***Project Οντοκεντρικού Προγραμματισμού 2020-2021***

* ***Στοιχεία μελών ομάδας:***

Βασίλειος Μπαρδάκης - 1088098 - up1088098@upnet.gr

Χαρίτων Κικίδης - 1084595 - up1084595@upnet.gr

Θεόδωρος Πλάτωνας - 1090073 - up1090073@upnet.gr

* **Σύνδεσμος για τον κώδικα :** [**https://www.mediafire.com/file/c5cwdx619rs00z9/Kwdikas\_Bardakis\_Kikidhs\_Platwnas.rar/file**](https://www.mediafire.com/file/c5cwdx619rs00z9/Kwdikas_Bardakis_Kikidhs_Platwnas.rar/file)
* **Διάγραμμα UML:**



* **Περιγραφή υλοποίησης.**

Δεν έχουμε γράψει για τα constructors, είναι απλά με μια super και μια αρχικοποιήση μέσα.

**Κλάση User**

Έχει πεδία όπως ζήταγε η εκφώνηση. Φτιάξαμε getters και setters για να μπορούμε να ορίσουμε και να ανακτήσουμε τις private μεταβλητές.

**Κλάση Admin**

Φτιάξαμε μόνο 1 set και get για το isAdmin αφού για name και phone χρησιμοποιούμε τις μεθόδους της User.

**Κλάση Donator**

Έχει μεθόδους add remove, monitor, reset και modify οι οποίες κανουν wrap τις μεθόδους της RequestDonationList και την commit() που κάνει wrap την commit της Offers. Επιπλέον έχουμε φτιάξει getter για το Offerslist.

**Κλάση Beneficiary**

Φτιάξαμε διαφορετικού τύπου add και modify επειδή θέλουμε να χρησιμοποιούν διαφορετικές μεθόδους. Η add, remove και modify είναι για την receivedList και χρησιμοποιούν την add και remove της RequestDonationList, ενώ οι addReq και modifyReq χρησιμοποιούν τις μεθόδους της Requests εφόσον θέλουμε να κάνουν έλεγχους πριν κάνουν τις λειτουργείες τους αφού αφορά της λίστα requestsList.

επιπλέον έχουμε κάνει wrap την commit της requests και έχουμε φτιάξει και getters για τις receivedList, requestsList και noPersons.

**Κλάση Entity**

Έχουμε φτιάξει getters και setters για τα στοιχεία της. Επιπλέον έχουμε την abstract μέθοδο getDetails ώστε να μπορούν να την χρησιμοποιούν αντικείμενα Material και Service ενώ τα βλέπουμε ώς Entity. Η getEntityInfo και toString είναι όπως ζητάει η εκφώνηση.

**Κλάση Material**

Υλοποιήσαμε την returnLevel να επιστρέφει το ανάλογο level ανάλογα με τον αριθμό int που του βάζουμε σαν όρισμα.

**Κλάση Service**

Η getDetails απλά επιστρέφει Service ώστε να μπορούμε να αναγνωρίσουμε μετά τον τύπο entity με το indexOf.

**Κλάση RequestDonation**

Μερικά getters και stters για Entity και Quantity.

**Κλάση Organization**

Μερικά setters και getters για τις μεταβλητές που ορίζουμε.

addEntity, insertDonator και insertBeneficiary προσθέτουν αντικείμενα στις λίστες του οργανισμού με χρήση exception. Οι remove αφαιρούν απο τις λίστες.

insertDonator και insertBeneficiary επιστρέφουν true ή false ανάλογα με άν έγινε η προσθήκη ή όχι.

Μερικές μεθόδοι list που τυπώνουν τα περιεχόμενα των λίστων του οργανισμού.

**Κλάση RequestDonationList**

Setters και getters για τις μεταβλητές που ορίζουμε.

Η μέθοδος get κανει iterate την rdEntities και αν το id του RequestDonation είναι ίδιο με το id που της έχουμε βάλει ως όρισμα, το επιστρέφει. Άμα δεν βρεί RequestDonation με ιδιο id επιστρέφει null.

Η add εξετάζει αν είναι στα Entities του Organization με χρήση εξαίρεσης. Μετά χρησιμοποιούμε την boolean check ώστε αν υπάρχει ήδη το αντικείμενο στην λίστα που πάμε να το προσθέσουμε, τα προστίθονται οι ποσότητες, ενώ αν δεν το βρεί στην λίστα να το προσθέτει ως καινούριο RequestDonation.

Οι remove και reset χρησιμοποιούν iterator αντί για for ώστε να μπορούμε να διαγράψουμε απο την λίστα ενώ την κάνουμε iterate.(άν και απο την remove ίσως να μην χρειάζεται εφόσον κάνουμε break; μετά απο όταν βρίσκουμε το αντικείμενο).

Η remove χρησιμοποιεί ένα boolean check για να ελέγχουμε την περίπτωση που δεν υπάρχει το αντικείμενο που βάλαμε σαν όρισμα στην rdEntities.

monitor και modify είναι όπως τις ζητάει η εκφώνηση.

**Κλάση Requests**

Η validRequestDonation, σε περίπτωση που είναι material (το βρίσκουμε με το indexOf) με ελέγχους που ψάχνουν τα requestsList και receivedList για να βρούν αν το RequestDonation του ορίσματος ανοίκει σε αυτές, θέτουμε το ανάλογο Quantity, με αυτό ελέγχουμε αν είναι στο επιτρεπόμενο όριο ανάλογα με τα levels του material και το noPersons του beneficiary. Αν είναι service απλα επιστρέφει true.

**Κλάση Offers**

Έχει μόνο μια commit η οποία χρησιμοποιεί iterator για να μπορεί να διαγράφει αντικείμενα απο την offersList ενω την κανει iterate.

**Κλάση Menu**

Οι βοηθητικές συναρτήσεις που χρησιμοποιήσαμε είναι στο κάτω μέρος της Menu και είναι οι ακόλουθες.

readAnswer() : με αμυντικό προγραμματισμό διαβάζει y/n και το επιστρέφει.

chooseAction() : με αμυντικό προγραμματισμό και χρήση Exception επιστρέφει το νούμερο που διαβάζει εφόσον είναι μέσα στο min/max του ορίσματος.

readDouble(): με αμυντικό προγραμματισμό και χρήση exception διαβάζει και επιστρέφει double.

printMatSer(): Τυπώνει τα materials ή services του organization ανάλογα με το string που βάζουμε ώς όρισμα (αυτό το κάνουμε με την μέθοδο indexOf).

getMatSer(): Τυπώνει ερώτηση ποιό entity θέλει ο χρήστης να επιλέξει και δίνει ένα αριθμό (counter+1) που πρέπει να πατήσουμε αν θέλουμε να πάμε πίσω. Σε περίπτωση που πατήσουμε το counter+1 επιστρέφει true. Σε περίπτωση που πατήσουμε απο το 1 μέχρι το counter αρχικοποιεί την int id με το id του entity και το RequestDonation rd με το RequestDonation της CurrentDonations με το ίδιο id. (Έχουμε βάλει να επιστρέφει true ώστε να μπορούμε να το χρησιμοποιήσουμε μετά για να πάει ένα βήμα πίσω το πρόγραμμα, το rd το χρησιμοποιούμε επίσης για να αρχικοποιήσουμε άλλα RequestDonation).

Στην αρχή του menu φτιάχνουμε την RunMenu() η οποία είναι ένας ατέρμον βρόγχος while, μέσα σε αυτήν χρησιμοποιούμε τις:

printHeader() : που απλά τυπώνει το welcome message.

printAuthentication() : ζητάει το τηλέφωνο του χρήστη, αν υπάρχει αναγνωρίζει αν είναι admin, donator ή beneficiary και ορίζει το κατάλληλο status (το status το χρησιμοποιεί η RunMenu ώστε μπεί στο κατάλληλο υπο-menu). Αν δεν υπάρχει το τηλέφωνο σε καμια λίστα, τότε το status παραμένει 0 και έτσι καλείται η register στην RunMenu().

Αν το status είναι 0 καλείται η register που πραγματοποιεί την εγγραφή του χρήστη.Στην register με χρήση της boolean τιμής που επιστρέφει η insertBeneficiary/insertDonator, σε περίπτωση που εισαχθεί όνομα που ήδη υπάρχει, ζητάει να εισαχθεί καινούργιο.

printDon/Admin/BenMenu : εμφανίζουν τα κατάλληλα υπο-menu.

Σε όλες τις μεθόδους perform (πχ performAdminAction) έχουμε πολλούς while βρόγχους ώστε να μπορούμε να δουλέψουμε με back.

performAdminAction(): Κάνει ότι λέει η εκφώνηση, με χρήση των chooseAction και switch επιλέγει ο χρήστης, με χρήση των printMatSer και getMatSer τυπώνει και επιλέγει αντίστοιχα τα material/service του organization, έχουμε και την περίπτωση που θέλει να επιλέξει να πάει πίσω όταν επιλέγει material/service με την boolean μεταβλητή της getMatSer. Στην επιλογή monitor organization όταν επιλέγει ο χρήστης donator ή beneficiary, έχουμε προσθέση επιλογή back πατώντας τον αριθμό πλήθος beneficiaries+1. Όταν επιλεχθεί αρχικοποιεί το object beneficiary ή donator και προχωράει στο επόμενο επίπεδο. Τα υπόλοιπα πιστεύω είναι αρκετά κατανοητά.

performBenAction(): Με την επιλογή add request χρησιμοποιούνται οι printMatSer και getMatSer όπως την performAdminAction(). Όταν επιλεχθεί ένα αντικείμενο για να γίνει request, το αρχικοποιούμε στο rd2 και μετά το βάζουμε στην addReq της Beneficiary ενώ έχουμε διαβάσει με αμυντικό προγραμματισμό την ποσότητα με την χρήση της readDouble(), αν είναι επιτυχής τυπώνεται κατάλληλο μύνημα και προστίθεται στα requestsList. Στην show requests δεν χρησιμοποιούμε την monitor επειδή θέλουμε την λίστα αριθμημένη, άν δεν υπάρχουν αντικείμενα στην λίστα μας επιστρέφει στο αρχικό υπο-menu.

Όταν επιλεχθεί να γίνει Add Request ή Modify Request του beneficiary (που καλεί την add της requests) πάντα χρησιμοποιείται η addReq και modifyReq ώστε να γίνουν οι έλεγχοι για το άν ο Beneficiary δικαιούται αυτήν την ποσότητα.

PerformDonAction(): Δουλεύει σχεδον ίδια με την performBenAction() μόνο που οι μεθόδοι που καλεί δεν είναι της requests οπότε δεν εκτελούν ελέγχους.

Πάντα στις παραπάνω κλάσεις όπου χρησιμοποιούμε switch έχουμε default που κλείνει το πρόγραμμα σε περίπτωση που κάτι πάει λάθος, κάθε είσοδο του χρήστη έχει αμυντικό προγραμματισμό. Δεν σχολίασα σε υπερβολική λεπτομέρεια τις μεθόδους επειδή πιστεύω θα είναι πιο εύκολο να τα δείτε με τα σχόλια που έχει μέσα ο κώδικας.

**Κλάση Main**

Αρχικοποιούμε τα στοιχεία της static organization, στην συνέχεια προσθέτουμε και αρχικοποιούμε έναν admin, έναν donator, 2 beneficiaries, 3 material και 3 services, έπειτα για να είναι πιο εύκολος ο έλεγχος προσθέτουμε μερικά donations απο τον dnator και τα κάνουμε commit και έπειτα μερικά requests απο τους 2 beneficiaries και κάνουμε μόνο μερικά απο αυτα commit (έτσι εξηγούνται και τα μυνήματα πρίν την menu).

Όταν προσθέτουμε τα Entities, τα προσθέτουμε και ώς RequestDonation με amount 0 στην λίστα του currentDonations Organization επειδή σε μερικές μεθόδους του προγράμματος την χρησιμοποιούμε για να αρχικοποιήσουμε μερικά RequestDonation.

**Κλάσεις Εξαιρέσεων**

**CustomExceptionClass**

Την χρησιμοποιούμε στα υπόλοιπα exceptions κάνωντας την throw.

**!Όλες οι κλάσεις εξαιρέσεων απο εδώ και κάτω είναι static ώστε να μην χρειάζεται να φτιάχνουμε αντικείμενα αυτών κάθε φορα που τις χρησιμοποιούμε!**

**ID\_Exception**

Για την περίπτωση που πάμε να προσθέσουμε ενα Entity στο EntityList του οργανισμού ενώ υπάρχει ήδη entity με ιδιο id στην λίστα.

**Ben\_Exception**

Για την περίπτωση που προσθέτουμε Beneficiary ο οποίος έχει το ίδιο όνομα με άλλον.

**Don\_Exception**

Για την περίπτωση που προσθέτουμε Donator ο οποίος έχει το ίδιο όνομα με άλλον.

**RequestException**

Περιέχει πολλού είδους exceptions. Αρχικά έχουμε βάλει την περίπτωση που το RequestDonation του ορίσματος είναι null. Άν δεν είναι null ελέγχουμε αν ο οργανισμός έχει αρκετό Quantity για να δώσει. Μετά καλούμε την validRequestDonation για το αν δικαιούται ο beneficiary αυτήν την ποσότητα.

**RequestDonation\_ExcStructure**

Εξετάζει την περίπτωση που το entity που θέλουμε να προσθέσουμε δεν είναι στα Entities του οργανισμού.

***ΚΩΔΙΚΑΣ***

* **Organization**

import java.util.List;

public class Organization

{

private static String name;

private static Admin admin;

private static List<Entity> entityList;

private static List<Donator> donatorList;

private static List<Beneficiary> beneficiaryList;

private static RequestDonationList currentDonations;

public static void setAdmin(boolean value){

admin.setIsAdmin(value);

}

public static Admin getAdmin(){

return admin;

}

//add & insert - remove

public static void addEntity(Entity enti){

try{

ID\_Exception.CheckID(enti);

entityList.add(enti);

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

}

public static void removeEntity(Entity enti){

entityList.remove(enti);

}

public static boolean insertDonator(Donator dona){

try{

Don\_Exception.CheckName(dona);

donatorList.add(dona);

return true;

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

return false;

}

public static void removeDonator(Donator dona){

donatorList.remove(dona);

}

public static boolean insertBeneficiary(Beneficiary bene){

try{

Ben\_Exception.CheckName(bene);

beneficiaryList.add(bene);

return true;

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

return false;

}

public static void removeBeneficiary(Beneficiary bene){

beneficiaryList.remove(bene);

}

//Getters

public static String getName(){return name;}

public static List<Entity> getEntityList() {

return entityList;

}

public static List<Donator> getDonatorList() {

return donatorList;

}

public static List<Beneficiary> getBeneficiaryList() {

return beneficiaryList;

}

public static RequestDonationList getCurrentDonations() {

return currentDonations;

}

//Setters

public static void setName(String namee){

name=namee;

}

public static void setAdmin(Admin adminn) {

admin = adminn;

}

public static void setEntityList(List<Entity> entityListt) {

entityList = entityListt;

}

public static void setDonatorList(List<Donator> donatorListt) {

donatorList = donatorListt;

}

public static void setBeneficiaryList(List<Beneficiary> beneficiaryListt) {

beneficiaryList = beneficiaryListt;

}

public static void setCurrentDonations(RequestDonationList currentDonationss) {

currentDonations = currentDonationss;

}

public static void listEntities(){

System.out.println("Materials:\n");

for (var ent:entityList){

if (ent instanceof Material)

System.out.println(ent.getName());

}

System.out.println("Services:\n");

for (var ent:entityList){

if (ent instanceof Service)

System.out.println(ent.getName());

}

}

public static void listBeneficiaries(){

System.out.println("\nList of beneficiaries:\n");

for (var ben:beneficiaryList){

ben.getReceivedList().monitor();

}

}

public static void listDonators(){

System.out.println("List of donators:\n");

for (var don:donatorList){

System.out.print("\n");

don.printName();

}

}

}

* **Entity**

abstract public class Entity {

private String name; //onoma

private String description; //perigrafi

private int id; //id

public Entity(String name,String description,int id){

this.name=name;

this.description=description;

this.id=id;

}

//Getters

public String getName() {

return name;

}

public String getDescription() {

return description;

}

public int getId() {

return id;

}

//Setters

public void setName(String name) {

this.name = name;

}

public void setDescription(String description) {

this.description = description;

}

public void setId(int id) {

this.id = id;

}

//Methods

public String getEntityInfo(){

return getName()+getDescription() +String.valueOf(getId());

}

public abstract String getDetails();

public String toString(){

return getEntityInfo()+getDetails();

}

}

* **Material**

public class Material extends Entity

{

private double level1,level2,level3;

public Material(String name,String description, int id,double level1,double level2,double level3)

{

super(name,description,id);

this.level1=level1;

this.level2=level2;

this.level3=level3;

}

public double returnLevel(int person)

{

if(person==1)

return level1;

else if(person>=2&&person<=4)

return level2;

else

return level3;

}

public String getDetails(){

return ("Material \nLevel 1 = "+level1+"\nLevel 2 = "+level2+"\nLevel 3 = "+level3);

}

}

* **Service**

public class Service extends Entity

{

public Service(String name,String description, int id)

{

super(name,description,id);

}

public String getDetails(){

return "\nService";

}

}

* **User**

abstract class User {

private String name;

private String phone;

public User(String name, String phone) {

this.name = name;

this.phone = phone;

}

public String getName() {

return name;

}

public String getPhone() {

return phone;

}

public void setName(String name) {

this.name = name;

}

public void setPhone(String phone) {

this.phone = phone;

}

}

* **Admin**

public class Admin extends User

{

private boolean isAdmin=true;

public Admin(String name, String phone) {

super(name,phone);

}

public void setIsAdmin(boolean bo){isAdmin=bo;}

public boolean getIsAdmin(){return isAdmin;}

}

* **Beneficiary**

public class Beneficiary extends User

{

private int noPersons=1;

private RequestDonationList receivedList;

private Requests requestsList;

//Constructor

public Beneficiary(String name, String phone,int noPersons, RequestDonationList receivedList,Requests requestsList){

super(name,phone);

this.noPersons=noPersons;

this.receivedList=receivedList;

this.requestsList=requestsList;

}

//Wrapper methodoi

public void add(RequestDonation obj){ receivedList.add(obj);}

public void remove(RequestDonation obj){

receivedList.remove(obj);

}

public void modify(RequestDonation obj,double quantity){receivedList .modify(obj,quantity); }

public void addReq(Beneficiary ben,RequestDonation obj){requestsList.add(ben,obj);}

public void modifyReq(Beneficiary ben,RequestDonation obj, double quantity){

requestsList.modify(ben,obj,quantity);

}

public void commit(Beneficiary ben){

requestsList.commit(ben);

}

//Setters

public void setPersons(int persons){this.noPersons=persons;}

public void setReceivedList(RequestDonationList receivedList){this.receivedList=receivedList;}

public void setRequestsList(Requests requestsList){this.requestsList=requestsList;}

//Getters

public RequestDonationList getReceivedList(){return receivedList;}

public Requests getRequestsList(){return requestsList;}

public int getPersons(){return noPersons;}

}

* **Donator**

import java.util.List;

public class Donator extends User {

private Offers offersList;

//Constructor

Donator(Offers offersList,String name, String phone){

super(name,phone);

this.offersList=offersList;

}

public void printName(){

System.out.println(getName());

}

public Offers getOffersList(){return offersList;}

//Wrapper methods

public void add(RequestDonation obj){

offersList.add(obj);

}

public void remove(RequestDonation obj){

offersList.remove(obj);

}

public void modify(RequestDonation obj,double quantity){

offersList.modify(obj,quantity);

}

public void reset(){

offersList.reset();

}

public void monitor(){

offersList.monitor();

}

public void commit(){

offersList.commit();

}

}

* **RequestDonation**

public class RequestDonation

{

private double quantity;

private Entity entity;

public RequestDonation(Entity entity, double quantity){

this.quantity=quantity;

this.entity=entity;

}

public double getQuantity(){return quantity;}

public Entity getEntity(){return entity;}

public void setEntity(Entity entity){this.entity=entity;}

public void setQuantity(double quantity){this.quantity=quantity;}

}

* **RequestDonationList**

import java.util.Iterator;

import java.util.List;

import java.util.ArrayList;

public class RequestDonationList

{

private int counter;

private RequestDonation don;

private List<RequestDonation> rdEntities = new ArrayList();

private boolean check;

//Methods

public void setRdEntities(List<RequestDonation> rdEntities){this.rdEntities=rdEntities;}

public List<RequestDonation> getlist(){return rdEntities;}

public RequestDonation get(int id){

for (var ent:rdEntities)

{

if (ent.getEntity().getId()==id)

return ent;

}

return null;

}

public void add(RequestDonation obj){

try{

check=false;

RequestDonation\_ExcStructure.RequestDonationAdd(obj.getEntity());

for (var ent:rdEntities)

{

if (ent.getEntity().getId()==obj.getEntity().getId()){

ent.setQuantity(ent.getQuantity()+obj.getQuantity());

check=true;

}

}

if(check==false){rdEntities.add(obj);}

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

}

public void remove(RequestDonation obj){

check=false;

//Kanei iterate thn rdEntities, vriskei to antikeimeno kai to diagrafei.

for (Iterator<RequestDonation> iterator = rdEntities.iterator(); iterator.hasNext(); ) {

RequestDonation don = iterator.next();

if (don.getEntity().getId() == obj.getEntity().getId()) {

iterator.remove();

check = true;

break;

}

}

//Se periptwsh pou to antikeimeno pou valame san orisma den uparxei

if (check==false){System.out.println("This entity does not exist in this list");}

}

public void modify(RequestDonation obj,double quantity){

for (var ent:rdEntities)

{

if (ent.getEntity().getId()==obj.getEntity().getId())

obj.setQuantity(quantity);

}

}

public void monitor(){

counter=0;

for (var ent:rdEntities)

{

counter+=1;

System.out.println("Name: "+ent.getEntity().getName()+"||"+" Quantity: "+ent.getQuantity());

}

if(counter==0){System.out.println("This list is empty!\n");}

}

public void reset(){

for (Iterator<RequestDonation> iterator = rdEntities.iterator(); iterator.hasNext(); ) {

RequestDonation don = iterator.next();

iterator.remove();

}

}

}

* **Offers**

import java.util.List;

import java.util.Iterator;

public class Offers extends RequestDonationList

{

public void commit(){

for (Iterator<RequestDonation> iterator = getlist().iterator(); iterator.hasNext();) {

RequestDonation off = iterator.next();

Organization.getCurrentDonations().add(off);

iterator.remove();

}

}

}

* **Requests**

import java.util.List;

import java.util.Iterator;

public class Requests extends RequestDonationList

{

private double Quantity;

private double Quantity2;

private RequestDonation don;

private boolean check;

public boolean validRequestDonation(Beneficiary ben,RequestDonation don)

{

Quantity2=0;

if(don.getEntity().getDetails().indexOf("Material")!=-1)

{

//Periptwsh pou einai sthn requestsList kai receivedList

if(ben.getRequestsList().get(don.getEntity().getId())!=null && ben.getReceivedList().get(don.getEntity().getId())!=null)

{

//Periptwsh pou to antikeimeno pou vrhkame sthn requestsList den einai to idio me to don ara to prosthetoume

if(ben.getRequestsList().get(don.getEntity().getId())!=don) {

Quantity2 = ben.getRequestsList().get(don.getEntity().getId()).getQuantity();

}

Quantity=Quantity2 + ben.getReceivedList().get(don.getEntity().getId()).getQuantity() + don.getQuantity();

}//Periptwsh pou einai sthn requestsList alla oxi sthn receivedList

else if(ben.getRequestsList().get(don.getEntity().getId())!=null)

{

//Periptwsh pou to antikeimeno pou vrhkame sthn requestsList den einai to idio me to don ara to prosthetoume

if(ben.getRequestsList().get(don.getEntity().getId())!=don) {

Quantity2 = ben.getRequestsList().get(don.getEntity().getId()).getQuantity();

}

Quantity= Quantity2+ don.getQuantity();

}//Periptwsh pou den einai se kamia

else

Quantity=don.getQuantity();

if(Quantity>((Material)don.getEntity()).returnLevel(ben.getPersons()))

return false;

else

return true;

}

else

return true;

}

public void add(Beneficiary ben,RequestDonation obj){

try{

//xrhsh exception

RequestException.RequestException(ben, obj);

ben.getRequestsList().add(obj);

System.out.println("Request added successfully");

}

catch(CustomExceptionClass e){System.out.println(e.getMessage());}

}

public void modify(Beneficiary ben,RequestDonation obj,double quantity){

try {

check = false;

Quantity2 = obj.getQuantity();

if (obj == null) {

System.out.println("You do not have an active request for this Entity.");

} else {

//xrhsh exception

RequestException.RequestException(ben, obj);

check = true;

ben.getRequestsList().modify(obj, quantity);

System.out.println("Successful modification!");

if (!check) {

obj.setQuantity(Quantity2);

}

}

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

}

public void commit(Beneficiary ben){

Iterator<RequestDonation> iterator = ben.getRequestsList().getlist().iterator();

try {

while(iterator.hasNext()) {

RequestDonation don = iterator.next();

RequestException.RequestException(ben,don);

Organization.getCurrentDonations().modify(Organization.getCurrentDonations().get(don.getEntity().getId()), Organization.getCurrentDonations().get(don.getEntity().getId()).getQuantity() - don.getQuantity());

ben.add(don);

System.out.println("Your request for " + don.getEntity().getName() + " and its quantity of " + don.getQuantity() + " has been fulfilled!");

iterator.remove();

}

}catch(CustomExceptionClass e){System.out.println(e.getMessage());}

}

}

* **CustomExceptionClass**

public class CustomExceptionClass extends Exception{

public CustomExceptionClass(String message){super (message);}

}

* **RequestDonation\_ExcStructure**

import java.util.List;

public class RequestDonation\_ExcStructure {

private static boolean check=false;

public static void RequestDonationAdd(Entity ob) throws CustomExceptionClass{

for(var x:Organization.getEntityList()){

if (ob.getId()==x.getId()){

check=true;}

}

if (check==false){

throw new CustomExceptionClass("This entity doesn't exist in the organization's archives");

}

}

}

* **ID\_Exception**

import java.util.List;

public class ID\_Exception{

public static void CheckID(Entity ob) throws CustomExceptionClass{

for(var x:Organization.getEntityList()){

if (ob.getId()==x.getId()){

throw new CustomExceptionClass("An entity with the same ID already exists!");}

}

}

}

* **Don\_Exception**

public class Don\_Exception

{

public static void CheckName(Donator ob) throws CustomExceptionClass{

for(var x:Organization.getDonatorList()){

if (ob.getName().equals(x.getName())){

throw new CustomExceptionClass("A donator with the same name already exists!");}

}

}

}

* **Ben\_Exception**

public class Ben\_Exception

{

public static void CheckName(Beneficiary ob) throws CustomExceptionClass{

for(var x:Organization.getBeneficiaryList()){

if (ob.getName().equals(x.getName())){

throw new CustomExceptionClass("A Beneficiary with the same name already exists!");}

}

}

}

* **RequestException**

import java.util.List;

public class RequestException {

public static void RequestException(Beneficiary ben,RequestDonation ob) throws CustomExceptionClass {

if(ob==null){throw new CustomExceptionClass("There are no donations for this object");}

else{

//Elegxos gia to a)

if (ob.getQuantity()>Organization.getCurrentDonations().get(ob.getEntity().getId()).getQuantity()){

throw new CustomExceptionClass("Organization doesn't have enough of the resource to complete.");

}

//Elegxos gia to b)

if (!ben.getRequestsList().validRequestDonation(ben,ob)) {

throw new CustomExceptionClass("The beneficiary's request for " + ob.getEntity().getName() + ": " + ob.getQuantity() + " is not in valid bounds");

}

}

}

}

* **Main**

import java.util.List;

import java.util.ArrayList;

public class Main {

public static void main(String args[]){

System.out.println("--------Start of Main--------");

Menu menu = new Menu();

//Ftiaxnoume to organization

Organization.setName("To Xamogelo Tou Ceida :)");

List<Entity> entityList=new ArrayList<Entity>();

Organization.setEntityList(entityList);

List<Donator> donatorList = new ArrayList<Donator>();

Organization.setDonatorList(donatorList);

List<Beneficiary> beneficiaryList = new ArrayList<Beneficiary>();

Organization.setBeneficiaryList(beneficiaryList);

RequestDonationList currentDonations = new RequestDonationList();

Organization.setCurrentDonations(currentDonations);

//listes poy pairnoyn oi beneficiary

List<RequestDonation> rdEntities = new ArrayList<RequestDonation>();

Offers offersList = new Offers();

RequestDonationList receivedList1 = new RequestDonationList();

Requests requestsList1 = new Requests();

RequestDonationList receivedList2 = new RequestDonationList();

Requests requestsList2 = new Requests();

currentDonations.setRdEntities(rdEntities);

//Eisagwgh xrhstwn

//Oi beneficiary pairnoyn (onoma,thlefwno,plhthos,tis lists me ta offer poy yparxoyn,request poy exei o beneficiary)

Admin admin1= new Admin("Miltos","00");

Organization.setAdmin(admin1);

Donator kostas = new Donator(offersList,"Kostas","11");

Organization.insertDonator(kostas);

Beneficiary giannis = new Beneficiary("Giannis","21",4,receivedList1,requestsList1);

Organization.insertBeneficiary(giannis);

Beneficiary mitsos = new Beneficiary("Mitsos","22",4,receivedList2,requestsList2);

Organization.insertBeneficiary(mitsos);

//Eisagwgh materials-services

Material sugar= new Material("Sugar","1kg bags",9991,7,8,9);

Organization.addEntity(sugar);

RequestDonation rdSugar = new RequestDonation((Entity)sugar,0);

Organization.getCurrentDonations().add(rdSugar);

Material milk= new Material("Milk","1lt. box",9990,7,8,9);

Organization.addEntity(milk);

RequestDonation rdMilk = new RequestDonation((Entity)milk,0);

Organization.getCurrentDonations().add(rdMilk);

Material rice= new Material("Rice","5 servings/package", 9992,2,5,9);

Organization.addEntity(rice);

RequestDonation rdRice = new RequestDonation((Entity)rice,0);

Organization.getCurrentDonations().add(rdRice);

Service medicalSupport = new Service("Medical Support","getting checked up by a doctor (1 visit).", 8880);

Organization.addEntity(medicalSupport);

RequestDonation rdMedicalSupport = new RequestDonation((Entity)medicalSupport, 0);

Organization.getCurrentDonations().add(rdMedicalSupport );

Service nurserySupport = new Service("Nursery Support","watching over sick people(per hour).", 8881);

Organization.addEntity(nurserySupport);

RequestDonation rdNurserySupport = new RequestDonation((Entity)nurserySupport,0);

Organization.getCurrentDonations().add(rdNurserySupport);

Service babySitting = new Service("Baby Sitting","watching over your kids(per hour).",8882);

Organization.addEntity(babySitting);

RequestDonation rdBabySitting = new RequestDonation((Entity)babySitting,0);

Organization.getCurrentDonations().add(rdBabySitting);

//Merika donations gia na einai pio eukolo to testing

RequestDonation kostasDon1 = new RequestDonation(milk,20);

kostas.add(kostasDon1);

RequestDonation kostasDon2 = new RequestDonation(sugar,10);

kostas.add(kostasDon2);

RequestDonation kostasDon3 = new RequestDonation(rice,2);

kostas.add(kostasDon3);

RequestDonation kostasDon4 = new RequestDonation(babySitting,20);

kostas.add(kostasDon4);

kostas.commit();

//Merika requests epishs gia testing

RequestDonation giannisReq5 = new RequestDonation(milk,5);

giannis.addReq(giannis,giannisReq5);

RequestDonation giannisReq2 = new RequestDonation(sugar,5);

giannis.addReq(giannis,giannisReq2);

RequestDonation mitsosReq2 = new RequestDonation(sugar,2);

mitsos.addReq(mitsos,mitsosReq2);

//commit kai ena akoma request pou den ginetai commit

giannis.commit(giannis);

RequestDonation giannisReq1 = new RequestDonation(milk,1);

giannis.addReq(giannis,giannisReq1);

System.out.println("--------End of main--------");

menu.RunMenu();

}

}

* **Menu**

import java.util.Scanner;

import java.lang.String;

public class Menu{

private String username;

private String phoneN;

private User user;

private int status;

private String input1=null;

private int input2;

private static final Scanner input = new Scanner(System.in);

private static final Scanner input3 = new Scanner(System.in);

private int counter;

private int counter2;

private int id;

private RequestDonation rd;

private Donator donator;

private Beneficiary beneficiary;

private double amount;

private boolean choice;

public void RunMenu(){

while (1==1){

printHeader();

printAuthentication();

//periptwsh pou den einai registered

while(status==0){

register();

}

//donator menu

while(status==1){

printDonMenu();

performDonAction(chooseAction(1,5));

}

//admin menu

while(status==2){

printAdminMenu();

int AdminChoice=chooseAction(1,4);

performAdminAction(AdminChoice);

}

//beneficiary menu

while(status==3){

printBenMenu();

int BenChoice=chooseAction(1,5);

performBenAction(BenChoice);

}

}

}

private void printHeader(){

System.out.println("+---------------------------------------+");

System.out.println("| Welcome to our |");

System.out.println("| Menu Application |");

System.out.println("+---------------------------------------+");

}

private void printAuthentication(){

System.out.println(" Authentication required...");

System.out.println("Please type your phone number: ");

phoneN=input.nextLine();

status=0;

//Anagnwrhsh tupou tou xrhsth (Donator, Beneficiary h Admin)

for(var usCheck : Organization.getDonatorList()){

if (usCheck.getPhone().equals(phoneN)){

user=(User)usCheck;

donator=usCheck;

status=1;

break;

}

}

//Kanoume if an status==0 wste na mhn ama einai Donator (h meta admin) na mhn kanoume extra anazhthseis.

if (status==0){

if (Organization.getAdmin().getPhone().equals(phoneN)){

user=(User)Organization.getAdmin();

status=2;

}

}

if (status==0){

for(var usCheck : Organization.getBeneficiaryList()){

if (usCheck.getPhone().equals(phoneN)){

user=(User)usCheck;

beneficiary=usCheck;

status=3;

break;

}

}

}

}

private void register(){

boolean choice0;

System.out.println("You are not a registered user, would you like to register? (y/n)");

input2=readAnswer();

if (input2 =='n'){

System.exit(0);

}

else if(input2 == 'y'){

System.out.println("Would you like to register as a donator or as a beneficiary? \nPress 1 for donator and 2 for beneficiary. ");

input2 = chooseAction(1,2);

if (input2 == 1){

choice0=false;

while(!choice0) {

System.out.println("What is your name? ");

username = input3.nextLine();

//Dhmiourgeia neou donator

Offers offersList = new Offers();

Donator newDon = new Donator(offersList, username, phoneN);

if (Organization.insertDonator(newDon)) {

donator = newDon;

status = 1;

choice0 = true;

}

}

}

else if(input2 == 2){

choice0=false;

while(!choice0) {

System.out.println("What is your name? ");

username = input3.nextLine();

//Dhmiourgeia neou beneficiary

RequestDonationList receivedListTemp = new RequestDonationList();

Requests requestsListTemp = new Requests();

System.out.println("How many persons are in your family group? ");

counter = chooseAction(1,20);

Beneficiary newBen = new Beneficiary(username, phoneN, counter, receivedListTemp, requestsListTemp);

if (Organization.insertBeneficiary(newBen)) {

beneficiary = newBen;

status = 3;

choice0=true;

}

}

}

}

}

//Menu gia kathe eidos xrhsth!!!

private void printDonMenu(){

System.out.println("Welcome "+donator.getName()+", Phone : "+user.getPhone()+" Donator to ||"+Organization.getName()+"||");

System.out.println("Please choose your action:");

System.out.println("1)Add Offer");

System.out.println("2)Show Offers");

System.out.println("3)Commit on your donations");

System.out.println("4)Log out");

System.out.println("5)Exit application");

}

private void printAdminMenu(){

System.out.println("Welcome "+user.getName()+", Phone : "+user.getPhone()+" Admin to ||"+Organization.getName()+"||");

System.out.println("Please choose your action:");

System.out.println("1)View");

System.out.println("2)Monitor Organization");

System.out.println("3)Log out");

System.out.println("4)Exit application");

}

private void printBenMenu(){

System.out.println("Welcome "+beneficiary.getName()+", Phone : "+user.getPhone()+" Beneficiary to ||"+Organization.getName()+"||");

System.out.println("Please choose your action:");

System.out.println("1)Add Request");

System.out.println("2)Show Requests");

System.out.println("3)Commit on your requests");

System.out.println("4)Log out");

System.out.println("5)Exit application");

}

//Actions gia kathe eidos xrhsth

private void performAdminAction(int AdminChoice){

boolean choice0=false;

boolean choice1=false;

boolean choice2=false;

boolean choice3=false;

while(!choice0){

switch(AdminChoice){

case 1://View

System.out.println("Choose the type of entity:\n1) Materials\n2) Services\n3) Back");

switch(chooseAction(1,3)){

case 1://Materials

//tupwnei kai diavazei material

printMatSer("Material");

choice2=getMatSer("Material",counter);

System.out.println(choice2);

if(choice2){choice0=true;}//periptwsh pou epelekse na paei pisw

if(!choice2) {

System.out.println(rd.getEntity().getName() + " - " + rd.getEntity().getDetails() + "\nCurrent quantity - " + rd.getQuantity());

}

break;

case 2://Services

//tupwnei kai diavazei service

printMatSer("Service");

choice2=getMatSer("Service",counter);

if(choice2){choice0=true;}//periptwsh pou epelekse na paei pisw

if(!choice2) {

System.out.println(rd.getEntity().getName() + " - " + rd.getEntity().getDetails() + "\nCurrent quantity - " + rd.getQuantity());

}

choice2=true;

break;

case 3://Back

choice0=true;

break;

default:

System.out.println("An unknown error has occured.");

System.exit(0);

break;

}

break;

case 2://Monitor Organization

System.out.println("Please choose an action\n1)List Beneficiaries\n2)List Donators\n3)Reset Beneficiaries\n4)Back");

switch(chooseAction(1,4)){

case 1://List Beneficiaries

//afou thelei arithmimenh lista kai meta thelei na epileksoume beneficiary, h listBeneficiaries() ths Organization den mas arkei.

counter=0;

for(var x:Organization.getBeneficiaryList()){

counter+=1;

System.out.println(counter+")"+x.getName());

}

//dialegei beneficiary

System.out.println("Please choose a beneficiary (Press "+(counter+1)+" to go back.)");

input2=chooseAction(1,counter+1);

if(input2==counter+1){choice0=true;}

if(input2!=counter+1) {

counter = 0;

for (var x : Organization.getBeneficiaryList()) {

counter += 1;

if (counter == input2) {

beneficiary = x;

break;

}

}

choice3 = false;

while (choice3 == false) {

System.out.println("\nPlease choose an action\n1)Print received list\n2)Clear received list\n3)Delete Beneficiary\n4)Back");

switch (chooseAction(1, 4)) {

case 1://Print received

beneficiary.getReceivedList().monitor();

break;

case 2://Clear received

beneficiary.getReceivedList().reset();

System.out.println("Beneficiary's received list was reset!");

break;

case 3://Delete

Organization.removeBeneficiary(beneficiary);

System.out.println("Beneficiary was removed!");

break;

case 4://Back

choice3 = true;

break;

}

}

}

break;

case 2://List Donators

counter=0;

for(var x:Organization.getDonatorList()){

counter+=1;

System.out.println(counter+")"+x.getName());

}

System.out.println("Please choose a donator (Press "+(counter+1)+" to go back.)");

input2=chooseAction(1,counter+1);

if(input2==counter+1){choice0=true;}

if(input2!=counter+1) {

counter = 0;

for (var x : Organization.getDonatorList()) {

counter += 1;

if (counter == input2) {

donator = x;

break;

}

}

choice3 = false;

while (choice3 == false) {

System.out.println("Please choose an action\n1)Print offers\n2)Delete donator\n3)Back");

switch (chooseAction(1, 4)) {

case 1://Print offers

donator.getOffersList().monitor();

break;

case 2://Delete donator

Organization.removeDonator(donator);

System.out.println("Donator was removed!");

break;

case 3://Back

choice3 = true;

break;

}

}

}

break;

case 3://Reset Beneficiaries

for(var x:Organization.getBeneficiaryList()){

x.getReceivedList().reset();

}

System.out.println("Beneficiary received lists were all reset!");

break;

case 4://Back

choice0=true;

break;

default:

System.out.println("An unknown error has occured.");

break;

}

break;

case 3://Log out

status=0;

choice0=true;

break;

case 4://Exit

input.close();

System.out.println("Thank you for using our application.");

System.exit(0);

break;

default:

System.out.println("An unknown error has occured.");

System.exit(0);

break;

}

}

}

private void performBenAction(int BenChoice) {

boolean checker=false;

boolean choice0=false;

boolean choice1;

boolean choice2;

choice = false;

while(choice0==false){

switch(BenChoice){

case 1://Add Request

counter=0;

input1=null;

input2=0;

System.out.println("What do you want to request?\n1)Material\n2)Service\n3)Back");

//Periptwsh epiloghs 1)Material

switch(chooseAction(1,3)){

case 1://1) Material

choice1=false;

while(!choice1){

//Tupwsh twn materials

printMatSer("Material");

//Epilogh material

choice2=getMatSer("Material",counter);

if(choice2==true){break;}

//Epilogh prosforas

input1=null;

System.out.println(rd.getEntity().getName()+"\n"+"Description: "+rd.getEntity().getDescription()+"\nCurrent quantity:"+rd.getQuantity()+"\n"+rd.getEntity().getDetails());

System.out.println("\nDo you want to request this Material? (y/n)");

input2=readAnswer();

if (input2=='y')

{

System.out.println("Please insert the amount you want to request");

amount=readDouble();

RequestDonation rd2 = new RequestDonation(rd.getEntity(),amount);

beneficiary.addReq(beneficiary,rd2);

break;

}

else if (input2=='n'){choice1=true;}

}

break;

case 2://1) Service

choice1=false;

while(!choice1){

//Tupwsh twn Services

printMatSer("Service");

//Epilogh Service

choice2=getMatSer("Service",counter);

if(choice2==true){break;}

//Epilogh prosforas

System.out.println(rd.getEntity().getName()+"\n"+"Description: "+rd.getEntity().getDescription()+"\nCurrent quantity:"+rd.getQuantity()+rd.getEntity().getDetails());

System.out.println("\nDo you want to request this service? (y/n)");

input2=readAnswer();

if (input2=='y')

{

System.out.println("Please insert the amount you want to request:");

amount=readDouble();

RequestDonation rd2 = new RequestDonation(rd.getEntity(),amount);

beneficiary.addReq(beneficiary,rd2);

break;

}

else if (input2=='n'){choice1=true;}

}

break;

case 3://3)Back

choice1=true;

choice0=true;

}

break;

case 2: //2)Show Requests

choice1=false;

counter2 = 0;

while(!choice1) {

counter2 = 0;

int tempcounter=0;

for (var y : beneficiary.getRequestsList().getlist()) {//tupwnei thn lista requestsList

counter2=counter2+1;

if (counter2 > 0) {

System.out.println(counter2 + ")" + y.getEntity().getName() + " - Quantity: " + y.getQuantity());

tempcounter=counter2;

}

}

if (tempcounter == 0) {//periptwsh pou den uparxoun requests

System.out.println("You have no active requests.");

choice1=false;

choice0=false;

}

System.out.print("\n");

//Epilogh action

System.out.println("What would you like to do?\n1)Choose a request.\n2)Clear requests.\n3)Commit requests.\n4)Back");

switch (chooseAction(1, 4)) {

case 1://1) Choose a request

counter2 = 0;

System.out.println("Which request do you want to choose?(Press "+(tempcounter+1)+" to go back)");

input2 = chooseAction(1, tempcounter + 1);

for (var x : beneficiary.getRequestsList().getlist()) { //Anazhthsh sthn requestList gia to request epiloghs

counter2 += 1;

if (counter2 == input2) {

System.out.println("Which action do you want to choose?\n1)Delete request\n2)Modify request\n3)Back");

switch (chooseAction(1, 3)) {

case 1://Delete request

beneficiary.getRequestsList().remove(x);

System.out.println("Succesfully removed!");

break;

case 2://Modify request

System.out.println("Please insert new amount");

amount=readDouble();

beneficiary.modifyReq(beneficiary,x,amount);

break;

case 3://Back

break;

}

break; //kanei break thn for

}

}

if (input2 == tempcounter + 1) {

choice1 = true;

}

break;

case 2://2) Clear Requests

beneficiary.getRequestsList().reset();

System.out.println("Requests cleared successfully");

choice1=true;

choice0=true;

break;

case 3://3) Commit Requests

beneficiary.getRequestsList().commit(beneficiary);

System.out.println("Requests commited successfully");

choice1=true;

choice0=true;

break;

case 4://4) Back

choice1 = true;

choice0=true;

break;

}

}

break;

case 3://Commit requests

beneficiary.commit(beneficiary); //ta munimata ektupwnontai mesw tis commit() ths Requests.

choice0=true;

break;

case 4://Logout

status = 0;

choice0=true;

break;

case 5://Exit

input.close();

System.out.println("Thank you for using our application.");

System.exit(0);

break;

default:

System.out.println("An unknown error has occured.");

System.exit(0);

break;

}

}

}

private void performDonAction(int DonChoice){

boolean choice0=false;

boolean choice1;

boolean choice2;

while (!choice0){

switch(DonChoice){

case 1://Add Offer

choice1=false;

while(!choice1) {

counter = 0;

input1 = null;

input2 = 0;

System.out.println("What do you want to donate?\n1)Material\n2)Service\n3)Back");

//Periptwsh epiloghs 1)Material

switch (chooseAction(1, 3)) {

case 1://1) Material

//Tupwsh twn materials

printMatSer("Material");

//Epilogh material

choice2 = getMatSer("Material", counter);

if (choice2) {

break;

}

//Epilogh prosforas

input1 = null;

System.out.println(rd.getEntity().getName() + "\n" + "Description: " + rd.getEntity().getDescription() + "\nCurrent quantity:" + rd.getQuantity() + "\n" + rd.getEntity().getDetails());

System.out.println("\nDo you want to donate this Material?(y/n)");

input2 = readAnswer();

if (input2 == 'y') {

System.out.println("Please insert the amount you want to donate");

amount=readDouble();

RequestDonation rd2 = new RequestDonation(rd.getEntity(),amount);

donator.getOffersList().add(rd2);

choice1 =true;

choice0 = true;

break;

}

break;

case 2://2)Service

//Tupwsh twn Services

printMatSer("Service");

//Epilogh Service

choice2 = getMatSer("Service", counter);

if (choice2) {

break;

}

//Epilogh prosforas

input1 = null;

System.out.println(rd.getEntity().getName() + "\n" + "Description: " + rd.getEntity().getDescription() + "\nCurrent quantity:" + rd.getQuantity());

System.out.println("\nDo you want to donate this Service?(y/n)");

input2 = readAnswer();

if (input2 == 'y') {

System.out.println("Please insert the amount of hours you want to donate");

amount=readDouble();

RequestDonation rd2 = new RequestDonation(rd.getEntity(),amount);

donator.getOffersList().add(rd2);

donator.getOffersList().add(rd2);

choice1 =true;

choice0 = true;

break;

}

break;

case 3://3)Back

choice1 =true;

choice0 = true;

break;

}

}

break;

case 2: //2)Show Offers

choice1=false;

while(!choice1){

counter2=0;

for(var y:donator.getOffersList().getlist()){

counter2+=1;

if(counter2>0){System.out.println(counter2+")"+y.getEntity().getName()+" - Quantity: "+y.getQuantity());}

}

if(counter2==0){

System.out.println("You have no active offers.");

choice1=true;

choice0=true;

break;

}

if(counter2>0) {

//Epilogh action

System.out.println("What would you like to do?\n1)Choose an offer.\n2)Clear Offers.\n3)Commit Offers.\n4)Back");

choice2=false;

while (!choice2) {

switch (chooseAction(1, 4)) {

case 1://1) Choose an offer

counter2 = 0;

System.out.println("Which offer do you want to choose?(Press "+(counter2+1)+" to go back)");

input2 = chooseAction(1, counter2 + 1);

for (var x : donator.getOffersList().getlist()) { //Anazhthsh sthn OffersList gia to offer epiloghs

counter2 += 1;

if (counter2 == input2) {

System.out.println("Which action do you want to choose?\n1)Delete offer\n2)Modify offer\n3)Back");

switch (chooseAction(1, 3)) {

case 1://Delete offer

donator.getOffersList().remove(x);

choice2=true;

break;

case 2://Modify offer

System.out.println("Please insert new amount");

amount=readDouble();

donator.getOffersList().modify(x, amount);

choice2=true;

break;

case 3://Back

choice2=true;

break;

}

break; //kanei break thn for

}

}

break;

case 2://2) Clear Offers

donator.getOffersList().reset();

System.out.println("Offers cleared successfully");

choice1=true;

break;

case 3://3) Commit Offers

donator.getOffersList().commit();

System.out.println("Offers commited successfully");

choice1=true;

break;

case 4://4) Back

choice1=true;

choice2 = true;

break;

}

}

}

choice2 = true;

choice1=true;

choice0=true;

}

break;

case 3://3) Commit offers

donator.getOffersList().commit();

System.out.println("Offers commited successfully");

choice0=true;

break;

case 4://4) Logout

status=0;

choice0=true;

break;

case 5://5) Exit

input.close();

System.out.println("Thank you for using our application.");

System.exit(0);

break;

default:

System.out.println("An unknown error has occured.");

System.exit(0);

break;

}

}

}

//VOHTHITIKES METHODOI

//kanei print lista services h materials tou organismou analoga me to String str

private void printMatSer(String str){

counter=0;

for(var ent:Organization.getEntityList()) {

if (ent.getDetails().indexOf(str)!=-1){

counter+=1;

if(Organization.getCurrentDonations().get(ent.getId())==null){

System.out.println(counter+") "+str+" - "+ent.getName()+" (Amount: "+"0)");}

else{System.out.println(counter+") "+str+" - "+ent.getName()+" (Amount: "+Organization.getCurrentDonations().get(ent.getId()).getQuantity()+")");}

}

}

}

//Pairnei ena sugkekrimeno material h service apthn lista analoga me to String str kai me auto kanoume initialize to rd

private boolean getMatSer(String str,int counter){

boolean check=false;

System.out.println("Which "+str+" do you want to select?(Press "+(counter+1)+" to go back.)");

input2=chooseAction(1,counter+1);

if(counter+1==input2){check=true;}

//Anazhthsh sthn lista gia to item epiloghs

counter=0;

if(!check) {

for (var ent : Organization.getEntityList()) {

if (ent.getDetails().indexOf(str) != -1) {

counter += 1;

if (counter == input2) {

id = ent.getId();

rd = Organization.getCurrentDonations().get(id);

break;

}

}

}

}

return check;

}

//diavazei y/n

private int readAnswer(){

int answer=0;

while(answer!='y' && answer!='n'){

answer = input.next().charAt(0);

input.nextLine();

if(answer!='y' && answer!='n'){System.out.println("Wrong input, please try again");}

}

return answer;

}

//diavazei arithmo apo to min mexri to max

private int chooseAction(int min, int max){

int Action=-1;

while(Action<min || Action>max){

try{

Action= Integer.parseInt(input.nextLine());

if (Action<min || Action>max){System.out.println("Wrong input, please try again");}

}catch(NumberFormatException ea){System.out.println("Invalid selection");}

}

return Action;

}

private double readDouble(){

double amount=0;

while(amount<=0){

try {

amount = Double.parseDouble(input.nextLine());

if (amount <= 0) { System.out.println("Wrong input, please try again"); }

}catch(NumberFormatException ea){System.out.println("Invalid selection");}

}

return amount;

}

}