Documentatie Tema5
-Lambda Expressions and Stream Processing-

Popa Alexandra Maria
Grupa 30224
CTI-ro

Cuprins:

1.Obiectivul Temei:

Enuntul Temei : Propunerea, proiectarea si implementarea unei aplicatii care are ca scop analizarea comportamentului unei persoane inregistrat de un set de senzori. Jurnalul de activitati al persoanei este stocat ca tuple (start_time, end_time, activity_label), unde start_time și end_time reprezintă data și ora la care fiecare activitate a început și sa încheiat, în timp ce eticheta de activitate reprezintă tipul de activitate desfasurat de persoana: iesirei, toaletare, dus, dormit, mic dejun, pranz, cina, snack, timp de rezervă / TV, îngrijire. Datele sunt distribuite pe parcursul mai multor zile ca multe intrări în jurnalul Activities.txt. Se cere dezvoltarea unui program care sa utilizeze expresii lambda și procesarea fluxurilor pentru a face sarcinile definite.

Obiective :

Obiectiv principal : Obiectivul acestei teme este proiectarea si implementarea unei aplicatii de monitorizarea a timpului petrecut de o persoana in scopul delrularii mai multor activitati avand in vedere utilzarea anumitor tehnici. Aplicatia trebuie sa utilizeze tehnici de procesare a flixurilor de date venite de la senzori si totodata Lambda Expressions in vederea preluarii si filtrarii acestor informatii.

Obiective secundare :

- dezvoltarea algoritmilor folosind POO -> algoritmi de preluarea si procesarea a informatiilor preluate dintr-un fisier .txt
- utilizarea structurilor de date -> descrierea structurilor de date utilizate in implementarea aplicatiei
- formularea de scenarii -> modalitati de preluare si filtrare a informatiilor provenite de la senzorii ce monitorizeaza activitatiile / timp ale unei persoane
- implementarea si testarea solutiilor obtinute -> testare prin vizualizarea datelor obtinute in urma unor opeartii efectuate de preluare, analizare si filtrare a datelor dintr-un fisier ; aceste informatii sunt stocate in struncturi de date de tip Map si HashMap .

2.Analiza problemei, scenarii, modelare, cazuri de utilizare:

Problema enuntata necesita cunostinte legate de lucrul cu fluxuri de date stocate dintr-un fisier .txt si provenite de la un set de senzori si totodata, procesarea acestora prin utilizarea Lambda Expressions. Aplicatia trebuie sa indeplineasca de asemenea urmatoarele functionalitati si cerinte necesare :

- definirea unei clase MonitoredData ce contine 3 variabile instanta : startTime, endTime ca String; se vaor citi de asemenea date din fisierul Activity.txt utilizand streams si se va imparti fiecare rand in 3 parti : start_time, end_time si activity_label si se va crea o lista de obiecte de tip MonitoredData.
- afisarea numarului de zile care apar pe parcurul monitorizarii activiatilor
- contorizarea numarului de aparitii pentru fiecare activitate pe intreaga perioada de monitorizare; rezultatul va fi reprezentat de o strunctura de date de tipul Map<String,Long> reprezentand maparea activitatiilor la numarul lor.
- afisarea numarului de aparitii pentru fiecare zi inclusa in perioada de monitorizare
- obtinerea duratei pentru fiecarea activitate din interiorul Map-ului (end_time – start_time)
- pentru fiecare activitate, calcularea duratei intregii perioade de monitorizare
- filtarea activitatiilor care au 90% din inregistrarile de monitorizare cu o durata mai mica de 5 minute

Din analiza problemei rezulta faptul ca avem nevoie de un set de date de intrare si un set de date de iesire. Datele de intrare sunt reprezentate de informatiile stocate in fisierul Activity.txt, in tip ce datele de iesire sunt reprezentate de rezultatele obtinute in urma apelarii metodelor din intermediul clasei MonitoredData.

3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator) :

In continuare voi prezenta proiectarea in mare a clasei ce formeaza aplicatia, metodele acesteia fiind descrise mai pe larg in capitolul 4.

Proiectarea aplicatiei de procesaee a stream-urilor a necesitat proiectarea unei singure clase : MonitoredData . Acesata clasa contine 3 variabile instanta : startTime – ce reprezinta timpul de incepul al unuei activitati; endTime- reprezentand timpul de sfarsit al unei activitati; si activity – reprezentand denumirea activitatii. Toate aceste variabile instanta sunt de tipul Sting si prin intermediul acestora se stocheaza datele de pe o linie din fisierul Activity.txt . Clasa contine metode de get si set pentru fiecare din variabilele instanta, precum: getStartTime( ), getEndTime( ), getActivity( ), setStartTime(String startTime), setEndTime(String endTime) si setActivity(String activity), totodata implementand metoda *toString( )* utilizata pentru afisarea obiectelor de tip MonitoredData. De asemenea, clasa contine urmatoarele metode ce implementeaza cazurile de utilizare ale aplicatiei : createList( ), countDays(List<MonitoredData> list), activityFrequence(List<MonitoredData> list), getDay( ), activityFrequenceForDay(List<MonitoredData> list), getDuration( ), addDuration(Duration sum,String startTime,String endTime), getDurationForLine(List<MonitoredData> list), activityEntireDuration(List<MonitoredData> list) si activityTimeFilter(List<MonitoredData> list).

Ca structura de date, a fost folosita structura de tip *HashMap*, utilizata pentru stocarea rezultatelor de tip cheie-valoare obtinute in urmaaprelucratii datelor de intrare. Totodata, am utilizat si strub=cturi de date de tipul List si ArrayList pentru stocarea linilor din interiorul fisierului Activity.txt.

4. Implementare :

In acest capitol se vor descrie deciziile de implementare ale metodelor clasei MonitoreData, impreuna cu algoritmii ce sunt folositi in realizarea acestora .

createList( ) -> metoda ce returneaza o lista de elemente de tip String, elemente ce rerezinta fiecare linie din fisierul Activity.txt

countDays(List<MonitoredData> list) -> metoda ce numara cate diferite apar pe parcurul intregii perioade de monitorizare a activitatiilor

activityFrequence(List<MonitoredData> list) -> returneaza un element de tipul Map<String,Long> ce reprezinta maparea fiecarei activiati impreuna cu numarul de aparitii.

getDay( ) -> returneaza un element ce reprezinta ziua sub forma de String continuta de variabila instanta startTime

activityFrequenceForDay(List<MonitoredData> list) -> returneaza numarul activitatea si numarul de aparitii al acestei activitati pe arcursul unei singure zile sub forma de Map<String, Map<String, Long>>

getDuration( ) -> returneaza un obiect de tip Duration ce reprezinta diferenta e timp dintre startTime si endTime

addDuration(Duration sum,String startTime,String endTime) -> un obiect de tip Duration ce reprezinta noua valoare obtinuta in uram adugarii duratei de timp dintre startTime si endTime la o valoare de timp Duration reprezentata de sum

getDurationForLine(List<MonitoredData> list) -> returneaza un obiect de tip Map<String, String> in care este stocata o linie dintr-un fisier si durata de timp dintre startTime si endTime aferenta acesteia

activityEntireDuration(List<MonitoredData> list) -> returneaza un obiect de tip Map<String,Duration> in care se stocheaza numele unei activitati si intreaga durata de desfasurare de pe perioada de minitorizare

activityTimeFilter(List<MonitoredData> list) -> returneza o strctura de date de tip Set<String> ce reprezinta rezultatele obtinute in urma filtrarii activitatiilor care au 90% din inregistrarile de monitorizare cu o durata mai mica de 5 minute

5. Rezultate:

Rezultatele obtinute reprezentate sunt afisate apelarii metodelor clasei MonitoreData.

6.Concluzii :

In urma acestei teme am invatat sa imi structurez mai bine codul in clasa de lucru, astfel incat acesta sa respecte pricipiile POO. Am fost pusa in situatia de a realiza monitorizarea datelor obtinute de la un set de sensor si totodata am fost pusa in situatia de a doband cunistiintele necesare prelucrarii stream-urilor dar si a lucrului cu Lambda Expresions.

Ca imbunatatiri ulterioare, aplicatia mea ar putea beneficea de o interfata grafica care sa afiseze informatii referitoare la activitatiile si monitorizarea acestora pentru o persoana pe parcurul zilei.

7. Bibliografie :

- Curs TP
- HW5_Indications
- https://stackoverflow.com/
- https://www.baeldung.com/java-date-difference?fbclid=IwAR3AGi6WiB2o7IRBlMF4NCsfwG4Wyt2c6BwiqMTdhHVrtsFlnS6IqDf-EB4
- https://www.mkyong.com/tutorials/java-8-tutorials/?fbclid=IwAR00WRDz08I9dy8Ah3lHuRE3saHmuF5ZQN__AlNn_w9PZBaMtwACc6FfEls