Міністерство освіти і науки України
Харківський національний університет ім. В. Н. Каразіна
Факультет комп’ютерних наук

Курсова робота

з дисципліни «Об’єктно-орієнтоване програмування»

Тема: «Розробка прототипу програмної системи управління реєстром автомобілів (модифікація 1.3)»

Оцінка _____ / _____

Члени комісії:

_____ Поклонський Є.В.

_____ Нагорний К.А.

_____ Богучарський С.І.

Виконав:
студент 2 курсу, групи КС- 21
Спеціальності:
122 «Комп’ютерні науки»
Безрук Юрій Русланович

Харків – 2019

ЗМІСТ

ВСТУП

Метою даної курсової роботи є створення прототипу програмної системи на базі основних принципів об’єктно-орієнтованого дизайну та сучасних практик програмування із використанням об’єктно-орієнтованої мови програмування Java.

В цій роботі розглядається прототип програмної системи (ПС) управління автомобільним салоном, що надає можливість користувачу маніпулювати реєстром транспортних засобів. Програма створена таким чином, що в результаті створення ПС ми будемо мати просту систему зберігання та обробки даних конкретної галузі користування (а саме даних автосалону), тож її можна вважати тренінгом, до нашої майбутньої професії.

Безпосередня розробка програмної системи проходила в IDE Eclipse, де знаходиться інтерфейс користувача – консоль, у якій реалізовано меню для зручнішого користування функціями реєстру.

Створення діаграм класів у нотації UML виконувалося за допомогою програми Enterprice Architect.

Створення та форматування пояснювальної записки до курсової роботи проходило в текстовому редакторі MS Word, через його зручний інтерфейс для редагування тексту.

Пояснювальна записка складається з таких основних частин: концепція програмної системи, аналіз поставленої задачі та вихідного формату файлу, загальний опис архітектури та класів предметної області (автомобільного салону), наведення результатів тестування окремих класів та програми загалом, інструкція для користувача.

Особливу увагу варто звернути на основні можливості та обмеження програми (див. 1.2) , та уважно ознайомитись з інструкцією для використання кінцевого продукту.

РОЗДІЛ 1 КОНЦЕПЦІЯ

1.1 Загальна характеристика продукту

Ціль моєї роботи – створити програмний продукт для зручного користування і упорядкування реєстру у предметній області проекту.

1.2 Основні можливості та обмеження

1. Додати транспортний засіб до реєстру;
2. Видалити транспортний засіб із реєстру;
3. Отримати список усіх транспортних засобів, що містяться у реєстрі;
4. Отримати список усіх транспортних засобів, тип палива яких відповідає заданому;
5. Продати обране авто покупцеві магазину, при цьому транспортний засіб видаляється із реєстру та записується ціна продажу;
6. Роздрукувати суму продажів за «день»
7. Зберегти реєстр до сховища (текстовий файл);
8. Отримати реєстр із сховища (текстового файлу).

Основні принципи та конструкції, що мають бути реалізовані:

1. Агрегація. Магазин – сутність, яка містить колекцію видань.
2. Наслідування. Автомобіль та Мотоцикл повинні мати базову абстракцію Транспортний Засіб (Vehicle);
3. Поліморфізм. Кожний Транспортний Засіб має поліморфний метод getType, який використовується під час вибірки та відображення тварини за типом (п.4, п.5 основних можливостей системи).
4. Типи палива транспортних засобів є фіксованою множиною та мають належати до перерахування (Enumeration): бензин, дизель, електрика, ядерний реактор.
5. Асоціативна сутність Продаж (Транспортний Засіб, Покупець, Ціна), для реалізації продажу транспортного засобу покупцеві магазину.

РОЗДІЛ 2 АНАЛІЗ

2.1 Склад системи

У комплект поставки входять файли Database.jar та database.csv.

Database.jar – Java-архів. Основне завдання цього архіву - зберігати файли з класами. А database.csv – текстовий табличний файл, який виступає в ролі контейнеру для збереження інформації реєстру транспорту.

2.2 Вимоги до вхідних даних

Програма приймає на вхід табличний файл “database.csv”, як початкові дані для реєстру транспорту, звідки зчитує інформацію (при запуску програми виконується автоматично) та куди записує інформацію (при завершенні програми, або за побажанням користувача). У цьому файлі зберігається інформація про транспорт різних типів, які розділені комами (між елементами одного об`єкту) та переносами ряду (між об`єктами. Типи, що зберігаються: Авто (*Avto*) та мотоцикл (*Moto*). Приклад форматування цього файлу надано у додатку А.

У самій програмі використовується валідація вхідних даних, тобто при введенні користувачем даних несумісного типу для певних запитів, програма попередить про це, і надасть можливість зробити повторне введення.

2.3 Основні вимоги програмної системи

Інтерфейсом користувача виступає консоль у яку має бути виведений список усіх можливих команд.

Програмна система має працювати до введення команди виходу із системи (останній пункт меню).

Перед завершенням роботи програмної системи, реєстр зберігається у постійному сховищі (persistence storage), у форматі CSV (coma separated values).

РОЗДІЛ 3 ПРОЕКТУВАННЯ

3.1 Опис предметної області

Предметна область цієї програми: автомобільний салон.

Основні сутності предметної області:

1. Магазин. Має назву, адресу, містить реєстр авто;
2. Автомобіль – транспортний засіб, який має назву, рік виробництва, модель, та виробника. Транспортний засіб має відповідний тип палива, та ємність паливного сховища. Авто в магазині характеризується ціною, за яку воно може бути куплена.
3. Мотоцикл – транспортний засіб, який має назву, рік виробництва, модель, та виробника. Транспортний засіб має відповідний тип палива, та ємність паливного сховища. Авто в магазині характеризується ціною, за яку воно може бути куплена.
4. Модель – характеризується назвою та номером.
5. Покупець магазину – людина, яка має ім’я та прізвище, рік народження та ідентифікаційний код.

3.2 Класи предметної області

Програма містить такі класи:

1. *Main;*
2. *Shop*;
3. *Client*;
4. *Vehicle*;
5. *Type*;
6. *Model*;
7. *Avto*;
8. *Moto*;
9. *PetrolType;*
10. *CSV*.

3.3 UML-діаграма класів

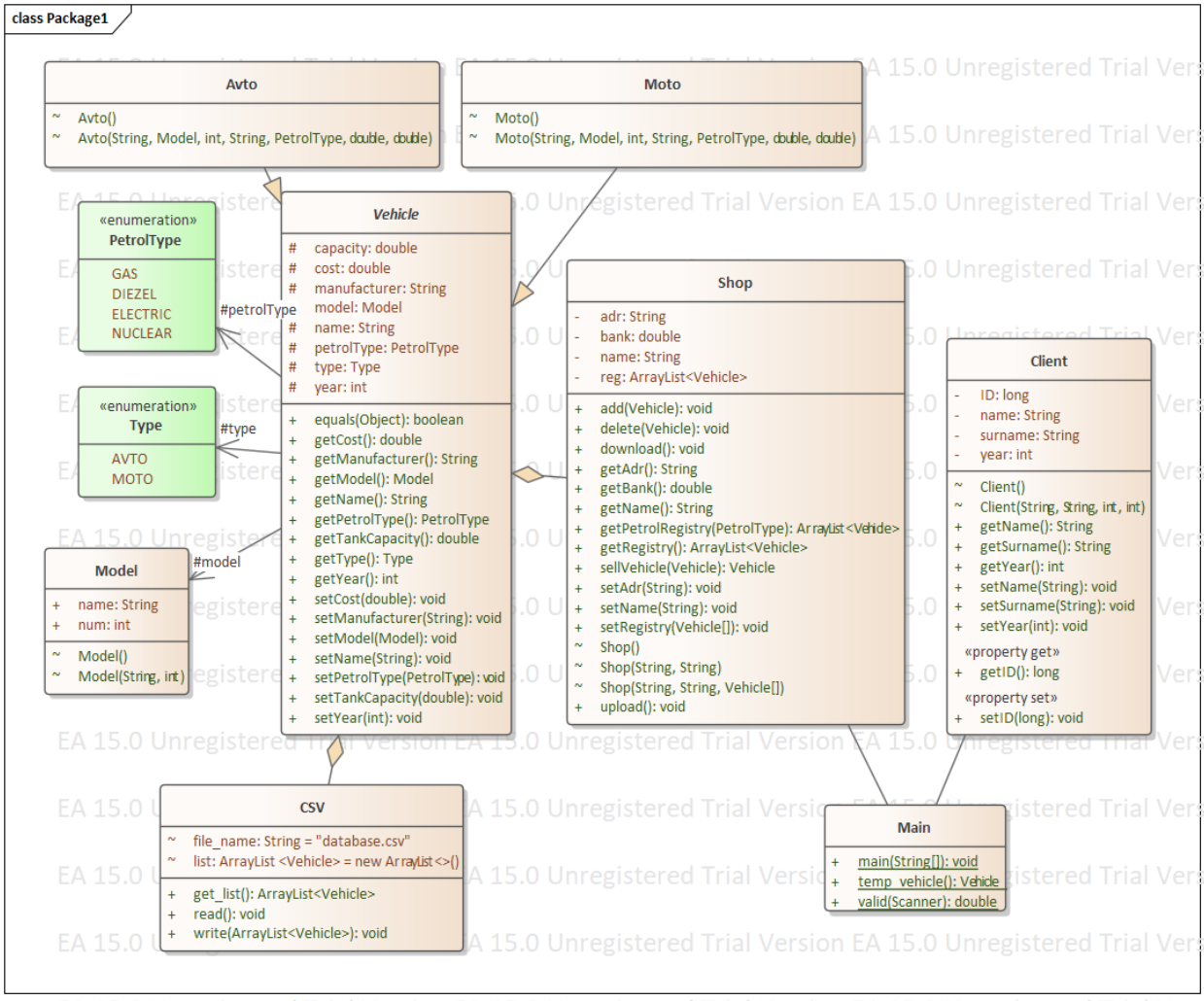

Рисунок 3.1 Діаграма класів проекту

3.4 Опис застосованих класів

Main – клас, де розташоване головне меню. Користувач має змогу додати транспортний засіб до реєстру, видалити транспортний засіб із реєстру, роздрукувати перелік усіх транспортних засобів, роздрукувати перелік усіх авто, роздрукувати перелік усіх мотоциклів, продати обране авто покупцеві магазину, роздрукувати суму продажів за «день».

Shop – клас, інкапсулюючий в собі поведінку самого магазину. Має назву та адресу, та зберігає в оперативній пам`яті відповідно реєстр транспорту і виконує усі відповідні дії над ним, такі як: збереженя/ завантаження з файлу, основні дії з реєстром, може продавати автомобілі.

Client – клас, що відповідає за клієнтів, який здатен пов’язуватися з

транспортним засобом через асоціацію «покупка».

Vehicle – абстрактний клас, що містить основні характеристики та властивості (поля та методи) для транспорту.

Type – перерахування, що допомагає вводити тип транспорту, де авто/мотоцикл є фіксованою множиною.

Model – клас, що містить модель та номер транспорту.

Avto – клас, що є спадкоємцем класу Vehicle, він приймає характеристики, які властиві автомобілю.

Moto – клас, що є спадкоємцем класу Vehicle, він приймає характеристики, які властиві молоциклу.

CSV – клас, який відповідає за процедуру запису/читання з файлу.

РОЗДІЛ 4 ТЕСТУВАННЯ

Класи *Avto* та *Moto* – нащадки клусу *Vehicle* , тож їх тестування проводиться під час тестування суперкласу.

Кінцеве меню програми у консолі виглядає так:

```
1. Додати транспортний засіб до реєстру
2. Видалити транспортний засіб із реєстру
3. Отримати список усіх транспортних засобів, що містяться у реєстрі
4. Отримати список усіх авто із реєстру
5. Отримати список усіх мотоциклів із реєстру
6. Продати обране авто покупцеві магазину, при цьому транспортний
        засіб   видаляється із реєстру та записується ціна продажу
7. Роздрукувати суму продажів за «день»
8. Зберегти реєстр до сховища (текстовий файл)
9. Отримати реєстр із сховища (текстового файлу)
10. Вихід
```

Рисунок 4.1 – Меню програми

Особлива увага під час тестування програми виділялася на перевірку коректності запису та читання з файлу, тобто роботу спеціалізованого класу CSV. Також проходила ретельну перевірку процедура продажу авто, з можливістю зв’язку клієнту та авто за допомогою асоціативного масиву HashMap. У програми доволі велике за обсягом меню взаємодії з користувачем, тож додатково довелося впроваджувати декілька додаткових функцій для спрощення процедур звернення.

РОЗДІЛ 5 ІНСТРУКЦІЯ ДЛЯ КОРИСТУВАЧА

Користувач працює з прототипом програмної системи управління реєстром автосалону. Під час запуску програми дані з реєстру (якщо вони є) завантажуються з текстового файлу. Користувач має можливості:

1. Додати транспортний засіб до реєстру. При цьому викликається підменю заповнення інформації про засіб.

```
Оберіть тип транспортного засобу: 1. Автомобіль 2. Мотоцикл
1
Введіть назву транспортного засобу: Chevrolet
Введіть модель транспортного засобу: Aveo
        номер моделі: 1
Введіть рік виробництва транспортного засобу: 2000
Введіть виробника: Chevrolet
Оберіть тип палива:
1. Бензин
2. Дизель
3. Електрокар
4. Ядерний
2
Введіть ємність паливного сховища: 200.6
---Uncorrect input---
200,9
```

Рисунок 5.1 – Приклад додавання нового транспорту типу Автомобіль

2. Видаляє транспорту із реєстру. Виконується аналогічно, користувач вводить усі необхідні дані, після чого програма, у разі наявності такого автомобілю у реєстрі, видаляє його.

3. Друк переліку усіх транспортних засобів, що містяться у реєстрі.

```
3
MOTO 2 2-2 2 DIEZEL 2.0 2.0
AVTO 1 1-1 1 GAS 1.0 1.0
MOTO 3 3-3 3 ELECTRIC 3.0 3.0
```

Рисунок 5.3 – Приклад переліку усіх транспортних засобів

4. Друк усіх авто з реєстру, які використовують певний тип палива:.

```
4
Оберіть тип палива:
1. Бензин
2. Дизель
3. Електрокар
4. Ядерний
1
MOTO JAWA starushka250 1969 Chech_Republic GAS 15.0 100000.0
```

Рисунок 5.4 - Приклад друку усіх авто з певним типом пального

5. Продаж обраного транспорту з магазину. Відкривається підменю реєстрації даних клієнта, після чого описується транспортний засіб. Виконується видалення з реєстру та збереження ціни продажу.

```
6
Введіть дані покупця:
Ім'я:Авт
Прізвище: ППП
Рік народження:
1999
Ідентифікаційний код:
60523
Опишіть транспортний засіб, який шукає покупець:
Оберіть тип транспортного засобу: 1. Автомобіль 2. Мотоцикл
1
Введіть назву транспортного засобу: Авт
Введіть модель транспортного засобу: ТП
        номер моделі: 25
Введіть рік виробництва транспортного засобу: 20019
Оберіть тип палива:
1. Бензин
2. Дизель
3. Електрокар
4. Ядерний
4
Введіть ємність паливного сховища: 50000
Введіть ціну: 4000000
Продаж відбувся!
```

Рисунок 5.5 – Приклад продажу транспорту з магазину

6. Друк суми продажів за «день»: Виводиться список продажів у форматі «Клієнт : авто ціна», після чого оголошується повна сума прибутку.

```
7

Авт: Авт 4000000.0
Прибуток за день: 4000000.0
```

Рисунок 5.6 – Приклад друку продажів за «день»

7. Зберегти реєстр до сховища. Дані записуються у файл database.csv.

```
8
Дані вигружено до сховища!
```

Рисунок 5.7 – Відклик програми, що вона записала до файлу

MOTO	2	2	2	2	DIEZEL	2.0	2.0
AVTO	1	1	1	1	GAS	1.0	1.0
MOTO	3	3	3	3	ELECTRIC	3.0	3.0
AVTO	Авт	ТП	25	20019	NUCLEAR	50000.0	4000000.0

Рисунок 5.8 – Приклад даних у таблиці

8. Отримати реєстр із сховища

```
3
MOTO 2 2-2 2 DIEZEL 2.0 2.0
AVTO 1 1-1 1 GAS 1.0 1.0
MOTO 3 3-3 3 ELECTRIC 3.0 3.0
AVTO Авт ТП-25 20019 NUCLEAR 50000.0 4000000.0
```

Рисунок 5.9 – Приклад отриманого реєстру з файлу

ВИСНОВКИ

В процесі написання даної курсової роботи була створена програма, яка добре ілюструє гідності даної технології і показує її перспективність. Частина коду програми знаходиться в додатку А. Таким чином, розроблений прототип програмної системи «Автосалон» дає можливість користувачу маніпулювати реєстром видань за допомогою інтерфейсу. Перед завершенням роботи програмної системи реєстр зберігається у постійному сховищі.

Із основних аспектів, які ми використовували у процесі написання були з'ясовані такі властивості мови Java:

- Мова програмування об'єктно-орієнтована, оснащена багатою бібліотекою класів і в той же час досить проста для освоєння
- За рахунок вбудованої системи збірки сміття програміст звільняється від необхідності явного управління пам'яттю
- Додаток легко супроводжується і модифікується, тому що модулі можуть бути завантажені з мережі
- В додатки вбудована система безпеки, що не допускає незаконного доступу і проникнення вірусів.

Отже, можна зробити висновок, що програма вирішує всі поставлені перед нею задачі і задовольняє всім критеріям.

ДОДАТОК А
ФРАГМЕНТИ ВИХІДНОГО КОДУ

Класс Shop:

```
import java.io.IOException;
import java.util.ArrayList;

public class Shop {
    private String name;
    private String adr;
    private ArrayList<Vehicle> reg;
    private double bank;

    Shop(){
      name="some_shop";
      adr="any_adress";
      reg = new ArrayList<>();
      try {
            download();
            } catch (IOException e) {
                  System.out.println("Не вдалося підгрузити реєстр!!!");
            }
    }
    Shop(String name, String adr){
      this.name=name;
      this.adr=adr;
      reg = new ArrayList<>();
      try {
            download();
            } catch (IOException e) {
                  System.out.println("Не вдалося підгрузити реєстр!!!");
            }
    }
    Shop(String name, String adr, Vehicle[] reg_arr){
      this.name=name;
      this.adr=adr;
      reg = new ArrayList<>();
      for (int i = 0; i < reg_arr.length; i++) {
                  reg.add(reg_arr[i]);
            }
    }


    public void download() throws IOException{
      //ф-я подкачки из файла, вызов автоматически при создании объекта
      CSV f = new CSV();
      reg=f.get_list();
    }
    public void upload() throws IOException{
      //ф-я загрузки в файл, перед завершением программы нужно дополнительно
вызвать извне.
      CSV f = new CSV();
      f.write(reg);
    }


    public void add(Vehicle v) {
      reg.add(v);
    }
    public void delete(Vehicle v) {
```

```
      for (int i = 0; i < reg.size(); i++) {
                  if(reg.get(i).equals(v)) {
                        reg.remove(i);
                        break;

                  }
            }
      }
      public Vehicle sellVehicle(Vehicle v) {
        for (int i = 0; i < reg.size(); i++) {
                  if(reg.get(i).equals(v)) {
                        bank+=reg.get(i).getCost();
                        return reg.remove(i);

                  }
            }
        return null;
      }
      public ArrayList<Vehicle> getPetrolRegistry(PetrolType type){
        ArrayList<Vehicle> tmp = new ArrayList<>();
        for (Vehicle vehicle : reg) {
                  if(vehicle.getPetrolType()==type)
                        tmp.add(vehicle);
            }
        return tmp;
      }

      public void setName(String name){
        this.name = name;
      }
      public String getName(){
        return name;
      }
      public void setAdr(String adr){
        this.adr = adr;
      }
      public String getAdr(){
        return adr;
      }
      public void setRegistry(Vehicle[] reg_arr){
        reg = new ArrayList<>();
        for (int i = 0; i < reg_arr.length; i++) {
                  reg.add(reg_arr[i]);
            }
      }
      public ArrayList<Vehicle> getRegistry(){
        ArrayList<Vehicle> tmp = new ArrayList<>();
        for (Vehicle vehicle : reg) {
                  tmp.add(vehicle);
            }
        return tmp;
      }
      public double getBank() {
            return bank;
        }
}
```

Абстрактий клас Vehicle:

```
public abstract class Vehicle {
     protected Type type;
     protected String name;
     protected Model model;
     protected int year;
     protected String manufacturer;
     protected PetrolType petrolType;
     protected double capacity;
     protected double cost;

    public Type getType() {
     return type;
    }
    public void setName(String name) {
     this.name=name;
    }
    public String getName() {
     return name;
    }
    public void setModel(Model model) {
     this.model=model;
    }
    public Model getModel() {
     return model;
    }
    public void setYear(int year) {
     this.year=year;
    }
    public int getYear() {
     return year;
    }
    public void setManufacturer(String manufacturer) {
     this.manufacturer=manufacturer;
    }
    public String getManufacturer() {
     return manufacturer;
    }
    public void setPetrolType(PetrolType petrolType) {
     this.petrolType=petrolType;
    }
    public PetrolType getPetrolType() {
     return petrolType;
    }
    public void setTankCapacity(double capacity) {
     this.capacity=capacity;
    }
    public double getTankCapacity() {
     return capacity;
    }
    public void setCost(double cost) {
     this.cost=cost;
```

```
    }
    public double getCost() {
     return cost;
    }
    @Override
    public boolean equals(Object obj) {
     // TODO Auto-generated method stub
     if(super.equals(obj)==true) return true;
     if(type!=((Vehicle) obj).getType()) return false;
     if(!name.equals(((Vehicle) obj).getName())) return false;
     if(!model.name.equals(((Vehicle) obj).getModel().name))
return false;
     if(model.num!=((Vehicle) obj).getModel().num) return false;
     if(year!=((Vehicle) obj).getYear()) return false;
     if(!manufacturer.equals(((Vehicle) obj).getManufacturer()))
return false;
     if(petrolType!=((Vehicle) obj).getPetrolType()) return
false;
     if(capacity!=((Vehicle) obj).getTankCapacity()) return
false;
     if(cost!=((Vehicle) obj).getCost()) return false;
     else return true;
    }
}
```

Виконуючий клас Main:

```
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class Main{
    public static void main(String []args){
     Shop shop=new Shop("Autoshop", "13 Baker st.");  //магазин
     Scanner s=new Scanner(System.in);
     Map<Client, Vehicle> Sells = new HashMap<>();    //MAP для
хранения продаж
     int ans=0;

     while (true) {
          System.out.println("1. Додати транспортний засіб до
реєстру");
          System.out.println("2. Видалити транспортний засіб із
реєстру");
          System.out.println("3. Отримати список усіх
транспортних засобів, що містяться у реєстрі");
          System.out.println("4. Отримати список усіх
транспортних засобів, тип палива яких відповідає заданому");
          System.out.println("5. Продати обране авто покупцеві
магазину, при цьому транспортний засіб видаляється із реєстру та
записується ціна продажу");
```

```
          System.out.println("6. Роздрукувати суму продажів за
«день»");
          System.out.println("7. Зберегти реєстр до сховища
(текстовий файл)");
          System.out.println("8. Отримати реєстр із сховища
(текстового файлу)");
          System.out.println("9. Вихід");
          ans=(int)valid(s);
          switch (ans) {
               case 1:
                    shop.add(temp_vehicle());
                    break;
               case 2:
                    shop.delete(temp_vehicle());
                    break;
               case 3:
                    for (Vehicle v : shop.getRegistry()) {
                         System.out.println(v.getType().name()+"
"+v.getName()+" "+v.getModel().name+"-"+v.getModel().num+"
"+v.getYear()+" "+v.getManufacturer()+"
"+v.getPetrolType().name()+" "+v.getTankCapacity()+"
"+v.getCost());
                    }
                    break;
               case 4:
                    PetrolType temp;
                    while (true) {
                         System.out.println("Оберіть тип палива:
\n1. Бензин \n2. Дизель \n3. Електрокар \n4. Ядерний");
                    ans=(int)valid(s);
                    switch (ans) {
                         case 1:
                              temp=PetrolType.GAS;
                              break;
                         case 2:
                              temp=PetrolType.DIEZEL;
                              break;
                         case 3:
                              temp=PetrolType.ELECTRIC;
                              break;
                         case 4:
                              temp=PetrolType.NUCLEAR;
                              break;
                         default:
                              System.out.println("---Uncorrect
input---");
                              continue;
                         }
                    break;
                    }
                    for (Vehicle v :
shop.getPetrolRegistry(temp)) {
```

```
                    System.out.println(v.getType().name()+"
"+v.getName()+" "+v.getModel().name+v.getModel().num+"
"+v.getYear()+" "+v.getManufacturer()+"
"+v.getPetrolType().name()+" "+v.getTankCapacity()+"
"+v.getCost());
                }
                break;
            case 5:        //транзакция
                Client client = new Client();
                System.out.print("Введіть дані покупця:
\nІм'я:");
                client.setName(s.next());
                System.out.print("Прізвище: ");
                client.setSurname(s.next());
                System.out.println("Рік народження: ");
                client.setYear((int)valid(s));
                System.out.println("Ідентифікаційний код: ");
                client.setID((long)valid(s));
                System.out.println("Опишіть транспортний
засіб, який шукає покупець: ");
                Vehicle purchase =
shop.sellVehicle(temp_vehicle());

                if(purchase!=null) {
                    Sells.put(client, purchase);
                    System.out.println("Продаж відбувся!");
                }
                else {
                    System.out.println("На жаль,
транспортного засобу, який шукає покупець, не знайдно в
реєстрі");
                }
                break;
            case 6:
                for (Client c : Sells.keySet()) {
                    System.out.println(c.getName()+":
"+Sells.get(c).getName()+" "+Sells.get(c).getCost());
                }
                System.out.println("Прибуток за день:
"+shop.getBank());
                break;
            case 7:
                try {
                    shop.upload();
                    System.out.println("Дані вигружено до
сховища!");
                } catch (IOException e) {
                    System.out.println("Файл не знайдено!");
                }
                break;
            case 8:
                try {
                    shop.download();
```

```
                        System.out.println("Дані загружено зі
сховища!");
                    } catch (IOException e) {
                        System.out.println("Файл не знайдено!");
                    }
                    break;
               case 9:
                    try {
                        shop.upload();
                        System.out.println("Дані вигружено до
сховища!");
                    } catch (IOException e) {
                        System.out.println("Файл не знайдено!");
                    }
                    s.close();
                    System.out.println("---end---");
                    return;
               default:
                    System.out.println("---Uncorrect input---");
                    break;
               }
          }

     }
     public static Vehicle temp_vehicle() {
      int ans;
      Scanner s=new Scanner(System.in);
      Vehicle v;

      while (true) {
                System.out.println("Оберіть тип транспортного
засобу: 1. Автомобіль 2. Мотоцикл");
          ans=(int)valid(s);
          switch (ans) {
               case 1:
                    v=new Avto();
                    break;
               case 2:
                    v=new Moto();
                    break;
               default:
                    System.out.println("---Uncorrect input---");
                    continue;
               }
          break;
          }
          System.out.print("Введіть назву транспортного засобу:
");
          v.setName(s.next());
          System.out.print("Введіть модель транспортного засобу:
");
          String tmp=s.next();
          System.out.print("  номер моделі: ");
```

```
          v.setModel(new Model(tmp, (int)valid(s)));
          System.out.print("Введіть рік виробництва транспортного
засобу: ");
          v.setYear((int)valid(s));
          System.out.print("Введіть виробника: ");
          v.setManufacturer((s.next()));
          while (true) {
               System.out.println("Оберіть тип палива: \n1.
Бензин \n2. Дизель \n3. Електрокар \n4. Ядерний");
          ans=(int)valid(s);
          switch (ans) {
               case 1:
                    v.setPetrolType(PetrolType.GAS);
                    break;
               case 2:
                    v.setPetrolType(PetrolType.DIEZEL);
                    break;
               case 3:
                    v.setPetrolType(PetrolType.ELECTRIC);
                    break;
               case 4:
                    v.setPetrolType(PetrolType.NUCLEAR);
                    break;
               default:
                    System.out.println("---Uncorrect input---");
                    continue;
               }
          break;
          }
          System.out.print("Введіть ємність паливного сховища:
");
          v.setTankCapacity(valid(s));
          System.out.print("Введіть ціну: ");
          v.setCost(valid(s));
          //s.close();
     return v;
     }
    public static double valid(Scanner s) {
     double ans;
     while(true) {
               try {
                    ans=s.nextDouble();
                    break;
               }catch(Exception e) {
                    System.out.println("---Uncorrect input---");
                    s.next();
               }
          }
     return ans;
    }

}
```