.In variabilele declarate mai sus adaugam datele de pe prima linie,iar apoi acestea le daugam in lista de tipul MonitoredData.Acest process se repeat pana cand am daugat toate elementele din fisier in lista.

A doua functie folosita este o functie de afisare in care pur si simplu afisam lista cu elementele pe care le contine.

Urmatoarea functie este cea care calculeaza nr de zile in care se desfasoara monitorizarea.Si aici utilizam un stream care filtreaza datele.Am ales data la care incepe o activitate cu ajutorul functie din MonitoredData start() care returneaza data la care incepe o activitate.Astfel filtrarea are loc in functie de aceasta data.Se mai foloseste functia distinct pentru a nu aparea o zi de doua ori si de asemenea functia counting() care Numara zilele.Nr acesta de zile este stocat intr-o variabila result de tip integer care la final este afisata.

Urmatoarea functie este cea care calculeaza de cate ori a aparut fiecare activitate pe parcursul perioadei de monitorizare.Pentru aceasta am folosit un map de tip<String,Long> si un string in care ne-am folosit de aceasta data de functia activities() pentru a primi activitatile.De asemenea am folosit si functia count pentru a le Numara.La final am parcurs aceste activitati si am afisat de cate ori a aparut fiecare din aceste activitati pe toata durata de monitorizare.

In clasa Main am creat un obiect de tip Operatii,iar mai apoi cu ajutorul acestui obiect am apelat functiile din Operatii in ordinea de mai sus.

.

**7.Concluzii si dezvoltari ulterioare**

In concluzie expresiile lambda sunt o metoda buna pentru a obtine anumite date din monitorizarea programului unei persoane.Acestea nu sunt o necessitate,dar sunt o metoda mult mai eficienta decat altele,in special pentru acest tip de problema.