

ОБ'ЄКТНО-ОРІЄНТОВАНА ДЕКОМПОЗИЦІЯ

Мета: Використання об'єктно-орієнтованого підходу для розробки об'єкта предметної (прикладної) галузі.

ВИМОГИ

Розробник:

- Веремчук Дарина Анатоліївна;
- КІТ-119д;
- Варіант №5.

Загальне завдання:

- 1) Використовуючи об'єктно-орієнтований аналіз, реалізувати класи для представлення сутностей відповідно прикладної задачі - domain-об'єктів.
- 2) Забезпечити та продемонструвати коректне введення та відображення кирилиці.
- 3) Продемонструвати можливість управління масивом domain-об'єктів.

Індивідуальне завдання:

Довідник покупця

Торгівельна точка: назва; адреса; телефони (кількість не обмежена); спеціалізація; час роботи (з зазначенням днів тижня).

ОПИС ПРОГРАМИ

Опис змінних:

Scanner in; // зчитування даних з клавіатури

```
int amount;                // кількість торгівельних точок
BuyersGuide[] TestStore;    // масив торгівельних точок
```

Ієрархія та структура класів:

class Veremchuk07 – точка входу в програму;

class BuyersGuide – клас-планувальник

ТЕКСТ ПРОГРАМИ

Текст класу Veremchuk07:

```
package ua.oop.khpi.veremchuk07;

import java.io.IOException;
import java.util.Scanner;

public class Veremchuk07 {
    /**
     * An entry point, the main method
     *
     * @param args - arguments of function
     */
    public static void main(String[] args) throws IOException {
        Scanner scan = new Scanner(System.in);
        System.out.print("Сколько торговых точек"
            + " добавить? ");
        int size = scan.nextInt();
        scan.nextLine();
        BuyersGuide[] stores = new BuyersGuide[size];
        for (int i = 0; i < stores.length; i++) {
            System.out.format("Торговая точка %d:%n", i + 1);
            stores[i] = BuyersGuide.generate();
        }
    }
}
```

```

        System.out.println();
        for (int i = 0; i < stores.length; i++) {
            System.out.format("Торговая точка %d:%n", i + 1);
            System.out.println(stores[i].toString());
        }
    }
}

```

Текст класу **BuyersGuide**:

```

package ua.oop.khpi.veremchuk07;

import java.io.*;
import java.util.*;

import java.util.List;

public class BuyersGuide implements Serializable {
    /** Identify key of serialization */
    private static final long serialVersionUID = 2845790659809642584L;
    /** Keep the name of store */
    private String name;
    /** Keep the address of store */
    private String address;
    /** Keep the numbers of store */
    private List<String> numbers;
    /** Keep the specialization of store */
    private String specialization;
    /** Keep a work time of store */
    private String workTime;

    public BuyersGuide(String name, String address, String spec, String WT, String... nums) {
        this.name = name;
        this.address = address;

```

```
        this.numbers = new ArrayList<>(nums.length);
        this.numbers.addAll(Arrays.asList(nums));
        this.specialization = spec;
        this.workTime = WT;
    }
```

```
public BuyersGuide() {
    this.setName(null);
    this.setAddress(null);
    this.numbers = new ArrayList<>();
    this.setWorkTime(null);
    this.setSpecialization(null);
}
```

```
public String getName() {
    return this.name;
}
```

```
public void setName(String name) {
    this.name = name;
}
```

```
public String getAddress() {
    return this.address;
}
```

```
public void setAddress(String address) {
    this.address = address;
}
```

```
public List<String> getNumbers() {
    return this.numbers;
}
```

```

public void setNumbers(final List<String> part) {
    if (this.numbers.size() == 0)
        this.numbers.addAll(part);
}

public String getSpecialization() {
    return this.specialization;
}

public void setSpecialization(String specialization) {
    this.specialization = specialization;
}

public String getWorkTime() {
    return this.workTime;
}

public void setWorkTime(String workTime) {
    this.workTime = workTime;
}

/**
 * Load a numbers of stores.
 * @param CountNumbers - count of numbers
 * @return список участников
 * @throws IOException - при
 * некорректном считывании
 */
public void fillNumbers(final int CountNumbers) throws IOException{
    BufferedReader reader = new BufferedReader(
        new InputStreamReader(System.in));

    System.out.format("Введите номера"
        + " %s торговых точек.%n", CountNumbers);

    String number;

```

```

        this.numbers = new ArrayList<>();
        for (int i = 0; i < CountNumbers; i++) {
            System.out.format("Номер №%d: ", i + 1);
            number = reader.readLine();
            this.numbers.add(number);
        }
    }

    public static BuyersGuide generate () throws IOException {
        Scanner in = new Scanner(System.in);
        BuyersGuide TestStore = new BuyersGuide();
        System.out.print("Введите название торговой точки: ");
        TestStore.setName(in.nextLine());
        System.out.print("Введите адрес торговой точки (Город, улица, номер дома): ");
        TestStore.setAddress(in.nextLine());
        System.out.print("Введите кол-во номеров у торговой точки: ");
        int amount = in.nextInt();
        in.nextLine();
        TestStore.fillNumbers(amount);
        System.out.print("Введите специализацию торговой точки: ");
        TestStore.setSpecialization(in.nextLine());
        System.out.print("Введите время работы торговой точки (День недели - время): ");
        TestStore.setWorkTime(in.nextLine());

        return TestStore;
    }

    /**
     * Overriding of method toString().
     */
    @Override
    public String toString() {
        StringBuilder builder = new StringBuilder();
        builder.append("Название торговой точки: ").append(this.getName()).append("\n");
    }

```

```

        builder.append("Адрес: ").append(
            this.getAddress()).append("\n");
        builder.append("Номер(a): ");
        if (this.getNumbers() != null) {
            for (String number : this.getNumbers()) {
                builder.append(number).append(" ");
            }
        } else {
            builder.append("null");
        }
        builder.append("\nСпециализация: ").append(
            this.getSpecialization()).append("\n");
        builder.append("Время работы (график): ").append(
            this.getWorkTime()).append("\n");
        return builder.append("\n").toString();
    }
}

```

РЕЗУЛЬТАТ РОБОТЫ ПРОГРАММЫ

```

Сколько торговых точек добавить? 2
Торговая точка 1:
Введите название торговой точки: Lba
Введите адрес торговой точки (Город, улица, номер дома): г. Харьков, ул. Пушкинская 111
Введите кол-во номеров у торговой точки: 2
Введите номера 2 торговых точек.
Номер №1: +380936476525
Номер №2: +380264789301
Введите специализацию торговой точки: Магазин товаров для красоты и ухода
Введите время работы торговой точки (День недели - время): ПН - ПТ: 08:00 - 20:00
Торговая точка 2:
Введите название торговой точки: ATB
Введите адрес торговой точки (Город, улица, номер дома): г. Харьков, ул. Пушкинская 222
Введите кол-во номеров у торговой точки: 1
Введите номера 1 торговых точек.
Номер №1: +380237489031
Введите специализацию торговой точки: Продуктовый магазин
Введите время работы торговой точки (День недели - время): ПН - ВС: 00:00 - 24:00

```

a)

```
Торговая точка 1:  
Название торговой точки: Ева  
Адрес: г. Харьков, ул. Пушкинская 111  
Номер(а): +380936476525 +380264789301  
Специализация: Магазин товаров для красоты и ухода  
Время работы (график): ПН - ПТ: 08:00 - 20:00  
  
Торговая точка 2:  
Название торговой точки: АТБ  
Адрес: г. Харьков, ул. Пушкинская 222  
Номер(а): +380237489031  
Специализация: Продуктовый магазин  
Время работы (график): ПН - ВС: 00:00 - 24:00  
  
Process finished with exit code 0
```

б)

Рисунок 7.1 – Результат роботи програми

ВАРІАНТИ ВИКОРИСТАННЯ

Програма може бути використана для створення та збереження інформації про торгівельні точки.

ВИСНОВКИ

Під час лабораторної роботи, набула практичних навичок щодо розробки класів для заданої предметної області.