МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Отчет

по лабораторной работе №1

по дисциплине

Шаблоны проектирования программного обеспечения

РУКОВОДИТЕЛЬ:

_____ Жевнерчук Д.В.
(подпись) (фамилия, и.,о.)

СТУДЕНТ:

_____ Соляник Д

_____ Куприхин Д
(подпись) (фамилия, и.,о.)

19-ИВТ-2
(шифр группы)

Работа защищена «___»_____

С оценкой _____

Нижний Новгород 2021

Вариант 17

Разработайте и реализуйте объектно-ориентированную модель, на основе которой можно будет создать автоматизированную систему регистрации и сопровождения индивидуальных траекторий обучения. Индивидуальная траектория обучения складывается из:

− обязательных для всех дисциплин;

− рекомендуемых факультативных дисциплин, из которых учащимся может быть выбрано не менее n дисциплин.

В рамках обязательных и факультативных дисциплин могут выполняться учебные проекты, которые представляют собой множество задач, причем каждая задача может быть зарегистрирована в качестве альтернативного варианта лабораторной или практической работы.

Обоснование выбора паттернов

Модель системы регистрации студента предполагает работу с различными видами дисциплин, которые могут иметь подзадачи. Это позволяет нам представить модель в виде дерева.

В рамках задания мы решили использовать паттерн Компоновщик. Компоновщик — это структурный паттерн, который позволяет сгруппировать множество объектов в древовидную структуру, а затем работать с ней так, как будто это единичный объект. Он позволяет рассматривать все подклассы через единый класс с общими методами.

Для решения задачи было принято решение использовать Посредник. Посредник – это поведенческий паттерн, который позволяет уменьшить связанность множества классов между собой, благодаря перемещению этих связей в один класс-посредник. Данный паттерн реализует класс AdjustmentRaspisanie. Он связывает подклассы Discipline и класс Raspisanie.

Программный код

Application.java

```
package com.solkup.lab1;


import java.util.ArrayList;
import Discipline.Facultative;
import Discipline.Obligatory;
import Discipline.FacultativeProject;
import Student.Student;
```

```
import WorkWithRaspisaniem.AdjustmentRaspisanie;
import WorkWithRaspisaniem.Raspisanie;

public class Application {

	public static void main(String[] args) {
//------------Создаем объект студент -----------------------------------
		Student accountStudent = new Student();


//------------Создаем список факультативных предметов---------------------

		ArrayList<Facultative> list_facultatives = new
ArrayList<Facultative>();
		Facultative robotics = new Facultative("Робототехника -
прикладная наука, занимающаяся разработкой автоматизированных технических
систем.",2);
		Facultative ads = new Facultative("Алгоритмы и структуры
данных.",2);
		list_facultatives.add(robotics);
		list_facultatives.add(ads);

//------------Создаем список факультативных задач-------------------------
		ArrayList<Facultative> tasks_facultatives = new
ArrayList<Facultative>();
		FacultativeProject roboticsProject1 = new FacultativeProject("
При успешном решении данной задачи Вам зачтут одну лабораторную по предмету
Машинное обучение.", "Машинное зрение и распознавание лиц в реальном
времени.","r486vy","Машинное обучение",1,0);
		FacultativeProject roboticsProject2 = new FacultativeProject("
При успешном решении данной задачи Вам зачтут одну практическую работу по
предмету ШППО.","Оптимизация и переобучение на примере мктода К-
соседей.","r846cu","ШППО",0,1);
		FacultativeProject adsProject1= new FacultativeProject(" При
успешном решении данной задачи Вам зачтут одну лабораторную по предмету
ШППО.","Анализ данных, визуализация классификация","a312bg","ШППО",1,0);
		FacultativeProject adsProject2= new FacultativeProject(" При
успешном решении данной задачи Вам зачтут одну практическую работу по
предмету Машинномое обучение.","Работа с OpenCV.","a764ih","Машинное
обучение",0,1);
		tasks_facultatives.add(roboticsProject1);
		tasks_facultatives.add(roboticsProject2);
		tasks_facultatives.add(adsProject1);
		tasks_facultatives.add(adsProject2);

//------------Создаем список обязательных предметов----------------------
		ArrayList<Obligatory> list_obligatory= new
ArrayList<Obligatory>();
		String[] lab_topic = new String[] {"1)Разработка слабо связанного
программного кода на основе шаблонов проектирования GoF","2)Spring
Framework","3)разработка многопоточных приложений с помощью
Concurrency","4)Разработка актор-ориентированных приложений в java akka"};
		String[] pract_topic = new String[] {"1) Основные шаблоны
проектирования","2) Порождающие шаблоны проектирования","3) Структурные
шаблоны проектирования","4) Поведенческие шаблоны проектирования","5)
Архитектурные шаблоны проектирования"};
```

```
            Obligatory hppo = new Obligatory("ШППО",lab_topic,pract_topic);
            lab_topic = new String[] {"1) Анализ данных с помошью Pandas","2)
Линейная регрессия","3) Деревья решений","4) Метод случайного леса"};
            pract_topic = new String[] {"1) Метрики качества задач
классификации","2) Предобработка данных. Отбор признаков","3) Функции ошибок
в машинном обучении","4) Алгоритмы кластеризации"};
            Obligatory machineLearning = new Obligatory("Машинное
обучение",lab_topic,pract_topic);
            list_obligatory.add(hppo);
            list_obligatory.add(machineLearning);

//------------Создаем объект обязательного расписания--------------------
            Raspisanie raspisanie= new Raspisanie(list_obligatory);

//------------Создаем объект факультативного расписания------------------
            AdjustmentRaspisanie facultative = new
AdjustmentRaspisanie(list_facultatives, tasks_facultatives,raspisanie);


////////////////////////////////////////////////////////////////////////////
////////////////////////
            accountStudent.initialization();
// Инициализируем студента
            System.out.println("Обязательное расписание для Вас:");
            raspisanie.printObligatoryRaspisanie();
// Выводим обязательное расписание
            facultative.facultativeRaspisanie();
// Выводим список факультативов, предлагаем выбрать задачи и корректируем
расписание
            System.out.println();
            System.out.println("Окончательное расписание для Вас:");
            raspisanie.printObligatoryRaspisanie();
// Выводи окончательное расписание
            facultative.printFacultative();
      }

}
```

Student.java

```
package Student;

import java.util.Scanner;


public class Student {
      private String name = "Student";
      private String group;
      private String id;


      public Student(String name, String group, String id) {
            this.setName(name);
            this.group = group;
            this.id = id;
      }
```

```
	public Student() {}

	public void initialization() {
		Scanner sc = new Scanner(System.in);
		System.out.print("Введите имя:");
		name = sc.nextLine();
		System.out.print("Введите название группы:");
		group = sc.nextLine();
		System.out.print("Введите номер студенческого билета :");
		id = sc.nextLine();
	}

	public String getName() {return name;}

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

	public String getGroup() {return group;}

}
```

Discipline.java

```
package Discipline;

public class Facultative extends Discipline {
	private String name;
	private String info;
	private String id;
	private String object;
	private int count_tasks = 0;

	public Facultative() {}

	public Facultative(String name, int count_tasks) {
		this.name = name;
		this.count_tasks = count_tasks;
	}

	public String getName() {return name;}

	public String getInfo() {return info;}

	public String getId() {return id;}

	public String getObject() {return object;}

	public int getCountTasks() {return count_tasks;}

	public int getLabWorks() {return 0;}

	public int getPractWorks() {return 0;}
}
```

Raspisanie.java

```
package WorkWithRaspisaniem;

import java.util.ArrayList;

import Discipline.Obligatory;


public class Raspisanie {
	private ArrayList<Tasks> list_lab_works= new ArrayList<Tasks>();
	private ArrayList<Tasks> list_pract_works= new ArrayList<Tasks>();
	private ArrayList<Obligatory> list_discipline= new
ArrayList<Obligatory>();

	public Raspisanie() {createTasks();}

	public Raspisanie(ArrayList<Obligatory> arrayList) {
		this.list_discipline = arrayList;
		createTasks();
	}

	public void printObligatoryRaspisanie() {
		for (int i = 0; i < list_lab_works.size(); i++) {

	list_discipline.get(i).printPredmet(list_lab_works.get(i).getColichestv
o(), list_pract_works.get(i).getColichestvo());
			System.out.println();
		}
	}

	private void createTasks() {
		for (int i = 0; i < list_discipline.size(); i++) {
			list_lab_works.add(new Tasks());
			list_pract_works.add(new Tasks());
		}
	}

	public void recalculation(int colvo_lab,int colvo_pract,String name) {
		for (int i = 0; i < list_discipline.size(); i++) {
			if (list_discipline.get(i).getName().equals(name)) {
				list_lab_works.get(i).setColichestvo(colvo_lab);
				list_pract_works.get(i).setColichestvo(colvo_pract);

			}
		}
	}

}
```

AdjustmentRaspisanie.java

```
package WorkWithRaspisaniem;

import java.util.ArrayList;
import java.util.Scanner;
```

```
import Discipline.Facultative;

public class AdjustmentRaspisanie{
	private ArrayList<Facultative> facultatives = new
ArrayList<Facultative>();		// Список факультативных дисциплин
	private ArrayList<Facultative> tasts_facultatives = new
ArrayList<Facultative>();	// Список задач по факультативным дисциплинам
	private Raspisanie raspisanie;
									// Обязательное расписание
	private int n = 2;
										// Минимальное число
факультативных задач для студента
	private int size = 0;
									// Вспомогательная переменная
	private String id = "";
									// Строка куда будут записываться id
задачи, которые выбрал студент
	private String[] real_id;
									// Вспомогательная переменная
	private int j = 0;
										// Вспомогательная переменная

	public AdjustmentRaspisanie() {}

	public AdjustmentRaspisanie(ArrayList<Facultative>
facultatives,ArrayList<Facultative> tasts_facultatives,Raspisanie
raspisanie1) {
		this.facultatives = facultatives;
		this.tasts_facultatives = tasts_facultatives;
		this.raspisanie = raspisanie1;
	}

	public void facultativeRaspisanie() {
//------------Выводит на экран список факультативных дисциплин и их задач----
----------------------
		System.out.println("Здравствуйте. Спешим сообщить Вам новость,
что с этого года каждый студент должен выполнять не менее " + n + "
дополнительные занятий по факультативным дисциплинам!\nВот список доступных
факультативов:");
		for (int i = 0; i < facultatives.size(); i++) {
			System.out.println("****" + facultatives.get(i).getName());
			calculateSize(facultatives.get(i).getCountTasks());
			System.out.println("К задачам данного факультатива
относят:");
			for ( int j ; this.j < size ; this.j++) {

	System.out.println(tasts_facultatives.get(this.j).getId() + " --- " +
tasts_facultatives.get(this.j).getName() +
tasts_facultatives.get(this.j).getInfo());
			}
		}

		choice();				// Запускает метод выбора
факультативной задачи
		processChoice();		// Запускает метод анализа
```

```
	}

	private void choice() {
//------------Запрашивает ввод не менее n числа задач из списка факультативных задач
		Scanner sc = new Scanner(System.in);
		System.out.println("Введите индекс (-ы) задач которые вам понрвились (в строку через пробел):");
		while ("".equals(id) == true) {
			id = sc.nextLine();
		}

		real_id = id.split("\\s");

		if (real_id.length < n) {
			System.out.println("Вы выбрали меньше " + n + " факультативов. Попробуйте снова.");
			id = "";
			choice();
		}
	}


	private void processChoice() {
//------------Анализирует выбранные задачи и запускает метод корректировки расписания
		for (int i = 0; i < real_id.length; i++) {
			for (int j = 0; j < tasts_facultatives.size(); j++) {
				if (real_id[i].equals(tasts_facultatives.get(j).getId())) {

	raspisanie.recalculation(tasts_facultatives.get(j).getLabWorks(),tasts_facultatives.get(j).getPractWorks(),tasts_facultatives.get(j).getObject());
				}
			}
		}
	}

	public void printFacultative() {
		for (int i = 0; i < real_id.length; i++) {
			for (int j = 0; j < tasts_facultatives.size(); j++) {
				if (real_id[i].equals(tasts_facultatives.get(j).getId())) {
					System.out.println("** " + tasts_facultatives.get(j).getName());
				}
			}
		}
	}
	private void calculateSize(int a) {this.size += a;}

}
```

Tasks.java

```
package WorkWithRaspisaniem;

public class Tasks {
	private int colichestvo;

	public Tasks() {this.colichestvo = 1 + (int) (Math.random() * 4);}

	public Tasks(int n) {this.colichestvo = n;}

	public int getColichestvo() {return colichestvo;}

	public void setColichestvo(int colicestvo) {this.colichestvo -=
colicestvo;}
}
```

Facultative.java

```
package Discipline;

public class Facultative extends Discipline {
	private String name;
	private String info;
	private String id;
	private String object;
	private int count_tasks = 0;

	public Facultative() {}

	public Facultative(String name, int count_tasks) {
		this.name = name;
		this.count_tasks = count_tasks;
	}

	public String getName() {return name;}

	public String getInfo() {return info;}

	public String getId() {return id;}

	public String getObject() {return object;}

	public int getCountTasks() {return count_tasks;}

	public int getLabWorks() {return 0;}

	public int getPractWorks() {return 0;}
}
```

Facultative.java

```
package Discipline;

public class Facultative extends Discipline {
```

```
	private String name;
	private String info;
	private String id;
	private String object;
	private int count_tasks = 0;

	public Facultative() {}

	public Facultative(String name, int count_tasks) {
		this.name = name;
		this.count_tasks = count_tasks;
	}

	public String getName() {return name;}

	public String getInfo() {return info;}

	public String getId() {return id;}

	public String getObject() {return object;}

	public int getCountTasks() {return count_tasks;}

	public int getLabWorks() {return 0;}

	public int getPractWorks() {return 0;}
}
```

Obligatory.java

```
package Discipline;

public class Obligatory extends Discipline {
	private String name;
	private String[] lab_topic;
	private String[] pract_topic;

	public Obligatory(String name,String[] lap_topic,String[] pract_topic)
{
		this.name = name;
		this.lab_topic = lap_topic;
		this.pract_topic = pract_topic;
	}

	public String getName() {return name;}

	public String[] getLabTopic() {return lab_topic;}

	public String[] getPractTopic() {return pract_topic;}

	private void printTopic(String[] topic, int n) {
		for (int i = 0; i < n; i++) {
			System.out.println(topic[i]);
		}
	}
```

```
	public void printPredmet(int n, int m) {
		System.out.println("   По предмету " + name + " у вас
запланировано:\n** " + n + " лабораторных работ:");
		printTopic(lab_topic, n);
		System.out.println("** " + m + " практических работ:");
		printTopic(pract_topic, m);
	}
}
```

Результаты работы программы

```
Введите имя:Дима
Введите название группы:19-ИВТ-2
Введите номер студенческого билета :777777
Обязательное расписание для Вас:
   По предмету ШППО у вас запланировано:
** 3 лабораторных работ:
1)Разработка слабо связанного программного кода на основе шаблонов проектирования GoF
2)Spring Framework
3)разработка многопоточных приложений с помощью Concurrency
** 4 практических работ:
1) Основные шаблоны проектирования
2) Порождающие шаблоны проектирования
3) Структурные шаблоны проектирования
4) Поведенческие шаблоны проектирования

   По предмету Машинное обучение у вас запланировано:
** 3 лабораторных работ:
1) Анализ данных с помошью Pandas
2) Линейная регрессия
3) Деревья решений
** 1 практических работ:
1) Метрики качества задач классификации

Здравствуйте. Спешим сообщить Вам новость, что с этого года каждый студент должен выполнять не менее 2 дополнительные занятий по факультативным дисциплинам!
Вот список доступных факультативов:
****Робототехника - прикладная наука, занимающаяся разработкой автоматизированных технических систем.
К задачам данного факультатива относят:
r486vy --- Машинное зрение и распрознавание лиц в реальном времени. При успешном решении данной задачи Вам зачтут одну лабораторную по предмету Машинное обучение.
r846cu --- Оптимизация и переобучение на примере мктода К-соседей. При успешном решении данной задачи Вам зачтут одну практическую работу по предмету ШППО.
****Алгоритмы и структуры данных.
К задачам данного факультатива относят:
a312bg --- Анализ данных, визуализация классификация При успешном решении данной задачи Вам зачтут одну лабораторную по предмету ШППО.
a764ih --- Работа с OpenCV. При успешном решении данной задачи Вам зачтут одну практическую работу по предмету Машинноме обучение.
Введите индекс (-ы) задач которые вам понрвились (в строку через пробел):
r846cu a312bg
Окончательное расписание для Вас:
   По предмету ШППО у вас запланировано:
** 2 лабораторных работ:
1)Разработка слабо связанного программного кода на основе шаблонов проектирования GoF
2)Spring Framework
** 3 практических работ:
1) Основные шаблоны проектирования
2) Порождающие шаблоны проектирования
3) Структурные шаблоны проектирования

   По предмету Машинное обучение у вас запланировано:
** 3 лабораторных работ:
1) Анализ данных с помошью Pandas
2) Линейная регрессия
3) Деревья решений
** 1 практических работ:
1) Метрики качества задач классификации

** Оптимизация и переобучение на примере мктода К-соседей.
** Анализ данных, визуализация классификация
```