MINISTRY OF EDUCATION AND RESEARCH
TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

TEHNICI DE PROGRAMARE

TEMA_5

PROCESSING SENSOR DATA OF DAILY LIVING ACTIVITIES

Student: Tóth Szilveszter

Gr.: 30224

1) Obiectivul temei

La aceasta tema am avut de implementat un program cu ajutorul careia putem prelucra date dintr-o lista de activitati. Fisierul de intrare are structura urmatoare: data in care a inceput activitatea, data in care a incheiat, si denumirea activitatii. Tema este impartita pe 6 task-uri, fiecare avamd o cerinta aparte.

2) Analiza probleme

Aceasta tema are in vedere exersarea si aplicarea java stream-urilor, de aceea ne vom axa pe cat se poate pe acest aspect. Un stream este o secventa de obiecte la care suporta a gama larga de functii cu ajutorul carora se pot face pipe-uri pentru a obtine rezultatul dorit. Cele mai importante functii ar fi urmatoarele: .filter (la care ii spune si numele, ne ajuta sa filtram tuplele dupa un criteriu anuma), .map (cu care putem mapa/descrie ceea ce se intampla in stream), .grouppingBy (cu care puteam grupa tuplele dupa un anumit camp), forEach(cu care putem traversa continuturi si a le afisa pe consola) etc.

Cum am mentionat si mai devreme, aceasta tema se imparte pe 6 subpuncte. Streamurile fiind destul de scurte in ceea ce priveste codatul lor, vom scrie toate taskurile intr-o clasa Main, ne avand rost sa le punem toate in clase diferite (ar fi un pic redundant). Vom avea nevoie de o clasa in care vom avea definitia unui obiect in care vom stoca datele din fisierul Activities.txt. Aceasta clasa (numita MonitoredData) are ca si parametri 3 stringuri: start_time, end_time si activity_label. In acestea vom stoca informatiile mentionate la punctul 1. Rezultatele la toate cele 6 taskuri va trebui sa afisam in fisiere output separate.

3) Proiectare si implementare

In ceea ce priveste implementarea, cum am mentionat si mai sus, vom avea doar doua clase: un main si o definitie de obiect.

Task1

Acesta consta in crearea clasei MonitoredData si citirea/testarea acesteia. Clasa dispune doar de un constructor (ceea ce se poate vedea mai jos) si niste getter-e si setter-e.

```
public String start_time;
public String end_time;
public String activity_label;
public MonitoredData(String start_time, String end_time, String activity_label) {
        super();
        this.start_time = start_time;
        this.end_time = end_time;
        this.activity_label = activity_label;
}
```

Dupa ce citim datele din fisier (ca si stringuri) trebuie sa le spargem ca sa putem construi obiecte de tip MonitoredData. Pentru asta vom apela la ajutorul functie split din regex, cu ajutorul careia vom sparge stringurile pe

rand in functie de spatiul dintre ele (mai precis, in fisierul activities se afla la o distanta de doua tab -uri). Mai apoi le salvam intr-o lista de tip MonitoredData. Un fragment din codul care se ocupa de aceste operatii este:

```
for(String s : inList)
{
        String arr[]=s.split("                    ",-2);
        int len=arr[2].length();
        for (int i=0; i<len; i++)
        {
                if (arr[2].charAt(i)==' ' || arr[2].charAt(i)='          ')
                {
                        arr[2]=arr[2].substring(0,i) + arr[2].substring(i+1);


                }
        }
        activityList.add(new MonitoredData(arr[0],arr[1],arr[2]));
}
```

Dupa asta mai urmeaza o simpla scriere in fisier cu numele Task_1.txt.

Task2

Acesta ne cere sa numaram toate zilele distincte in care are loc monitorizarea. Acest lucru este foarte simpla avand in vedere ca stream-urile au o functie de filtrare implicita. Filtram un functie de data, mai apoi rezultatele le punem intr-o lista. La final vom Numara elementele acestei liste, care va fi totodata rezultatul droit pe care o vom scrie in fisierul Task_2.txt.

```
Map<Object, List<MonitoredData>> groupDate = activityList.stream()
        .collect(Collectors.groupingBy(x -> x.getDate(0,10), Collectors.toList()));
int cont=groupDate.size();
```

Task3

La acest subpunct trebuie sa numaram fiecare activitate de cate ori a fost ,,mentionat,, in fisierul de intrare. Ca si la exercitiul anterior, rezolvarea consta din cateva linii de cod, urmat de o scriere in fisier. Grupam activitatile in functie de numele lor, mai apoi numaram de cate ori apare in lista noua creata in timpul streamului.

```
Map<Object, List<MonitoredData>> groupActivities = activityList.stream()
        .collect(Collectors.groupingBy(x -> x.getActivity_label(), Collectors.toList()));

groupActivities.forEach((activity, number)-> {
        try {
                outWrite3.write(activity + " -> appears " + number.size() + " times!\n");
        } catch (IOException e) {
                System.out.println("Can't write the output at Task3!");
        }
});
```

Task4

Acesta este o continuare a problemelor anterioare, mai precis o combinatie intre ele, fiindca trebuia sa numaram de cate ori apare fiecare activitate in fiecare zi. In primul rand trebuie sa grupam activitatile in functie de data in care au loc, mai apoi sa numaram in aceasta lista create de cate ori apare fiecare activitate.

```
Map<Object, List<MonitoredData>> groupActivitiesDate = activityList.stream()
        .sorted(Comparator.comparing(x -> x.getDate(0,10)))
        .collect(Collectors.groupingBy(x -> x.getDate(0,10)));

groupActivitiesDate.forEach((date, activities)-> {
                try {
                        outWrite4.write(date + "\n");
                } catch (IOException e) {
                        e.printStackTrace();
                }

Map<Object, List<MonitoredData>> groupActivitiesNumber = activities.stream()
        .collect(Collectors.groupingBy(x -> x.getActivity_label(), Collectors.toList()));
        groupActivitiesNumber.forEach((activity, number)-> {
                try {
                        outWrite4.write(activity + " -> appears " + number.size() + " times!\n");
```

```
                } catch (IOException e) {
                        e.printStackTrace();
                }
        });
});
```

Task5

Aici ni se cerea sa calculam timpul total cat a tinut o activitate. Pentru asta am scris o functie auxiliara, care primeste doua stringuri, si calculeaza diferenta lor (defapt cat timp a tinut o activitate). Aici trebuie doar se spargem stringurile, sa convertim numerele obtinute in inturi si sa scadem timpii de start din timpii de finalizare. Functia va returna durata activitatii in secunde. In prealabil am grupat tuplele in functie de numele activitatilor. Dup ace am aflat timpul total, trebuie sa o afisam corespunzator (ora, minute, secunde), pentru a se putea verifica mai usor.

```
groupActivitiesTimes.forEach((activity, number)-> {
        int time=0;
        for (MonitoredData m : number)
        {
                time+=calculateDuration(m.getStart_time(),m.getEnd_time());
        }
        try {
                outWrite5.write(activity + " took place for " + Math.abs(time/3600) + " hours "
+ Math.abs(time/3600/60) + " minutes and " + Math.abs(time%60) + " seconds\n");
        } catch (IOException e) {
                System.out.println("Can't write the output at Task5!");
        }
});
```

Task6

Ne cerea sa filtram activitatile in functie de niste criterii impuse.

4) Rezultate

Rezultatele rularii programului se pot vedea in fisierele de output. De exemplu pentru Task3, rezultatul arata in felul urmator:

Breakfast -> appears 14 times!

Grooming -> appears 51 times!

Toileting -> appears 44 times!

Sleeping -> appears 14 times!

Leaving -> appears 14 times!

Spare_Time/TV -> appears 77 times!

Snack -> appears 11 times!

Showering -> appears 14 times!

Lunch -> appears 9 times!

5) Concluzii

La aceasta tema am avut foarte mult de invatat. Pana acum nu am mai facut java stream, de aceea mi s-a parut destul de greu. Dar dupa ce am facut primele taskuri, mergea mai bine. Aceasta abordare difera destul de mult (in ceea ce priveste modul de scriere) fata de modalitatile cu care am fost obisnuit. Dar in mare, m-a ajutat destul de mult aceasta lectie.

6) Bibliografie

https://winterbe.com/posts/2014/07/31/java8-stream-tutorial-examples/

https://www.youtube.com/watch?v=Q93JsQ8vcwY

https://www.youtube.com/watch?v=t1-YZ6bF-g0&t=364s