

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №4

Специальность ПО

Выполнил

Войтюк Е.О.

студент группы ПО-8

Проверил

А. А. Крощенко,

ст. преп. кафедры ИИТ,

«__k_____2024 г.

Брест 2024

Цель работы: приобрести практические навыки в области объектно-ориентированного проектирования

Задание 1. 6) Создать класс Catalog (каталог) с внутренним классом, с помощью объектов которого можно хранить информацию об истории выдач книги читателям.

Выполнение:

Код программы

Catalog:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class Catalog {
    private List<BookHistory> bookHistories;

    public Catalog() {
        this.bookHistories = new ArrayList<>();
    }

    public void addBookHistory(String bookName, String readerName) {
        this.bookHistories.add(new BookHistory(bookName, readerName));
    }

    public void printBookHistories() {
        for (BookHistory bookHistory : bookHistories) {
            System.out.println(bookHistory);
        }
    }

    public void addBookHistoryFromFile(String filename) {
        try (BufferedReader reader = new BufferedReader(new FileReader(filename))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] parts = line.split(";", 2);
                if (parts.length >= 2) {
                    String bookName = parts[0];
                    String readerName = parts[1];
                    addBookHistory(bookName, readerName);
                } else {
                    System.out.println("Ignoring invalid line: " + line);
                }
            }
        } catch (IOException e) {
```

```

        System.out.println("Error reading file: " + e.getMessage());
    }
}

private class BookHistory {
    private String bookName;
    private String readerName;

    public BookHistory(String bookName, String readerName) {
        this.bookName = bookName;
        this.readerName = readerName;
    }

    @Override
    public String toString() {
        return "Book " + bookName + " was issued to " + readerName;
    }
}
}

```

Main:

```

import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        Catalog catalog = new Catalog();
        catalog.addBookHistoryFromFile("C:/Users/egor-/IdeaProjects/lab4task1/src/books.txt");
        catalog.printBookHistories();
    }
}

```

Рисунки с результатами работы программы

```

C:\Users\egor-\.jdk\openjdk-21.0.2\bin\java.exe "-javaagent:
Book 'War and Peace' was issued to Ivan Ivanov
Book 'Pride and Prejudice' was issued to Maria Petrova

Process finished with exit code 0

```

Задание 2.

6) Создать класс Страница, используя класс Слово.

Выполнение:

Код программы

Main:

```
import java.util.ArrayList;

class Word {
    private String word;

    public Word(String word) {
        this.word = word;
    }

    public String getWord() {
        return word;
    }

    public int length() {
        return word.length();
    }
}

class Page {
    private ArrayList<Word> words;

    public Page(ArrayList<Word> words) {
        this.words = words;
    }

    public void addWord(Word word) {
        words.add(word);
    }

    public int wordCount() {
        return words.size();
    }

    public void displayWords() {
        System.out.println("Слова на странице:");
        for (Word word : words) {
            System.out.print(word.getWord() + " ");
        }
        System.out.println();
    }
}
```

```

    public int totalLength() {
        int totalLengthOfWords = 0;
        for (Word word : words) {
            totalLengthOfWords += word.length();
        }
        return totalLengthOfWords;
    }
}

public class Main2 {
    public static void main(String[] args) {
        Word word1 = new Word("Егор");
        Word word2 = new Word("Войтюк");

        ArrayList<Word> pageWords = new ArrayList<>();
        pageWords.add(word1);
        pageWords.add(word2);
        Page page = new Page(pageWords);

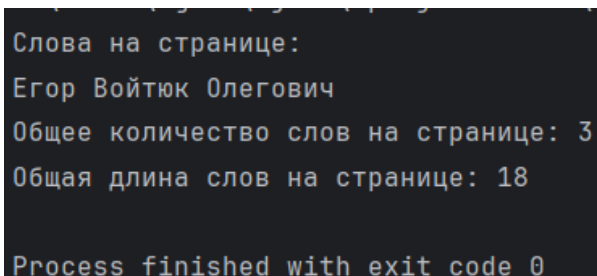
        Word word3 = new Word("Олегович");
        page.addWord(word3);

        page.displayWords();

        System.out.println("Общее количество слов на странице: " + page.wordCount());
        System.out.println("Общая длина слов на странице: " + page.totalLength());
    }
}

```

Рисунки с результатами работы программы



```

Слова на странице:
Егор Войтюк Олегович
Общее количество слов на странице: 3
Общая длина слов на странице: 18

Process finished with exit code 0

```

Задание 3.

6) Система Телефонная станция. Абонент оплачивает Счет за разговоры и Услуги, может попросить Администратора сменить номер и отказаться от услуг. Администратор изменяет номер, Услуги и временно отключает Абонента за неуплату.Выполнение:

Код программы

```
import java.util.ArrayList;
import java.util.List;

class Arrears {
    private double amount;

    public Arrears(double amount) {
        this.amount = amount;
    }

    public double getAmount() {
        return amount;
    }

    public void setAmount(double amount) {
        this.amount = amount;
    }
}

class Service {
    private String name;
    private double price;

    public Service(String name, double price) {
        this.name = name;
        this.price = price;
    }

    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }
}

class Subscriber {
    public String phoneNumber;
    public List<Service> services;
    public Arrears arrears;
    public boolean isActive;

    public Subscriber(String phoneNumber) {
```

```

        this.phoneNumber = phoneNumber;
        this.services = new ArrayList<>();
        this.arrears = new Arrears(0);
        this.isActive = true;
    }

    public void requestPhoneNumberChange(Administrator administrator, String newNumber) {
        administrator.changePhoneNumber(this, newNumber);
    }

    public void requestService(Administrator administrator, Service service) {
        administrator.requestService(this, service);
    }

    public void cancelService(Administrator administrator, Service service) {
        administrator.cancelService(this, service);
    }

    public void payArrears(double amount) {
        this.arrears.setAmount(this.arrears.getAmount() - amount);
        System.out.println("Абонент " + this.phoneNumber + " положил сумму" + amount + " на счет.");
        if(!this.isActive && !checkUnpaidArrears())
        {
            this.isActive = true;
            System.out.println
                ("Абонент " + this.phoneNumber + " вновь подключен.");
        }
    }

    public void accountAmount()
    {
        if(checkUnpaidArrears())
        {
            System.out.println
                ("У абонента " + this.phoneNumber + " имеется задолженность суммой " +
this.arrears.getAmount());
        }
        else
        {
            System.out.println
                ("У абонента " + this.phoneNumber + " имеется остаток на счете суммой " + (-
this.arrears.getAmount()));
        }
    }

    public boolean checkUnpaidArrears() {
        return this.arrears.getAmount() > 0;
    }
}

class Administrator {
    public void changePhoneNumber(Subscriber subscriber, String newNumber) {

```

```
        System.out.println("Абонент с номером " + subscriber.phoneNumber + " сменил номер телефона  
на " + newNumber);  
        subscriber.phoneNumber = newNumber;  
    }
```

```
    public void requestService(Subscriber subscriber, Service service) {  
        subscriber.services.add(service);  
        subscriber.arrears.setAmount(subscriber.arrears.getAmount() + service.getPrice());  
        System.out.println("Абонент " + subscriber.phoneNumber + " подписался на услугу: " +  
service.getName());  
    }
```

```
    public void cancelService(Subscriber subscriber, Service service) {  
        if (subscriber.services.contains(service)) {  
            subscriber.services.remove(service);  
            subscriber.arrears.setAmount(subscriber.arrears.getAmount() - service.getPrice());  
            System.out.println("Абонент " + subscriber.phoneNumber + " отписался от услуги: " +  
service.getName());  
        }  
    }
```

```
    public void temporarilyDisableSubscriber(Subscriber subscriber) {  
        if (subscriber.checkUnpaidArrears()) {  
            subscriber.isActive = false;  
            System.out.println("Абонент " + subscriber.phoneNumber + " отключен за неуплату.");  
        }  
    }  
}
```

```
class TelephoneStation {  
    private List<Subscriber> subscribers;  
  
    public TelephoneStation() {  
        this.subscribers = new ArrayList<>();  
    }  
  
    public void addSubscriber(Subscriber subscriber) {  
        this.subscribers.add(subscriber);  
    }  
  
    public List<Subscriber> getSubscribers() {  
        return subscribers;  
    }  
}
```

```
public class Main3 {  
    public static void main(String[] args) {  
        TelephoneStation telephoneStation = new TelephoneStation();  
  
        Service service1 = new Service("Интернет обслуживание", 34.0);  
        Service service2 = new Service("Подписка на музыку", 89.0);  
        Service service3 = new Service("Подписка на анекдоты", 15.0);
```



```

Subscriber subscriber1 = new Subscriber("297228696");

telephoneStation.addSubscriber(subscriber1);

Administrator administrator = new Administrator();

subscriber1.requestPhoneNumberChange(administrator, "336663432");

subscriber1.payArrears(15);

subscriber1.requestService(administrator, service1);
subscriber1.requestService(administrator, service2);
subscriber1.requestService(administrator, service3);

subscriber1.accountAmount();

subscriber1.cancelService(administrator, service2);

administrator.temporarilyDisableSubscriber(subscriber1);

subscriber1.payArrears(1000);

subscriber1.accountAmount();

}
}

```

Рисунки с результатами работы программы

```

Абонент с номером 297228696 сменил номер телефона на 336663432
Абонент 336663432 положил сумму15.0 на счет.
Абонент 336663432 подписался на услугу: Интернет обслуживание
Абонент 336663432 подписался на услугу: Подписка на музыку
Абонент 336663432 подписался на услугу: Подписка на анекдоты
У абонента 336663432 имеется задолженность суммой 123.0
Абонент 336663432 отписался от услуги: Подписка на музыку
Абонент 336663432 отключен за неуплату.
Абонент 336663432 положил сумму1000.0 на счет.
Абонент 336663432 вновь подключен.
У абонента 336663432 имеется остаток на счете суммой 966.0

```

Вывод: приобрел практические навыки в области объектно-ориентированного проектирования