Министерство образования Калининградской области

государственное бюджетное учреждение Калининградской области

профессиональная образовательная организация

«Колледж информационных технологий и строительства»

(ГБУ КО ПОО «КИТиС»)

**Отчет по учебной практике**

УП.01 Разработка модулей программного обеспечения для компьютерных систем

по ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

Специальность 09.02.07 «Информационные системы и программирование»

Сроки прохождения практики:

с «17» ноября 2022 г. по «07» декабря 2022 г.

Место практики ГБУ КО ПОО «КИТиС»

|  |  |
| --- | --- |
| Выполнил: | студент 4 курса,  группы ИСп 19-1  Иванов Илья Денисович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Проверила: | Большакова-Стрекалова Анна Викторовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Калининград, 2022

Содержание

|  |  |
| --- | --- |
| Введение | 3 |
| 1 Решение задач | 4 |
| 1.1 Задача 1 | 4 |
| 1.2 Задача 2 | 6 |
| 1.3 Задача 3 | 12 |
| 1.4 Задача 4 | 21 |
| 1.5 Задача 5 | 25 |
| 1.6 Задача 7 | 26 |
| 1.7 Объединение задач | 28 |
| 2 Разработка АИС «ВУЗы г. Калининграда» | 32 |
| 2.1 ER-диаграмма сущность-связь | 32 |
| 2.2 Техническое задание | 32 |
| 3 Разработка сайта | 35 |
| Заключение | 37 |
| Список использованных источников |  |

Введение

Учебная практика по модулю ПМ 01 «Разработка модулей программного обеспечения для компьютерных систем» проходит на базе колледжа.

Целями практики являются закрепление знаний в области разработки автоматизированных информационных систем, разработки мобильных приложений, развитие профессиональных навыков в области проектирования баз данных и проектирования программных интерфейсов.

Задачами практики являются разработка приложений по работе с матрицами, классами и вложенными классами, работе с циклами, разработки визуального интерфейса, проектирование базы данных и мобильного приложения. Также в ходе работы требуется составить техническое задание и разработать диаграммы.

Решение задач

Задача 1

Найти сумму элементов матрицы, расположенных между первым и вторым положительными элементами каждой строки.

Код программы:

import java.util.Arrays;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

int width = new Scanner(System.in).nextInt();

int hight = new Scanner(System.in).nextInt();

mean(matrixFilling(width, hight));

}

public static int[][] matrixFilling(int width, int hight) {

int[][] array = new int[hight][width];

for (int i = 0; i < hight; i++) {

for (int j = 0; j < width; j++) {

array[i][j] =(int) (Math.random() \* 99)-44;

}

}

System.out.println(Arrays.deepToString(array));

return array;

}

public static void mean(int matrix[][]) {

int sum = 0;

int count = 0;

for (int i = 0; i < matrix.length; i++) {

for (int j = 0; j < matrix[i].length; j++) {

if(count == 1 && matrix[i][j] < 0){

sum += matrix[i][j];

}

if(matrix[i][j] > 0){

count++;

}

}

System.out.println(sum);

sum = 0;

count = 0;

}

}

}

Блок-схема

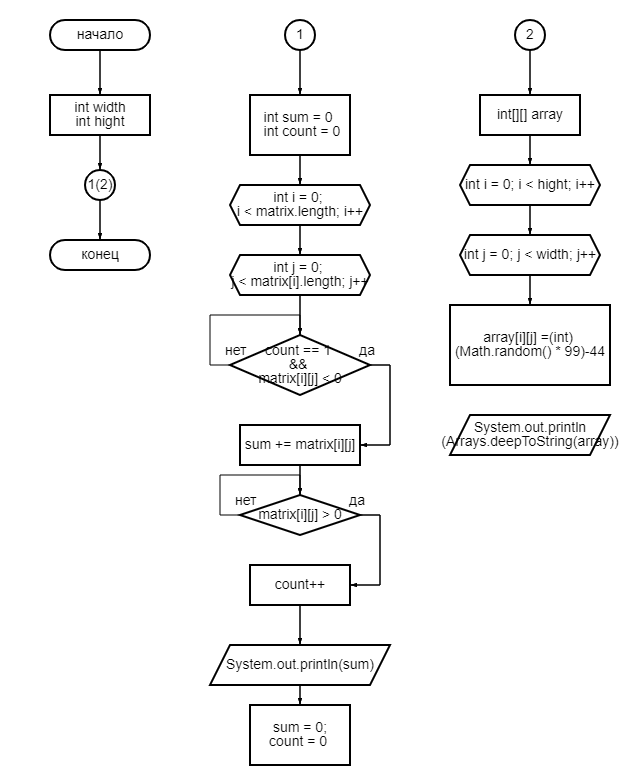


Рисунок 1 – Блок-схема к задаче 1

Скриншот результата

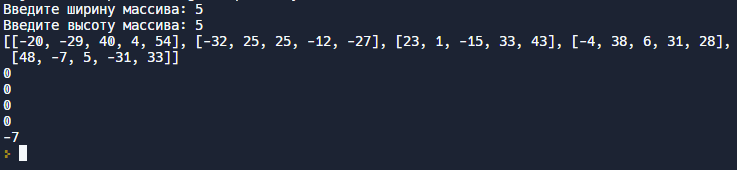


Рисунок 2у – Результат выполнения задачи 1

Задача 2

Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. В каждом классе, обладающем информацией, должно быть объявлено несколько конструкторов. Вариант 8: Car: id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер.

Создать массив объектов. Вывести:

a) список автомобилей заданной марки;

b) список автомобилей заданной модели, которые эксплуатируются больше n лет;

c) список автомобилей заданного года выпуска, цена которых больше указанной.

Код программы:

Класс Main:

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

CarList list = new CarList();

list.addCar(new Car(

1,

2016,

16200,

"BMW",

"Gran Tourer 216d F46",

"White",

"AB 123 FG (AL)"));

list.addCar(new Car(

2,

2016,

18009,

"BMW",

"X1",

"White",

"S 578 IL (A)"));

list.addCar(new Car(

3,

2021,

22281,

"Nissan",

"Qashqai II",

"Gray",

"C 5096 МК (BG)"));

list.addCar(new Car(

4,

2015,

24450,

"Infiniti",

"Q50",

"Brown",

"MAA 001 (CY)"));

list.addCar(new Car(

5,

2019,

15734,

"Kia",

"Rio",

"Gray",

"307 RTB (EST)"));

list.addCar(new Car(

6,

1993,

920,

"Hyundai Pony",

"X1",

"Red",

"1234 AB-7 (BY)"));

list.addCar(new Car(

7,

2020,

15511,

"LADA",

"Vesta SW Cross",

"Blue",

"M 777 CK 777 (RU)"));

System.out.println("\nCar list filtered with 'BMW' brand:");

ArrayList<Car> listBrand = list.getCarBrandFiltered("BMW");

for (Car car : listBrand) {

System.out.println(car);

}

System.out.println("\nCar list filtered with 'X1' model and used more than 10 years:");

ArrayList<Car> listModel = list.getCarModelFiltered("X1", 2012);

for (Car car : listModel) {

System.out.println(car);

}

System.out.println("\nCar list filtered with 2016 year of manufacture and price more than $18k:");

ArrayList<Car> listYearPrice = list.getCarYearPriceFiltered(2016, 18000);

for (Car car : listYearPrice) {

System.out.println(car);

}

}

}

Класс Car:

public class Car {

private int id;

private int carYear;

private int carPrice;

private String carBrand;

private String carModel;

private String carColor;

private String carNumber;

public String getCarBrand() {

return carBrand;

}

public String getCarModel() {

return carModel;

}

public int getCarYear() {

return carYear;

}

public int getCarPrice() {

return carPrice;

}

public Car(

int id,

int carYear,

int carPrice,

String carBrand,

String carModel,

String carColor,

String carNumber) {

this.id = id;

this.carYear = carYear;

this.carPrice = carPrice;

this.carBrand = carBrand;

this.carModel = carModel;

this.carColor = carColor;

this.carNumber = carNumber;

}

public String toString(){

return String.format("id: #%s, ", id) +

String.format("car brand: %s, ",carBrand) +

String.format("model: %s, ", carModel) +

String.format("year: %s, ", carYear) +

String.format("price: $%s ", carPrice) +

String.format("color: %s ", carColor) +

String.format("number: %s ", carNumber);

}

}

Класс CarList:

import java.util.ArrayList;

public class CarList {

private ArrayList<Car> carList = new ArrayList<>();

public void addCar(Car car){

carList.add(car);

}

public ArrayList<Car> getCarBrandFiltered (String carBrand){

ArrayList<Car> carArrayList = new ArrayList<>();

for (Car car : carList){

if (car.getCarBrand() == carBrand) carArrayList.add(car);

}

return carArrayList;

}

public ArrayList<Car> getCarModelFiltered (String carModel, int carYear){

ArrayList<Car> carArrayList = new ArrayList<>();

for (Car car : carList){

if (car.getCarModel() == carModel && car.getCarYear() < carYear)

carArrayList.add(car);

}

return carArrayList;

}

public ArrayList<Car> getCarYearPriceFiltered(int carYear, int carPrice){

ArrayList<Car> carArrayList = new ArrayList<>();

for (Car car : carList){

if (car.getCarYear() == carYear && car.getCarPrice() > carPrice) {

carArrayList.add(car);

}

}

return carArrayList;

}

}

Скриншот результата

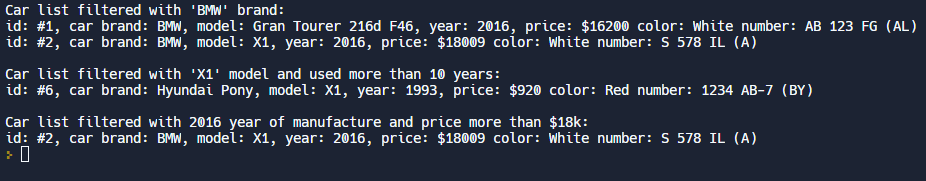


Рисунок 3 – Результат выполнения задачи 2

Задача 3

Создать объект класса Квадрат, используя классы Точка, Отрезок. Методы: задание размеров, растяжение, сжатие, поворот, изменение цвета.

Код программы:

Класс Main:

public class Main {

public static void main(String[] args) {

Point pointX1 = new Point(1, 2);

Point pointX2 = new Point(3, 4);

Point pointX3 = new Point(5, 6);

Point pointX4 = new Point(7, 8);

Line lineAB = new Line(pointX1, pointX2);

Line lineBC = new Line(pointX2, pointX3);

Line lineCD = new Line(pointX3, pointX4);

Line lineDA = new Line(pointX4, pointX1);

Square square = new Square.SquareBuilder(lineAB, lineBC, lineCD, lineDA)

.color("Black").name("Test").build();

System.out.println(square);

}

}

Класс Line:

import java.util.Objects;

public class Line {

private String name;

private Point pointA;

private Point pointB;

public Line(Point pointA, Point pointB) {

this(pointA, pointB, "Unknown");

}

public Line(Point pointA, Point pointB, String name) {

this.pointA = pointA;

this.pointB = pointB;

this.name = name;

}

public void setPointA(Point pointA) {

this.pointA = pointA;

}

public void setPointB(Point pointB) {

this.pointB = pointB;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Point getPointA() {

return pointA;

}

public Point getPointB() {

return pointB;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (!(o instanceof Line)) return false;

Line line = (Line) o;

return Objects.equals(name, line.name) &&

Objects.equals(pointA, line.pointA) &&

Objects.equals(pointB, line.pointB);

}

@Override

public int hashCode() {

return Objects.hash(name, pointA, pointB);

}

@Override

public String toString() {

return "Line{" +

"name='" + name + '\'' +

", pointA=" + pointA +

", pointB=" + pointB +

'}';

}

}

Класс Point:

import java.util.Objects;

public class Point {

private String name;

private double x;

private double y;

public Point(double x, double y) {

this(x, y, "Unknown");

}

public Point(double x, double y, String name) {

this.name = name;

this.x = x;

this.y = y;

}

public void setName(String name) {

this.name = name;

}

public void setX(double x) {

this.x = x;

}

public String getName() {

return name;

}

public double getX() {

return x;

}

public double getY() {

return y;

}

public void setY(double y) {

this.y = y;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (!(o instanceof Point)) return false;

Point point = (Point) o;

return Double.compare(point.x, x) == 0 &&

Double.compare(point.y, y) == 0 &&

Objects.equals(name, point.name);

}

@Override

public int hashCode() {

return Objects.hash(name, x, y);

}

@Override

public String toString() {

return "Point{" +

"name='" + name + '\'' +

", x=" + x +

", y=" + y +

'}';

}

}

Класс Square:

import java.util.Objects;

public class Square {

private String name;

private String color;

private Line lineAB;

private Line lineBC;

private Line lineCD;

private Line lineDA;

public Square(Line lineAB, Line lineBC, Line lineCD, Line lineDA) {

this.lineAB = lineAB;

this.lineBC = lineBC;

this.lineCD = lineCD;

this.lineDA = lineDA;

}

public static class SquareBuilder {

private String name;

private String color;

private Line lineAB;

private Line lineBC;

private Line lineCD;

private Line lineDA;

public SquareBuilder(Line lineAB, Line lineBC, Line lineCD, Line lineDA) {

this.lineAB = lineAB;

this.lineBC = lineBC;

this.lineCD = lineCD;

this.lineDA = lineDA;

}

public SquareBuilder color(String color) {

this.color = color;

return this;

}

public SquareBuilder name(String name) {

this.name = name;

return this;

}

public Square build() {

return new Square(this);

}

}

private Square(SquareBuilder squareBuilder) {

name = squareBuilder.name;

color = squareBuilder.color;

lineAB = squareBuilder.lineAB;

lineBC = squareBuilder.lineBC;

lineCD = squareBuilder.lineCD;

lineDA = squareBuilder.lineDA;

}

public void changeLines(Line lineAB, Line lineBC, Line lineCD, Line lineDA) {

this.lineAB = lineAB;

this.lineBC = lineBC;

this.lineCD = lineCD;

this.lineDA = lineDA;

}

public void increaseSquere(double percent) {

if (lineAB != null && percent != 0) {

lineAB.getPointA().setX(lineAB.getPointA().getX() \* percent);

lineAB.getPointA().setY(lineAB.getPointA().getY() \* percent);

//the same for each side

}

}

public void compressSquere(double percent) {

if (lineAB != null && percent != 0) {

lineAB.getPointA().setX(lineAB.getPointA().getX() / percent);

lineAB.getPointA().setY(lineAB.getPointA().getY() / percent);

//the same for each side

}

}

public void turnSquere(double percent) {

if (lineAB != null && percent != 0) {

lineAB.getPointA().setX(lineAB.getPointA().getX() + percent);

lineAB.getPointA().setY(lineAB.getPointA().getY() - percent);

//the same for each side

}

}

public void changeColor(String color) {

this.color = color;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

public Line getLineAB() {

return lineAB;

}

public void setLineAB(Line lineAB) {

this.lineAB = lineAB;

}

public Line getLineBC() {

return lineBC;

}

public void setLineBC(Line lineBC) {

this.lineBC = lineBC;

}

public Line getLineCD() {

return lineCD;

}

public void setLineCD(Line lineCD) {

this.lineCD = lineCD;

}

public Line getLineDA() {

return lineDA;

}

public void setLineDA(Line lineDA) {

this.lineDA = lineDA;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (!(o instanceof Square)) return false;

Square square = (Square) o;

return Objects.equals(name, square.name) &&

Objects.equals(color, square.color);

}

@Override

public int hashCode() {

return Objects.hash(name, color);

}

@Override

public String toString() {

return "Square{" +

"name='" + name + '\'' +

", color='" + color + '\'' +

", lineAB=" + lineAB +

", lineBC=" + lineBC +

", lineCD=" + lineCD +

", lineDA=" + lineDA +

'}';

}

}

Скриншот результата

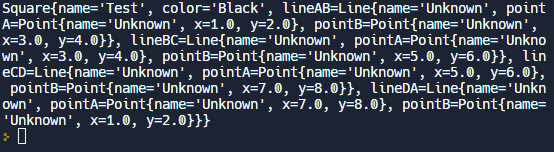


Рисунок 4 – Результат выполнения задачи 3

Задача 4

Создать класс City с внутренним классом, с помощью объектов которого можно хранить информацию о проспектах, улицах, площадях.

Код программы:

import java.util.ArrayList;

public class City {

public static void main(String[] args) {

ArrayList<CityObject> City = new ArrayList<>();

City.add(new Street("Гайдара"));

City.add(new Street("Красная"));

City.add(new Square("Площадь Победы"));

City.add(new Prospect("Советский"));

City.add(new Prospect("Московский"));

print(City);

}

public static void print(ArrayList<CityObject> city) {

System.out.println("В городе:");

for (CityObject object: city) {

System.out.println(object.toString());

}

}

}

class CityObject {

String name;

String info = "";

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

public String getInfo() {

return info;

}

public void setInfo(String info) {

this.info = info;

}

@Override

public String toString() {

return "CityObject{" +

"name='" + name + '\'' +

'}';

}

public void printInfo() {

System.out.println(info);

}

public CityObject(String name) {

this.name = name;

}

}

class Street extends CityObject {

public Street(String name) {

super(name);

}

public Street(String name, String info) {

super(name);

this.info = info;

}

@Override

public String toString() {

if (info.isEmpty()) {

return "Улица - " + name;

} else {

return "Улица - " + name + " - " + info;

}

}

}

class Prospect extends CityObject {

public Prospect(String name) {

super(name);

}

public Prospect(String name, String info) {

super(name);

this.info = info;

}

@Override

public String toString() {

if (info.isEmpty()) {

return "Проспект - " + name;

} else {

return "Проспект - " + name + " - " + info;

}

}

}

class Square extends CityObject {

public Square(String name) {

super(name);

}

public Square(String name, String info) {

super(name);

this.info = info;

}

@Override

public String toString() {

if (info.isEmpty()) {

return "Площадь - " + name;

} else {

return "Площадь - " + name + " - " + info;

}

}

}

Скриншот результата

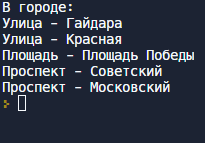


Рисунок 5 – Результат выполнения задачи 4

Задача 5

Удалить из текста его часть, заключенную между двумя символами, которые вводятся (например, между скобками «(» и «)» или между звездочками «\*» и т. п.).

Код программы:

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Main {

public static void main(String[] args) {

System.out.print("Enter your text here: ");

Scanner scanner = new Scanner(System.in);

String s = scanner.nextLine();

String [] str = s.split(" ");

StringBuffer sb = new StringBuffer();

Pattern p1 = Pattern.compile("^[(].\*[)]$");

for (String string : str) {

Matcher m = p1.matcher(string);

if(!m.matches()){

sb.append(string);

sb.append(" ");

}

}

System.out.println(sb);

}

}

Скриншот результата

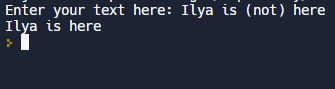


Рисунок 6 – Результат выполнения задачи 5

Задача 7

Код программы:

import javax.swing.\*;

import java.awt.\*;

public class Main extends JFrame {

Main(String s) {

super(s);

setLayout(null);

setSize(540, 540);

setVisible(true);

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

}

public void paint(Graphics g) {

g.setColor(Color.GRAY);

g.fillOval(175,345, 100, 130);

g.fillOval(190, 260, 70, 90);

g.fillOval(180, 470, 40, 10);

g.fillOval(230, 470, 40, 10);

g.fillOval(130, 450, 70, 12);

g.setColor(Color.BLACK);

g.fillOval(235,285, 10, 10);

g.drawOval(232,282, 15, 15);

g.fillOval(210,285, 10, 10);

g.drawOval(207,282, 15, 15);

g.fillOval(205,370, 40, 60);

g.fillOval(220,300, 10, 10);

g.drawLine(225, 305, 160, 295);

g.drawLine(225, 305, 290, 295);

g.drawLine(225, 305, 160, 305);

g.drawLine(225, 305, 290, 305);

g.drawLine(225, 305, 160, 315);

g.drawLine(225, 305, 290, 315);

int[] tr1 = {190, 190, 202};

int[] tr2 = {300, 240, 270};

int[] tr3 = {260, 260, 248};

g.setColor(Color.GRAY);

g.fillPolygon(tr1, tr2, 3);

g.fillPolygon(tr3, tr2, 3);

}

public static void main(String[] args) {

new Main("Кот");

}

}

Скриншот результата

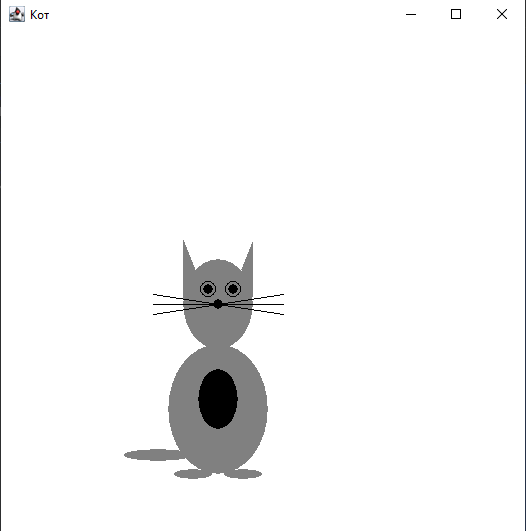


Рисунок 7 – Результат выполнения задачи 7

Объединение задач

Код программы:

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.net.URL;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.JTextField;

public class Main extends JFrame {

private JTextField textField;

public Main() {

super("Week 1");

createGUI();

}

public void createGUI() {

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel panel = new JPanel();

panel.setLayout(new FlowLayout());

ActionListener actionListener = new TestActionListener();

JButton button1 = new JButton("Задание 1");

button1.setActionCommand("https://replit.com/@iahld56pro/Task-1#Main.java");

panel.add(button1);

button1.addActionListener(actionListener);

JButton button2 = new JButton("Задание 2");

button2.setActionCommand("https://replit.com/@iahld56pro/Task-2#Main.java");

panel.add(button2);

button2.addActionListener(actionListener);

JButton button3 = new JButton("Задание 3");

button3.setActionCommand("https://replit.com/@iahld56pro/Task-3#Main.java");

panel.add(button3);

button3.addActionListener(actionListener);

JButton button4 = new JButton("Задание 4");

button4.setActionCommand("https://replit.com/@iahld56pro/Task-4#Main.java");

panel.add(button4);

button4.addActionListener(actionListener);

JButton button5 = new JButton("Задание 5");

button5.setActionCommand("https://replit.com/@iahld56pro/Task-5#Main.java");

panel.add(button5);

button5.addActionListener(actionListener);

JButton button7 = new JButton("Задание 7");

button7.setActionCommand("https://replit.com/@iahld56pro/Task-7#Main.java");

panel.add(button7);

button7.addActionListener(actionListener);

getContentPane().add(panel);

setPreferredSize(new Dimension(320, 200));

}

public class TestActionListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

try {

Desktop.getDesktop().browse(new URL(e.getActionCommand()).toURI());

} catch (Exception ex) {

System.out.println(ex.toString());

}

}

}

public static void main(String[] args) {

javax.swing.SwingUtilities.invokeLater(new Runnable() {

public void run() {

JFrame.setDefaultLookAndFeelDecorated(true);

Main frame = new Main();

frame.pack();

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

});

}

}

Скриншот результата

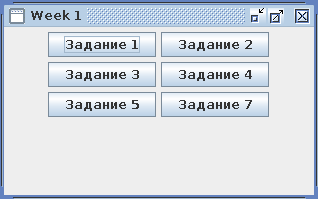


Рисунок 8 – Результат выполнения объединения задач

Разработка АИС «ВУЗы г. Калининграда»

ER-диаграмма сущность-связь

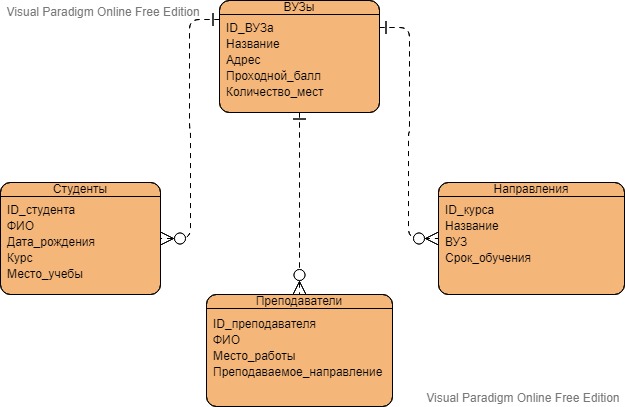


Рисунок 9 – ER-диаграмма сущность-связь

Техническое задание

Основания для разработки

Для выполнения задания учебной практики «Неделя 2» необходимо разработать базу данных «ВУЗы г. Калининграда» и визуальный интерфейс для работы с базой.

Назначение и область применения

Область применения программы – автоматизация поиска ВУЗов. Назначение программы – облегчить взаимодействие с данными, ускорить поиск и выбор ВУЗа для поступления.

Требования к функциональным характеристикам

Необходим функционал выборки из базы данных списка ВУЗов, количества мест для поступления, направления обучения и в каких ВУЗах их преподают, а также списка преподавателей и уже обучающихся студентов. Приложение должно позволять добавлять, удалять и обновлять данные. При входе в приложение должно открываться окно авторизации. Реализовать роль админа и пользователя. Пользователю должна быть недоступна таблица преподавателей.

Требования к надежности

Надёжность программы зависит от:

− уровня подготовки пользователей;

− используемых аппаратных и программных средств;

− соблюдения условий эксплуатации программы;

− сложности пароля администраторов.

От надежности зависит общая производительность и эффективность программы. Программа должна выполнять поставленные перед ней задачи.

Требования к транспортированию и хранению

Требования к серверной комнате установлены в госте TIA/EIA-569.

Требования к составу и параметрам технических средств

Рекомендуемые системные требования компьютера пользователя и сервера:

− Операционная система: Microsoft Windows 10;

− Процессор: Pentium 4 2.0Hz;

− Оперативная память: 40 МБ;

− Свободное место на жестком диске: 200 КБ;

Требования к исходным кодам и языкам программирования

Знания языка Java, средства разработки интерфейса WPF и СУБД Microsoft SQL Server.

Условия эксплуатации

Надежность программы также зависит от условий ее эксплуатации, она должна производиться на исправном компьютере, не содержащем вирусов, и других вредоносных программ.

Требования к квалификации и численности персонала

Минимальный уровень знаний пользователя ПК:

1. Использовать манипулятор-мышь для работы с элементами графического пользовательского интерфейса;

2. Пользователь должен уметь работать в операционной системе;

3. Системный администратор должен поддерживать работу серверной части проекта.

Разработка сайта

Для разработки сайта был использован сайт Google Sites, в котором есть все необходимые инструменты и шаблоны для создания сайта с простейшим функционалом. Этих инструментов хватило для разработки сайта, скриншоты с которого прикреплены ниже.

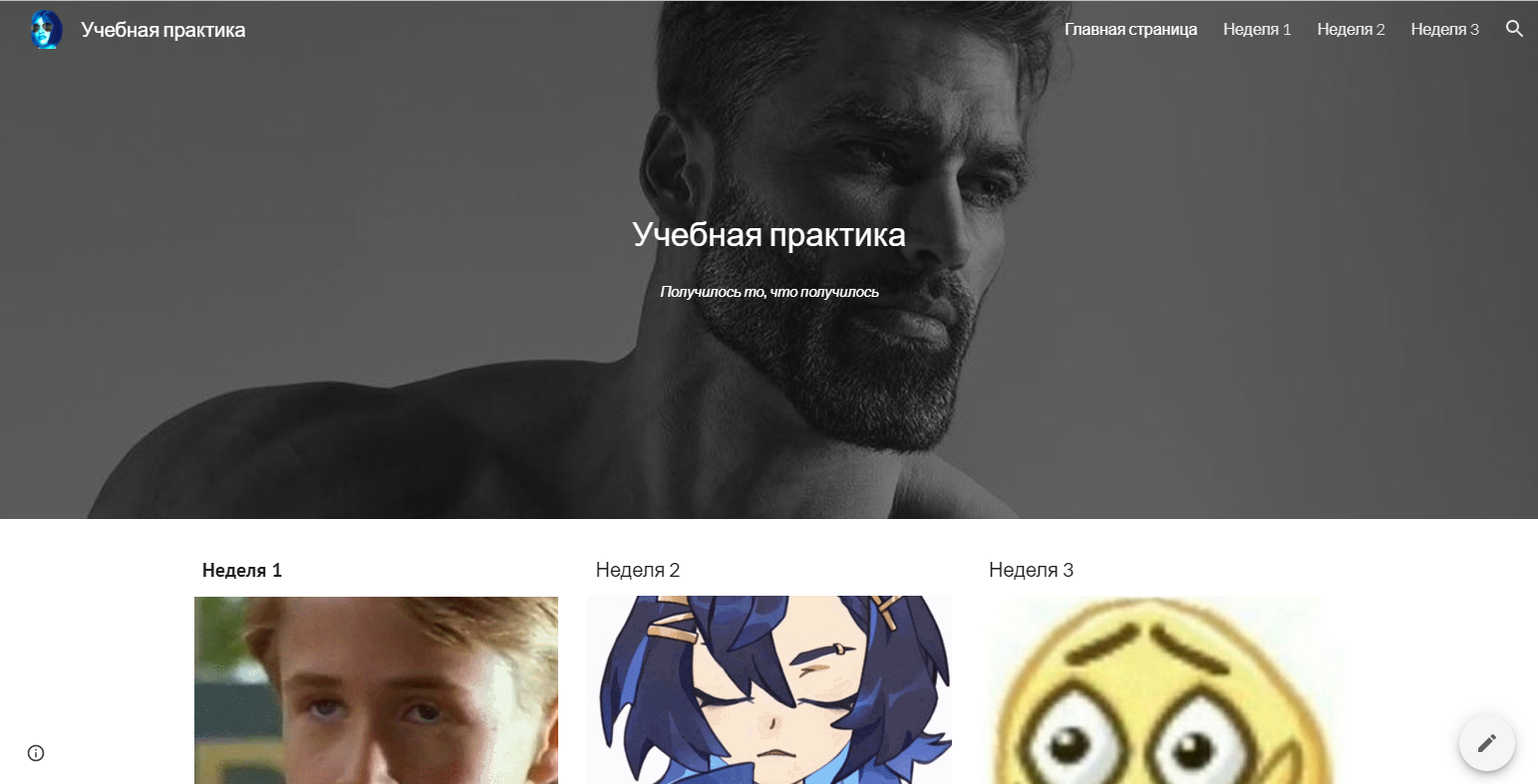


Рисунок 10 – Главная страница сайта

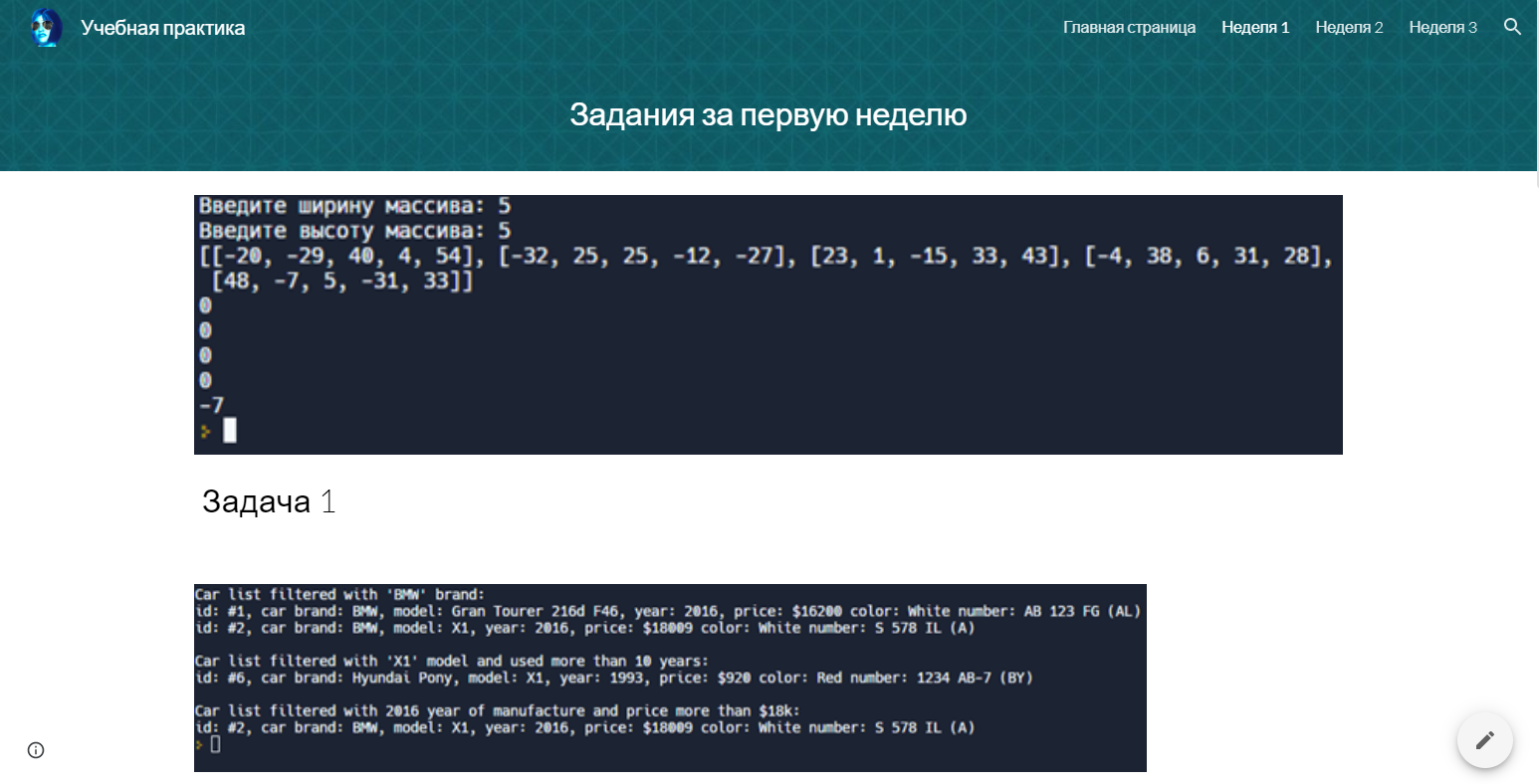


Рисунок 11 – Страница с заданиями за неделю 1

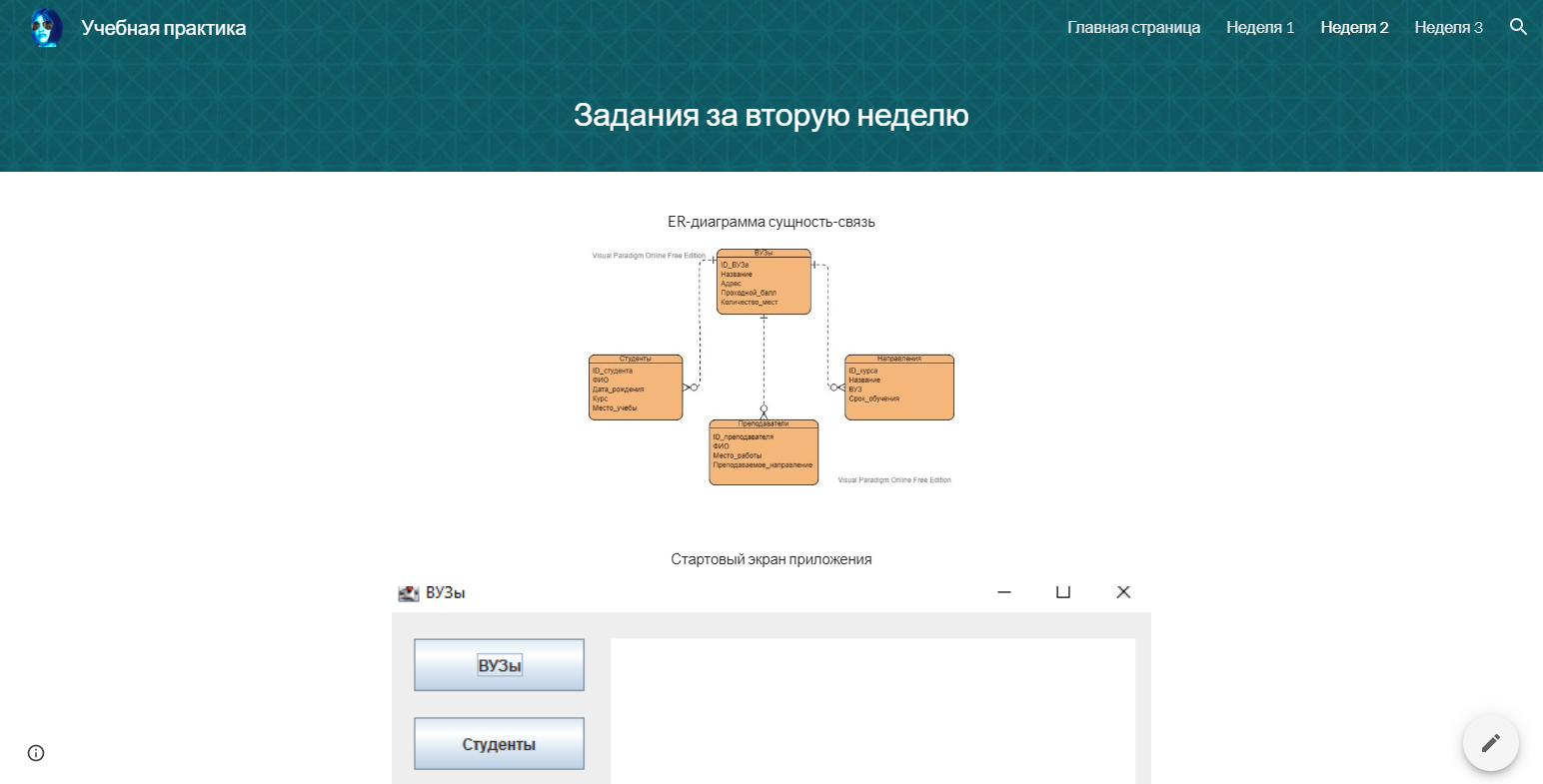


Рисунок 12 – Страница с заданиями за неделю 2

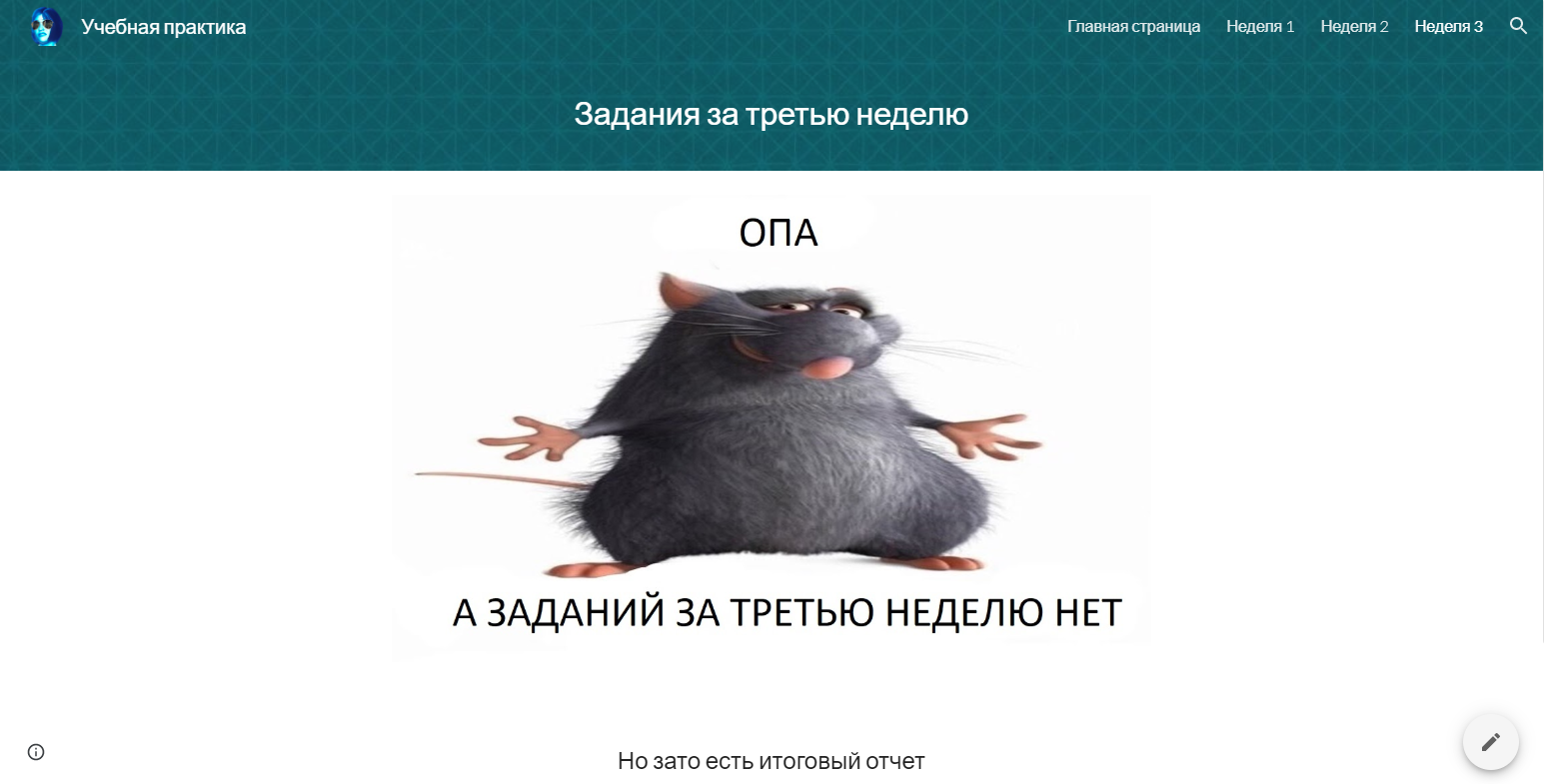


Рисунок 13 – Страница с заданиями за неделю 3

Заключение

В ходе прохождения практики были освоены навыки по разработке автоматических информационных систем, закреплены знания в области проектирования программных интерфейсов и баз данных.

Цели и задачи практики достигнуты. Разработаны алгоритмы решения задач с матрицами и циклами, разработана АИС «ВУЗы г. Калининграда», в которую входит база данных с системой управления SQLiteStudio, а также десктопное приложение. Для демонстрации выполненных заданий был сконструирован сайт с использованием хостинга Google Site.

Список использованных источников

1. Google Sites – <https://sites.google.com/>
2. Онлайн-редактор блок-схем – <https://programforyou.ru/>
3. учебник «Java – Методы программирования»
4. Семакин И. Г. – «Основы алгоритмизации и программирования»