Міністерство освіти і науки, молоді та спорту України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



Розрахункова робота

З дисципліни

“Практичне програмування”

Виконав:

Слівінський М.Р.

Група: КН-203

Перевірив:

Скорохода О.В.

**Львів 2019**

**11 – варіант**

Страхування. Визначити ієрархію страхових зобов’язань. Зібрати із зобов’язань

дериватив. Підрахувати вартість. Здійснити сортування зобов’язань у деривативі на основі зменшення рівня ризику. Знайти зобов’язання у деривативі, що відповідає заданому діапазону параметрів.

**Код програми: (Main.java)**

import Commands.\*;

import Saver.\*;

import java.util.Scanner;

public class Main {

public static void main(String argsp[]) {

Controller controller = new Controller();

Derevatuv.add(new UsualVehicle(5));

Derevatuv.add(new Invenstution(10));

Derevatuv.add(new UnfortunatelyCase(2));

Derevatuv.add(new UsualVehicle(3));

Derevatuv.add(new LifeSaver(1));

Derevatuv.add(new HealthAndSick(3));

while (true) {

System.out.println("1 - показати дереватив\n2 - вивести ціну\n3 - відсортувати\n4 - знайти по параметру чисел\n5 - знайти по параметру імені");

Scanner scanner = new Scanner(System.in);

int c = scanner.nextInt();

if (c == 1) {

System.out.println("Deruvatuv ");

Derevatuv.show();

System.out.println();

}

if (c == 2) {

System.out.println();

controller.setCommand(new Price());

System.out.println();

}

if (c == 3) {

controller.setCommand(new Sort());

System.out.println();

}

if (c == 4) {

System.out.println("Веддіть мінімальне");

int n1 = scanner.nextInt();

System.out.println("Введіть максимальне значення");

int n2 = scanner.nextInt();

System.out.println();

controller.setCommand(new FindByParametr(n1, n2));

System.out.println();

}

if (c == 5) {

System.out.println();

controller.setCommand(new FindByParametr("document for save from damage from usual vehicle"));

System.out.println();

}

}

}

}

**Код (Derevatuv.java)**

package Commands;

import Logger.Logger;

import Saver.Saver;

import java.util.ArrayList;

import java.util.List;

public class Derevatuv {

static List<Saver> savers = new ArrayList();

public Derevatuv() {

}

public static void add(Saver t){

savers.add(t);

Logger.log("В дереватив було додано " + t.toString());

}

public static void delete(int t){

try {

Saver saver = savers.get(t);

savers.remove(t);

Logger.log("З дереватива було видалено " + saver);

}catch (IndexOutOfBoundsException e){

Logger.error("В деревативі не існує такого елемента");

}

}

public static void show(){

savers.stream().forEach(e -> System.out.println(e));

Logger.log("Був виведений список");

}

public static ArrayList<Saver> getSaver(){

return (ArrayList<Saver>) savers;

}

public static void setSavers(ArrayList<Saver> toSet){

savers = toSet;

Logger.log("Дереватив було оновлено на: ");

savers.stream().forEach(e -> Logger.log(" " + e.toString()));

}

public static void deleteAll(){

savers = new ArrayList<>();

Logger.log("Дереватив було видалено");

}

}**Код (CommandBase.java)**

package CommandPattern;  
  
public interface CommandBase {  
 public void execute();  
}

**Код (Controller.java)**

package Commands;

public class Controller {

CommandBase command;

public Controller()

{

}

public void setCommand(CommandBase command)

{

this.command = command;

buttonWasPressed();

}

public void buttonWasPressed()

{

command.execute();

}

}

**Код (FindByParametr.java)**

package Commands;

import Logger.Logger;

public class FindByParametr implements CommandBase {

private int countMin;

private int countMax;

private String name;

@Override

public void execute() {

if(countMax != 0 && countMin != 0){

System.out.println("Find save between " + countMin + " and " + countMax);

Derevatuv.getSaver().stream().filter(x -> x.getPrice() > countMin && x.getPrice() < countMax).

forEach(x -> System.out.println(x.toString()));

Logger.log("Був виведений список, ціна між " + countMin + " і " + countMax);

}else {

System.out.println("Find by " + name);

Derevatuv.getSaver().stream().filter(x -> x.getName().equals(name)).

forEach(x -> System.out.println(x.toString()));

Logger.log("Був виведений список, елементи якого називаються " + name);

}

}

public FindByParametr(int countMin, int countMax) {

this.countMin = countMin;

this.countMax = countMax;

}

public FindByParametr(String name) {

this.name = name;

}

}

**Код (Price.java)**

package Commands;

import Logger.Logger;

import Saver.Saver;

public class Price implements CommandBase {

@Override

public void execute() {

int count = 0;

for (Saver s : Derevatuv.getSaver()) {

count += s.getPrice();

}

System.out.println("Price of derevatuv is " + count);

Logger.log("Була знайдена ціна всього дереватива " + count);

}

}

**Код (Sort.java)**

package Commands;

import Logger.Logger;

import java.util.Collections;

public class Sort implements CommandBase {

@Override

public void execute() {

Collections.sort(Derevatuv.getSaver());

Derevatuv.setSavers(Derevatuv.getSaver());

Logger.log("Дереватив було відсортовано до зниженю урону");

}

}

**Код (Saver.java)**

package Saver;

public interface Saver extends Comparable<Saver>{

int getLevelDamage();

int getPrice();

String getName();

@Override

public default int compareTo(Saver o) {

return o.getLevelDamage() - this.getLevelDamage();

}

}

**Код (HealthAndSick.java)**

package Saver;

import java.util.Objects;

public class HealthAndSick implements Saver{

static private final String NAME = "document for save from problem with health";

private final int levelDamage = 4;

private int price;

private int count;

public HealthAndSick(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

public int getCount() {

return count;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public void setCount(int count) {

this.count = count;

}

@Override

public String getName() {

return NAME;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

HealthAndSick that = (HealthAndSick) o;

return levelDamage == that.levelDamage &&

price == that.price &&

count == that.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "HealthAndSick{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (Invenstution.java)**

package Saver;

import java.util.Objects;

public class Invenstution implements Saver{

static private final String NAME = "document for save from bad investution";

private final int levelDamage = 3;

private int price;

private int count;

public Invenstution(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

@Override

public String getName() {

return NAME;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Invenstution that = (Invenstution) o;

return levelDamage == that.levelDamage &&

price == that.price &&

count == that.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "Invenstution{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (LifeSaver.java)**

package Saver;

import java.util.Objects;

public class LifeSaver implements Saver{

static private final String NAME = "document for save Life";

private final int levelDamage = 6;

private int price;

private int count;

public LifeSaver(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

public LifeSaver() {

}

@Override

public String getName() {

return NAME;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

LifeSaver lifeSaver = (LifeSaver) o;

return levelDamage == lifeSaver.levelDamage &&

price == lifeSaver.price &&

count == lifeSaver.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "LifeSaver{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (UnfortunatelyCase.java)**

package Saver;

import java.util.Objects;

public class UnfortunatelyCase implements Saver{

static private final String NAME = "document for save from unfortunately case";

private final int levelDamage = 5;

private int price;

private int count;

public UnfortunatelyCase(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

@Override

public String getName() {

return NAME;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

UnfortunatelyCase that = (UnfortunatelyCase) o;

return levelDamage == that.levelDamage &&

price == that.price &&

count == that.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "UnfortunatelyCase{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (UnusualVehicle.java)**

package Saver;

import java.util.Objects;

public class UnusualVehicle implements Saver {

static private final String NAME = "document for save from damage from unusual vehicle";

private final int levelDamage = 2;

private int price;

private int count;

public UnusualVehicle(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

public UnusualVehicle() {

}

@Override

public String getName() {

return NAME;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

UnusualVehicle that = (UnusualVehicle) o;

return levelDamage == that.levelDamage &&

price == that.price &&

count == that.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "UnusualVehicle{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (UsualVehicle.java)**

package Saver;

import java.util.Objects;

public class UsualVehicle implements Saver{

static private final String NAME = "document for save from damage from usual vehicle";

private final int levelDamage = 1;

private int price;

private int count;

public UsualVehicle(int count) {

this.count = count;

this.price = count \* levelDamage \* 10;

}

public UsualVehicle() {

}

@Override

public String getName() {

return NAME;

}

@Override

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

@Override

public int getLevelDamage() {

return levelDamage;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

UsualVehicle that = (UsualVehicle) o;

return levelDamage == that.levelDamage &&

price == that.price &&

count == that.count;

}

@Override

public int hashCode() {

return Objects.hash(levelDamage, price, count);

}

@Override

public String toString() {

return "UsualVehicle{" +

"levelDamage=" + levelDamage +

", price=" + price +

", count=" + count +

'}';

}

}

**Код (EmailSender.java)**

package Logger;

import com.sun.xml.internal.messaging.saaj.packaging.mime.MessagingException;

import sun.plugin2.message.Message;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

import java.util.Properties;

public class EmailSender {

public static void sendEmail(String errorMessage) throws MessagingException {

final String username = "mykhailo.slivinskyi.knm.2018@lpnu.ua";

final String password = "08.06.2001";

Properties prop = new Properties();

prop.put("mail.smtp.host", "smtp.gmail.com");

prop.put("mail.smtp.port", "587");

prop.put("mail.smtp.auth", "true");

prop.put("mail.smtp.starttls.enable", "true"); //TLS

try {

Session session = Session.getInstance(prop,

new javax.mail.Authenticator() {

protected javax.mail.PasswordAuthentication getPasswordAuthentication() {

return new javax.mail.PasswordAuthentication(username, password);

}

});

MimeMessage message = new MimeMessage(session);

message.setFrom(new InternetAddress("mykhailo.slivinskyi.knm.2018@lpnu.ua"));

message.setRecipients(MimeMessage.RecipientType.TO, InternetAddress.parse("miwka0806@gmail.com"));

message.setSubject("Exception thrown at laba 8");

message.setText(errorMessage);

Transport.send(message);

} catch (javax.mail.MessagingException e) {

e.printStackTrace();

}

}

}

**Код (Logger.java)**

package Logger;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Arrays;

public class Logger {

public static boolean log(String any){

try

{

String filename= "D:/Laba4/logs.txt";

FileWriter fw = new FileWriter(filename,true);

fw.write(any + "\n");

fw.close();

return true;

}

catch(IOException ioe)

{

Logger.error(ioe.getMessage() + "\n" + Arrays.toString(ioe.getStackTrace()));

return false;

}

}

public static boolean error(String err) {

err+="\n";

try

{

String filename= "D:/Laba4/logs.txt";

FileWriter fw = new FileWriter(filename,true);

fw.write(err);

fw.close();

try {

EmailSender.sendEmail(err);

} catch (com.sun.xml.internal.messaging.saaj.packaging.mime.MessagingException e) {

e.printStackTrace();

}

return true;

}

catch(IOException e)

{

return false;

//Logger.error(ioe.getMessage() + "\n" + Arrays.toString(ioe.getStackTrace()));

}

}

}

Unit Test

package Commands;

import Saver.\*;

import org.junit.Assert;

import org.junit.Test;

import static org.junit.Assert.\*;

public class DerevatuvTest {

@Test

public void add() {

Derevatuv.add(new UsualVehicle(5));

Assert.assertEquals(new UsualVehicle(5), Derevatuv.getSaver().get(0));

}

@Test

public void delete() {

Derevatuv.add(new UsualVehicle(5));

Derevatuv.add(new LifeSaver(2));

Derevatuv.delete(1);

Derevatuv.add(new UsualVehicle(2));

Assert.assertNotNull(Derevatuv.getSaver().get(1));

}

@Test

public void deleteAll() {

Derevatuv.add(new UsualVehicle(5));

Derevatuv.add(new LifeSaver(2));

Derevatuv.deleteAll();

Assert.assertEquals(0, Derevatuv.getSaver().size());

}

}

package Commands;

import Saver.\*;

import org.junit.Assert;

import org.junit.Test;

import static org.junit.Assert.\*;

public class SortTest {

@Test

public void execute() {

Derevatuv.deleteAll();

Derevatuv.add(new Invenstution(10));

Derevatuv.add(new UnfortunatelyCase(2));

Derevatuv.add(new UsualVehicle(3));

Derevatuv.add(new LifeSaver(1));

Derevatuv.add(new HealthAndSick(3));

Sort sort = new Sort();

Derevatuv.getSaver().stream().forEach(e -> System.out.println(e.toString()));

sort.execute();

Assert.assertEquals(new UsualVehicle(3), Derevatuv.getSaver().get(4));

}

}

package Saver;

import org.junit.Assert;

import org.junit.Test;

import static org.junit.Assert.\*;

public class LifeSaverTest {

LifeSaver lifeSaver = new LifeSaver(5);

@Test

public void getPrice() {

Assert.assertEquals(300, lifeSaver.getPrice());

}

}

**Вивід**

****

