Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Ярославский государственный технический университет»
Кафедра «Информационные системы и технологии»

Отчет защищен
с оценкой _____
Преподаватель
_____Д.В.Дидковская
«___»_____ 2022

КЛАССЫ, ИНТЕРФЕЙСЫ, НАСЛЕДОВАНИЕ, ПОЛИМОРФИЗМ

Отчёт о лабораторной работе №7 по курсу “Информационные технологии”
ЯГТУ 09.03.02-024 ЛР

Отчет выполнил
студент группы ЭИС-26
_____ А.А. Хрящев
«___»_____ 2022

2022

Цель работы:

Ознакомился с главными парадигмами языка Java, а также создания приложение с их использованием наследования или интерфейса.

Задание:

Создать суперкласс Учащийся и подклассы Школьник и Студент. Создать массив объектов суперкласса и заполнить этот массив объектами. Показать отдельно студентов и школьников.

Требования и рекомендации к выполнению задания:

1. проанализировать полученное задание, выделить информационные объекты и действия;
2. разработать программу с использованием абстрактных классов или интерфейсов;
3. использовать при разработке наследование и полиморфизм.
4. Работу выполнить с использованием графического интерфейса по аналогии с примером (см. папку «Приложение Классы»).

Код программы:

```
package View;


import io.reactivex.rxjava3.core.Observable;

import io.reactivex.rxjava3.disposables.Disposable;

import io.reactivex.rxjava3.subjects.BehaviorSubject;

import models.Learner;

import models.SchoolChild;

import models.Student;

import javax.swing.*;
```

```
public class Work7 {
    private JPanel jPanel;
    private JTextArea textArea1;
    private JButton createLearnersButton;
    private JButton unsubscribeLearnersButton;
    private JButton showMeStudentsButton;
    private JButton showMeSchoolChildButton;
    private Observable<Learner> learnerObservable = null;
    private BehaviorSubject<String> behaviorSubject = BehaviorSubject.create();
    private Disposable disposableStudent = null, disposableSchoolChild = null;

    public Work7() {
        behaviorSubject.subscribe(text -> textArea1.setText(textArea1.getText() +
text + "\n\n"));

        createLearnersButton.addActionListener(e -> {
            createLearner();
        });

        unsubscribeLearnersButton.addActionListener(e -> {
            if(disposableStudent != null) {
                behaviorSubject.onNext("unsubscribe Student Data");
                if(!disposableStudent.isDisposed()) {
                    disposableStudent.dispose();
```

```
            }
        }

        if(disposableSchoolChild != null) {
            behaviorSubject.onNext("unsubscribe SchoolChild Data");
            if(!disposableSchoolChild.isDisposed()) {
                disposableSchoolChild.dispose();
            }
        }
    });

    showMeStudentsButton.addActionListener(e -> {
        if(learnerObservable != null) {
            disposableStudent = learnerObservable
                    .filter(learner -> learner instanceof Student)
                    .subscribe(student -> behaviorSubject.onNext(student.toString()));
        }
    });

    showMeSchoolChildButton.addActionListener(e -> {
        if(learnerObservable != null) {
            disposableSchoolChild = learnerObservable
                    .filter(learner -> learner instanceof SchoolChild)
                    .subscribe(schoolChild ->
behaviorSubject.onNext(schoolChild.toString()));
```

```
            }
        });


    }


    private void createLearner() {
        if(learnerObservable == null) {
            learnerObservable = Observable.fromArray(new SchoolChild(0,
"SchoolChild1", "4A"),
                    new SchoolChild(1, "SchoolChild2", "6D"),
                    new SchoolChild(2, "SchoolChild3", "5F"),
                    new SchoolChild(3, "SchoolChild4", "1S"),
                    new Student(0, "Student1", "Faculty1", 2, "PI"),
                    new Student(1, "Student2", "Faculty2", 3, "TA"),
                    new Student(2, "Student3", "Faculty3", 1, "PI"),
                    new Student(3, "Student4", "Faculty4", 4, "AP"));
            behaviorSubject.onNext("Create data");
        }
    }


    public JPanel getJPanel() {
        return jPanel;
    }
}
```

Суперкласс:

```
package models;

import com.google.gson.Gson;

public class Learner {
    private int id;
    private String name;

    public Learner(int id, String name) {
        this.id = id;
        this.name = name;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

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

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

    @Override
    public String toString() {
        return new Gson().toJson(this);
    }
}
```

Классы для учащихся:

```
package models;

public class Student extends Learner {
    private final String faculty;
    private final int course;
    private final String group;

    public Student(int id, String name, String faculty, int course, String group) {
        super(id, name);
        this.faculty = faculty;
        this.course = course;
        this.group = group;
```

```
    }

    public String getFaculty() {
        return faculty;
    }

    public int getCourse() {
        return course;
    }

    public String getGroup() {
        return group;
    }
}
package models;

public class SchoolChild extends Learner {
    private String educationalClass;

    public SchoolChild(int id, String name, String educationalClass) {
        super(id, name);
        this.educationalClass = educationalClass;
    }
```

```
    public String getEducationalClass() {
        return educationalClass;
    }

    public void setEducationalClass(String educationalClass) {
        this.educationalClass = educationalClass;
    }
}
```

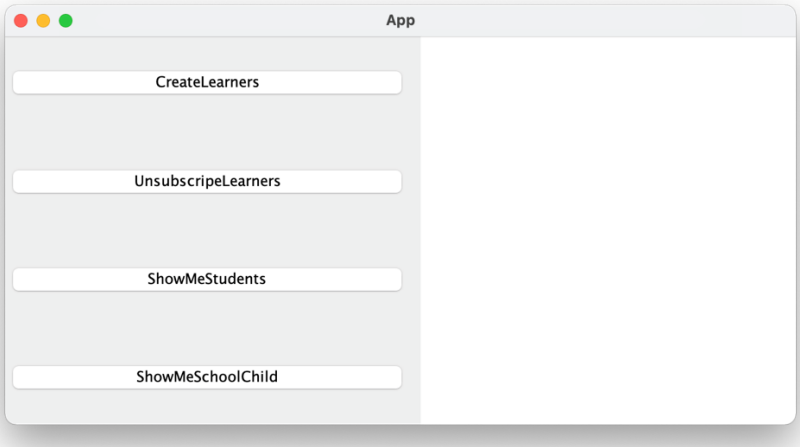

Рисунок 1 – Скриншот интерфейса

Скриншоты выполнения:

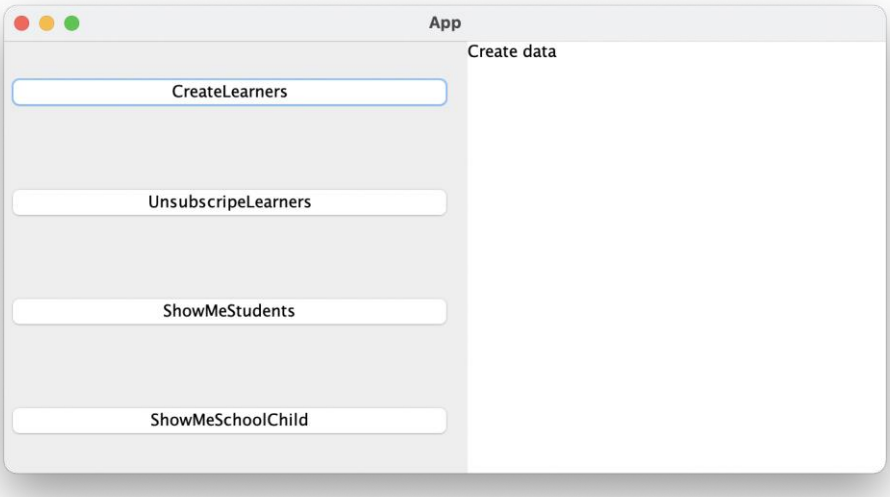

Рисунок 2 – Создание учащихся

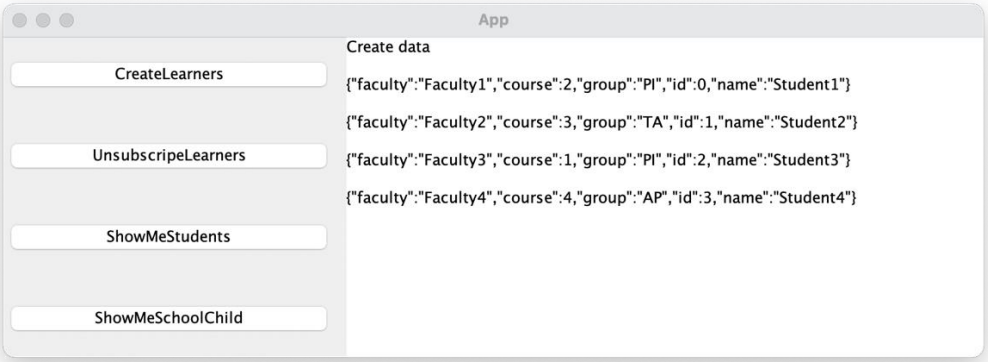

Рисунок 3 – Фильтрация по студентам

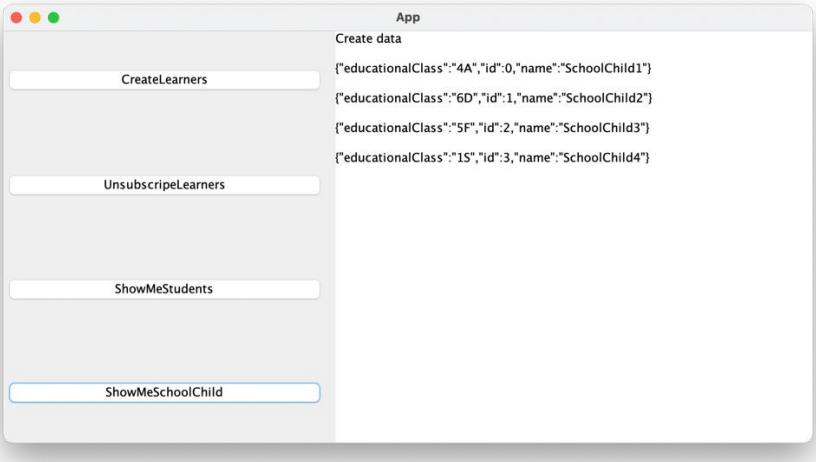

Рисунок 4 – Фильтрация по школьникам

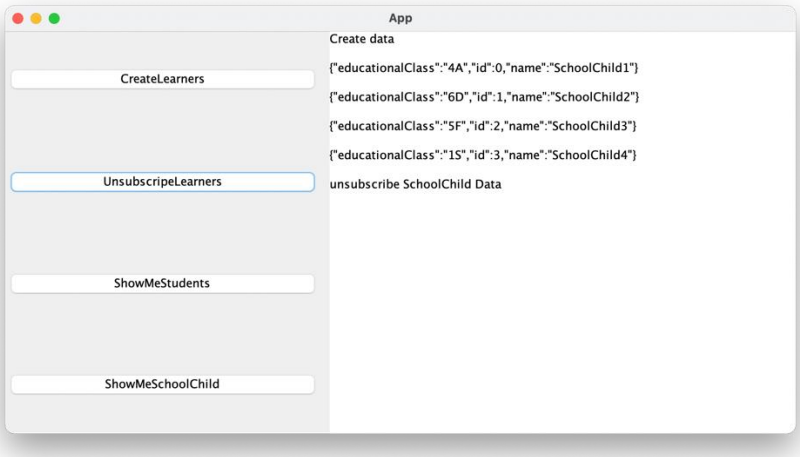

Рисунок 5 – Роспуск учащихся

Вывод:

Я ознакомился с главными парадигмами языка Java, а также создал приложение с их использованием наследования.