Documentatie – Tema 5

PROCESSING SENSOR DATA OF DAILY LIVING ACTIVITIES

Nume: Dascal Raluca Georgiana

Grupa: 30226

Profesor coordonator: Dorin Moldovan

Cuprins:

1. Cerinte Functionale

Sa se implementeze si sa se testeze o aplicatie pentru analiza comportamentului unei personae inregistrat de catre un set de senzori instalat in casa ei.

2. Obiective

2.1 Obiectiv Principal

Dezvoltarea unei aplicatii care ca stocheze istoricul de activitati al unei persoane care contine toate actiunile pe care acesta le interprinde si care sunt caracterizate prin tripletul (start_time, end_time, activity_label), unde start_time si end_time reprezinta datele si orele intre care activitatea se desfasoara, in timp ce activity_label reprezinta tipul de activitate interprinsa de persoana. Tipurile de activitati posibile sunt: Leaving, Toileting, Showering, Sleeping, Breakfast, Lunch, Dinner, Snack, Spare_Time/TV, Grooming. Istoricul cuprinde activitatile pentri cateva zile si este retinut in fisierul text Activities.txt. Se cere generarea unor rezultate in ceea ce privesc activitati. Aceste rezultate vor fi explicate in urmatoarele capitole

Pe langa aceasta functionalitate aplicatia trebuie sa mai respecte cateva reguli precum: folosirea programarii orientate pe obiect(concept aprofundat in semestrul trecut), crearea unor clase cu maxim 300 de linii de cod, implemantarea unor metode cu maxim 30 de linii de cod, folosirea conventiilor pentru nume Java, folosirea expresiilor lambda, precum si generarea unui fisier executabil .jar care sa respecte formatul de apel.

2.2 Obiective Secundare

Obiective secundare	Capitol
Descriera generala a aplicatiei	3
Descriera algoritmilor utilizati	5.2
Prezentarea claselor	6
Rezultatele obtinute	7

3. Descrierea Generala a Aplicatiei

Aceasta aplicatie este utila pentru orice persoana deoarece ii permite acesteia sa isi monitorizeze toate activitatile si astfel sa o ajute in definirea unui program mai eficient. Pe langa stocarea istoricului de activitati aceasa aplicatie permite si aplicarea unor algoritmi ce presupun generarea unor rezultate privind numarul de activitati si de zile a caror istoric s-a realizat, sau returnarea numarului de activitati diferite atat pe parcursul unei zile cat si pe parcursul intregului experiment. Aplicatia vine atat in ajutorul persoanelor care au un program foarte aglomerat si nu reusesc sa isi duca la indeplinire totalitatea sarcinilor deoarece aceasta ii prezinta un rezumat al activitatilor sale astfel reusind sa isi defineasca un program mai eficient, cat si in ajutorul persoanelor care pierd foarte mult timp fie in fata televizorului fie in orice alt timp de activitate- aplicatia o ajuta sa constientizeze cat de mult timp se risipeste astfel determinand-o sa isi recalculeze intervalul de timp petrecut facand activitati neproductive.

4. Analiza problemei

In figura de mai jos (fig 4.1) este descrisa intr-o forma grafica relatia dintre utilizator (orice persoana), aplicatie si furnizarea rezultatelor obtinute in urma parcurgerii unui algoritm. Am ales sa realizez aplicatie si in aceasta forma de prezentare pentru a fi mai usor de inteles pentru un potent utilizator atat din punct de vedere functional cat si din punct de vedere a datelor necesare pentru a putea fi utilizata.

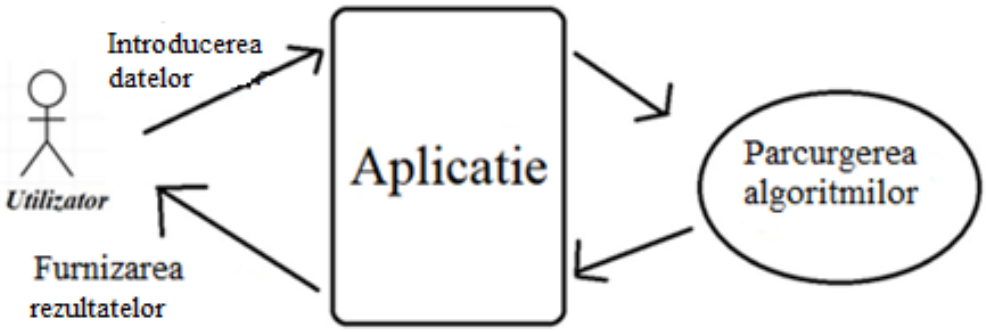

Fig 4.1

5. Proiectare

5.1 Diagrama de clase

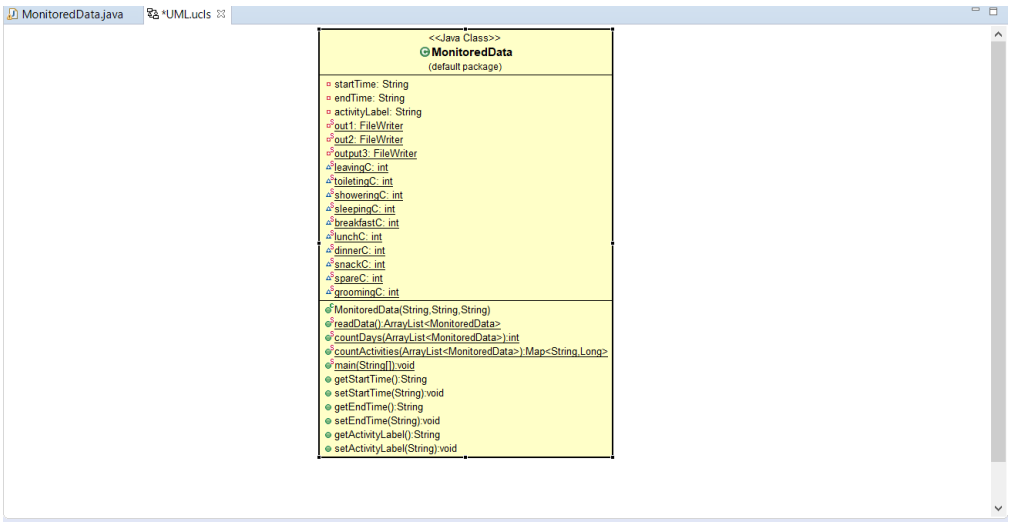

Fig5.1.1

In figura de mai sus (fig 5.1.1) este prezentata diagrama UML a clasei proiectului realizat. Aceasta driagrama a fost sintetizata de catre programul in care a fost scris si codul, Eclipse Java 2019. In aceasta imagine se evidentiaza atat clasa proiectului cat si continutul acesteia (cu varabilele declarate si metodele ce definesc fie constructori, gettere ,settere fie algoritmi utili in realizarea functionalitatilor aplicatiei) Aceste diagrama UML ne ajuta la formarea unei imagini in ansamblu a proiectului ajutand totodata si la intelegerea unor concepte si paradigme a programarii orientate pe obiect.

5.2 Descrierea algoritmilor implementati

Fiecare dintre metodele definite si descrise in acesat aplicatie Java, cu exceptia constructorilor sau a celor care seteaza si returneaza valorile variabilelor declarate in clase, descriu cate o sarcina pe care proiectul trebuie sa o indeplineasca folosind un algoritm mai mult sau mai putin riguros.Algoritmii implementati in acest proiect au o saricna foarte bine definita si urmaresc realizarea functionalitatii corecte a cerintelor acestei teme. Descrierea implementarii si functionalizarii fiecare metode si utilitatea fiecarei variabile (aparute in diagram UML a claselor prezentata in capitolul anterior) vor fi prezentate in amanunt in urmatorul capitol.

6. Implementare

In acest capitol se va prezenta clasa proiectului si se va explica utilitatea sau scopul fiecarei variabile declarate si metode implementate in aceste proiect avand la baza diagrama UML sintetizata de catre programul in care a fost scris codul.Crearea clasei s-a realizat in pachetul default al aplicatiei utilizate Eclipse 2019

6.1 Clasa MonitoredData

In figura alaturata este prezentata diagrama UML a clasei programului MonitoredData. Aceasta clasa indeplineste toate cerintele functionalitatii aplicatiei solicitate in tema curenta. Variabilele (startTime, endTime si activityLabel) reprezinta termenii de caracterizare a fiecarei activitati. startTime si endTime reprezinta timpul de incepul, respectiv timpul finalizarii fiecarei activitati, in timp ce activityLabel reprezinta denumirea activitatilor effectuate de catre o persoana intr-un interval de timp marit. Ultima variabila poate sa ia oricare din urmatoarele valori: Leaving, Toileting, Showering, Sleeping, Breakfast, Lunch, Dinner, Snack, Spare_Time/TV, Grooming. Aplicatia stocheaza istoricul activitatilor pe o perioada de cateva zile, adica contine detaliile prezentate mai sus despre fiecare activitate. Variabilele out1, out2, out3 reprezinta fisierele in care se vor scrie rezultatele obtinute in urma parcurgerii metodelor implementate in aceasta clasa si explicate in cele ce urmeaza. Urmatoare variabile reprezinta cate un counter pentru fiecare tip de activitatea. Utilizarea si necesitatea acestora vor fi prezentate in acest capitol. Metoda MonitoredData reprezinta una dintre metodele de baza ale claselor Java, adica un constructor. Metoda readData() este o metoda care returneaza o lista ce contine elemente de tip MonitoredData, adica reprezinta istoricul de activitati realizate de catre o persoana pe parcursul a catorva zile. Aceasta metoda realizeaza cititrea detaliilor despre activitati dintr-un fisier text care primeste aceste informatii de la un set de senzori amplasati in casa persoanei tintite. Pentru metoda countDays am realizat

doua moduri de a implementa aceasta metoda deoarece prima metodaa presupune realziarea acesteia cu concept de baza, in timp ce cea de a doua implementare utilizeaza expresii lambda, un concept nou. Acesta metoda presupune numararea zilelor pe parcursul carora s-a realizat experimental, adica numarul de zile a caror istoric de activitato este retinut in aplicatie. Metoda countActivities esre implementata de asemenea in doua moduri, iar motivul este explicat mai sus, este acelasi cu cel pentru metoda countDays. Aceasta metoda presupune numararea tuturor activitatilor in functie de denumirea lor realizate pe parcusul a catorva zile. Acesta metoda returneaza o variabila de tip MAP care contine ca si cheie denumirea activitatii, in timp ce cheia este reprezentata de catre numarul de repetari pentru activitatea cu denumirea respective in intervalul de timp calculate de catre metoda precedenta. Metoda Main contine definirea a 3 fisire de text, fiecare dintre ele continand rezultatele obtinute la metodele implementate in aceasta clasa adica: readData, countDays. countActivities. Fisierele de text s-au definit cu nume corespunzator urmatorului format: Task_number.txt, number reprezentand ordinea implementarii metodelor – rezultatul readData- 1, rezultatul countDays- 2, rezultatul countActivities3. Tot in metoda main se realizazeaza si scrierea in aceste metode a rezultatelor prezentate mai sus. Metodele getStartTime, getEndTime, getActivityLabel reprezinta getterele pentru variabilele ce contin informatiile despre activitati, adica returneaza valorile acestora in timp ce setStartTime, setEndTime, setActivityLabel reprezinta settere pentru variabilele ce contin informatiile despre activitati, adica permite schimbarea valorii acestestora.

7.Rezultate

2011-11-28 02:27:59	2011-11-28 10:18:11	Sleeping
2011-11-28 10:21:24	2011-11-28 10:23:36	Toileting
2011-11-28 10:25:44	2011-11-28 10:33:00	Showering
2011-11-28 10:34:23	2011-11-28 10:43:00	Breakfast
2011-11-28 10:49:48	2011-11-28 10:51:13	Grooming
2011-11-28 10:51:41	2011-11-28 13:05:07	Spare_Time/TV
2011-11-28 13:06:04	2011-11-28 13:06:31	Toileting
2011-11-28 13:09:31	2011-11-28 13:29:09	Leaving
2011-11-28 13:38:40	2011-11-28 14:21:40	Spare_Time/TV
2011-11-28 14:22:38	2011-11-28 14:27:07	Toileting
2011-11-28 14:27:11	2011-11-28 15:04:00	Lunch
2011-11-28 15:04:59	2011-11-28 15:06:29	Grooming
2011-11-28 15:07:01	2011-11-28 20:20:00	Spare_Time/TV
2011-11-28 20:20:55	2011-11-28 20:20:59	Snack
2011-11-28 20:21:15	2011-11-29 02:06:00	Spare_Time/TV
2011-11-29 02:16:00	2011-11-29 11:31:00	Sleeping
2011-11-29 11:31:55	2011-11-29 11:36:55	Toileting
2011-11-29 11:37:38	2011-11-29 11:48:54	Grooming
2011-11-29 11:49:57	2011-11-29 11:51:13	Showering
2011-11-29 12:08:28	2011-11-29 12:18:00	Breakfast
2011-11-29 12:19:01	2011-11-29 12:22:00	Grooming
2011-11-29 12:22:38	2011-11-29 12:24:59	Spare_Time/TV
2011-11-29 13:25:29	2011-11-29 13:25:32	Snack
2011-11-29 13:25:38	2011-11-29 15:12:26	Spare_Time/TV
2011-11-29 15:13:28	2011-11-29 15:13:57	Toileting
2011-11-29 15:14:33	2011-11-29 15:45:54	Lunch
2011-11-29 15:49:51	2011-11-29 15:50:54	Grooming
2011-11-29 15:52:04	2011-11-29 16:17:58	Spare_Time/TV
2011-11-29 16:18:00	2011-11-29 16:31:27	Toileting
2011-11-29 16:34:17	2011-11-29 17:08:07	Spare_Time/TV
2011-11-29 17:08:58	2011-11-29 17:09:29	Toileting
2011-11-29 17:43:00	2011-11-29 18:57:22	Spare_Time/TV
2011-11-29 19:02:15	2011-11-29 20:23:38	Leaving

Fig 7.1

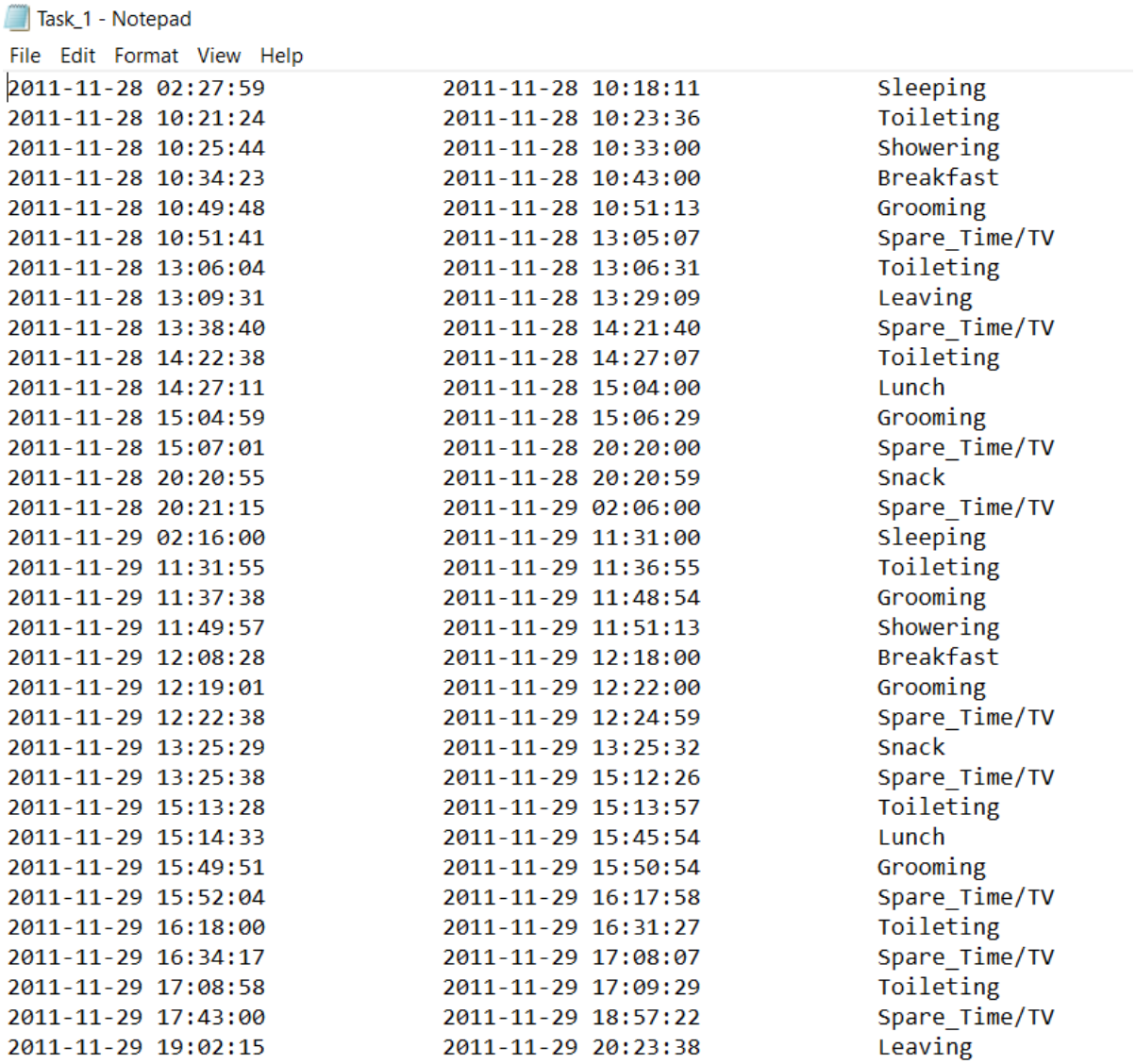
Task_1 - Notepad
File Edit Format View Help
2011-11-28 02:27:59 2011-11-28 10:18:11 Sleeping
2011-11-28 10:21:24 2011-11-28 10:23:36 Toileting
2011-11-28 10:25:44 2011-11-28 10:33:00 Showering
2011-11-28 10:34:23 2011-11-28 10:43:00 Breakfast
2011-11-28 10:49:48 2011-11-28 10:51:13 Grooming
2011-11-28 10:51:41 2011-11-28 13:05:07 Spare_Time/TV
2011-11-28 13:06:04 2011-11-28 13:06:31 Toileting
2011-11-28 13:09:31 2011-11-28 13:29:09 Leaving
2011-11-28 13:38:40 2011-11-28 14:21:40 Spare_Time/TV
2011-11-28 14:22:38 2011-11-28 14:27:07 Toileting
2011-11-28 14:27:11 2011-11-28 15:04:00 Lunch
2011-11-28 15:04:59 2011-11-28 15:06:29 Grooming
2011-11-28 15:07:01 2011-11-28 20:20:00 Spare_Time/TV
2011-11-28 20:20:55 2011-11-28 20:20:59 Snack
2011-11-28 20:21:15 2011-11-29 02:06:00 Spare_Time/TV
2011-11-29 02:16:00 2011-11-29 11:31:00 Sleeping
2011-11-29 11:31:55 2011-11-29 11:36:55 Toileting
2011-11-29 11:37:38 2011-11-29 11:48:54 Grooming
2011-11-29 11:49:57 2011-11-29 11:51:13 Showering
2011-11-29 12:08:28 2011-11-29 12:18:00 Breakfast
2011-11-29 12:19:01 2011-11-29 12:22:00 Grooming
2011-11-29 12:22:38 2011-11-29 12:24:59 Spare_Time/TV
2011-11-29 13:25:29 2011-11-29 13:25:32 Snack
2011-11-29 13:25:38 2011-11-29 15:12:26 Spare_Time/TV
2011-11-29 15:13:28 2011-11-29 15:13:57 Toileting
2011-11-29 15:14:33 2011-11-29 15:45:54 Lunch
2011-11-29 15:49:51 2011-11-29 15:50:54 Grooming
2011-11-29 15:52:04 2011-11-29 16:17:58 Spare_Time/TV
2011-11-29 16:18:00 2011-11-29 16:31:27 Toileting
2011-11-29 16:34:17 2011-11-29 17:08:07 Spare_Time/TV
2011-11-29 17:08:58 2011-11-29 17:09:29 Toileting
2011-11-29 17:43:00 2011-11-29 18:57:22 Spare_Time/TV
2011-11-29 19:02:15 2011-11-29 20:23:38 Leaving

Fig 7.2


Task_2 - Notepad
File Edit Format View Help
Numarul de zile e 14

Fig 7.3

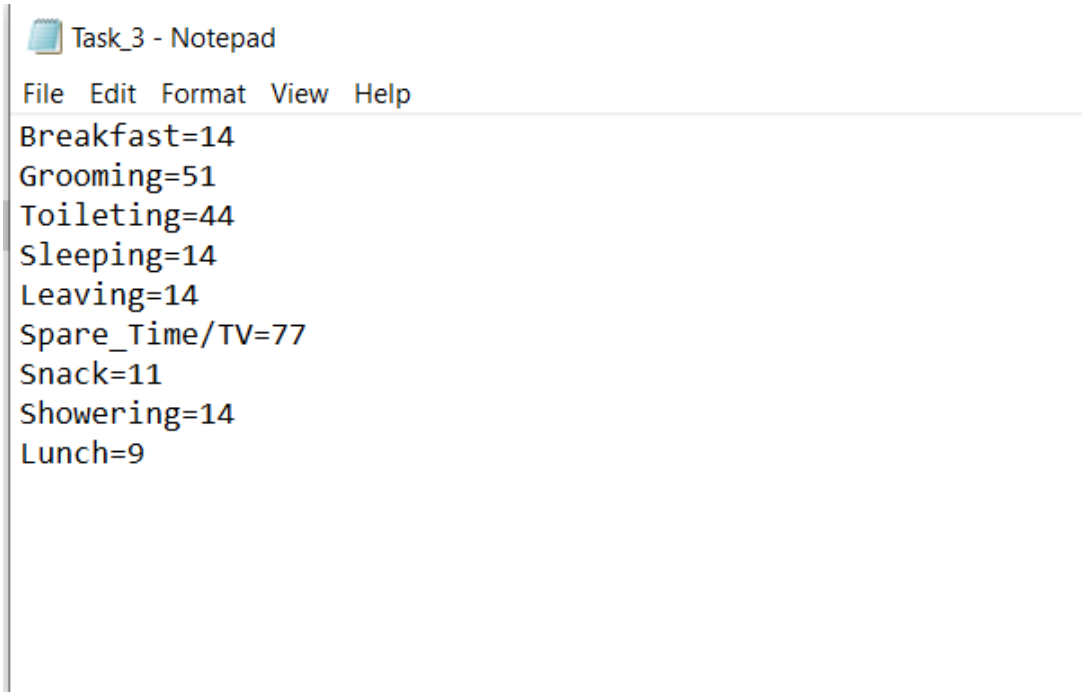

Fig 7.4

In figura 7.1 este prezentat fisierul text de intrare, adica fisierul care contine istoricul de activitati realizat pe o perioada de cateva zile si primite de catre setul de senzori amplasati in casa persoanei tintite. In figura 7.2 este realizata prima cerinta, aceea de a retine toate informatiile intr-o lista si de afisa continutul fisierului text de intrare. Figura 7.3 reprezinta cea de a doua cerinta, adica aceea de a Numara zilele pe parcursul carora s-a realizat experimentul. In ultima imagine este realiza cea de a treia cerinta, adica numararea activitatilor in functie de denumirea lor. In acest capitor se poate observa atat functionatlitatea aplicatiei, cat si corectitudinea rezultatelor obtinute in urma parcurgerii algoritmilor implementati.

8. Concluzii si posibilitati de dezvoltare ulterioare

In timpul dezvoltarii acestei teme, adica in timpul realizarii unei aplicatii care ca stocheze istoricul de activitati al unei persoane care contine toate actiunile pe care acesta le interprinde si care sunt caracterizate prin tripletul (start_time, end_time, activity_label), unde start_time si end_time reprezinta datele si orele intre care activitatea se desfasoara, in timp ce activity_label reprezinta tipul de activitate interprinsa de persoana precum si realizarea unor operatii pe aceste date, consider ca am reusit sa aprofundez atat unele paradigme si concepte ale programarii orientate pe obiect cat si imbunatariea abilitatile mele de a scrie cod si/ sau a corecta eventualele erori atat de sintaxa cat si la nivel de gandire aparute pe parcursul descrierii algoritmilor necesare functionalitatilor acestei aplicatii. Pe langa acestea am realizat cat de importanta este citirea cu atentie si intelegerea unei cerinte si capabilitatea de a structura/ imparti problema initiala in mai multe probleme mai mici si inceperea rezolvarii acestora ca un prim pas in indeplinirea sarcinilor unei teme,ca apoi sa asociezi toate rezultatele obtinute din surse pentru a ajunge la un rezultat corect in ansamblu.. Mi-am imbunatatit si abilitatea de a cauta informatii utile si corecte pentru a reusi ducerea proiectului meu intr-o stare mai complexa si mai corecta din punct de vedere a functionalitatii.

O posibilitate de dezvoltare a proiect realizat si descris in acesta documentatie ar putea fi:

- Adaugarea mai multor operatii care sa ajute persoanele sa isi realizeze un program mai eficients
- mai mult de atat s-ar putea face o revizuire asupra tuturor algoritmilor folositi si realizarea implementarii acestora intr-o maniera mai eficienta din punct de vedere a timpului de executie sau a memoriei folosite sau mai usor de descris (pentru programator) folosind functii deja definite ale pachetelor Java, astfel reusind o reducere a timpului petrecut pentru asigurarea functionarii unui astfel de aplicatii.
- o alta posibilitate de dezvoltare a proiectului ar putea fi imbunatatirea proiectului realizand chiar si o interfata grafica pentru ca utilizatorului sa-I fie cat mai usor intelegerea rezultatelor si a utilizarii aplicatiei

9. Bibliografie

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

https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html

https://www.baeldung.com/java-groupingby-collector

https://www.jackrutorial.com/2018/07/how-to-print-map-in-java-10.html