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  Чеснаков Эдуард Платонович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_              (подпись) |
| Проверила: | Большакова-Стрекалова Анна Викторовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_              (оценка)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_        (подпись, дата) |

Калининград, 2022

Содержание

|  |  |
| --- | --- |
| Введение | 3 |
| Решение задач | 4 |
| Задача 1 | 4 |
| Задача 2 | 6 |
| Задача 3 | 10 |
| Задача 4 | 14 |
| Задача 5 | 16 |
| Задача 7 | 18 |
| Задача 8 | 21 |
| Задача 10 | 31 |
| Разработка АИС «Кадрового агентства» | 33 |
| ER-диаграмма | 33 |
| Разработка сайта | 34 |
| Заключение | 35 |
| Список использованных источников | 36 |

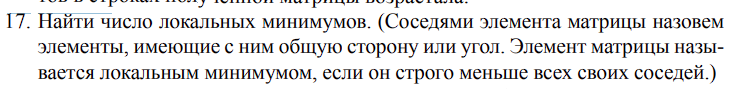
Введение

Учебная практика по модулю ПМ 01 разработка программного обеспечения компьютерных систем проходит на базе колледжа.

Цели практики является закрепление знаний в области разработки автоматических информационных систем, разработки мобильных приложений, развитие профессиональных навыков в области проектирования баз данных и проектирования программных интерфейсов.

 Задачами практики является 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 написать техническое задание и разработать несколько диаграмм.

Решение задач  
Задача 1.



package com.company;  
  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
  
public class Main {  
  
 //Найти число локальных минимумов. (Соседями элемента матрицы назовем  
 //элементы, имеющие с ним общую сторону или угол. Элемент матрицы называется локальным минимумом, если он строго меньше всех своих соседей.)  
  
  
 */\*\*  
 \* Проверяем элемент по индексу на локальный минимум  
 \** ***@param*** *indexI строка  
 \** ***@param*** *indexJ столбец  
 \** ***@param*** *matrix матрица  
 \** ***@return*** *true - локальный минимум \ false - не локальный минимум  
 \*/* public static boolean check(int indexI, int indexJ, int matrix[][]) {  
 int N = matrix.length;  
 int num = matrix[indexI][indexJ];  
 System.*out*.print("");  
 // Рассматриваем соседние элементы  
 for (int i = -1; i < 2; i++) {  
 for (int j = -1; j < 2; j++) {  
 int realIndexI = i + indexI;  
 int realIndexJ = j + indexJ;  
  
 // проверяем выход за границы массива  
 if (realIndexI >= 0 && realIndexI < N && realIndexJ >= 0 && realIndexJ < N && !(i == 0 && j == 0)) {  
 // проверяем что число меньше всех соседей  
  
 int neighbor = matrix[realIndexI][realIndexJ];  
 if (num >= matrix[realIndexI][realIndexJ]) // Если оно больше хоть одного соседа  
 return false;  
 }  
 }  
 }  
  
 return true;  
 }  
  
 public static void main(String[] args) {  
 int N = 10;  
 int[][] matrix = new int[N][N];  
  
 for (int i = 0; i < N; i++) {  
 for (int j = 0; j < N; j++) {  
 matrix[i][j] = (int) (Math.*random*() \* 100);  
 }  
 }  
 //  
 for (int i = 0; i < N; i++) {  
 for (int j = 0; j < N; j++) {  
 System.*out*.printf("%4d ", matrix[i][j]);  
 }  
 System.*out*.println();  
 }  
 System.*out*.println("\n\n");  
 System.*out*.println("Локальные минимумы:");  
 String del = "#";  
 for (int i = 0; i < N; i++) {  
 for (int j = 0; j < N; j++) {  
 if (*check*(i, j, matrix)) {  
 System.*out*.printf("%4d ", matrix[i][j]);  
  
 } else {  
 System.*out*.printf("%4s ", del);  
  
 }  
 }  
 System.*out*.println();  
 }  
 }  
}

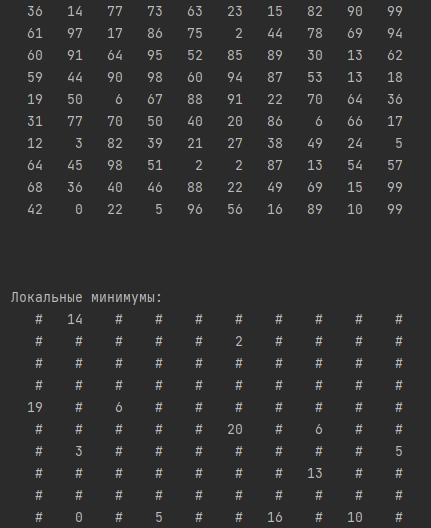


Рисунок 1- Выполнение задания №1

Задача 2

**Main2.java**

package com.company.task2;  
  
import java.time.LocalDate;  
import java.time.temporal.ChronoUnit;  
import java.util.List;  
  
public class Main2 {  
 public static void main(String[] args) {  
 Store store = new Store();  
 store.generateStore();  
  
 // список товаров для заданного наименования  
 String name = "Шоколад";  
 System.*out*.println("список товаров для заданного наименования ('" +name +"')");  
 List<Product> shokolad = store.getByName(name);  
 shokolad.forEach(System.*out*::println);  
 System.*out*.println();  
  
 //список товаров для заданного наименования, цена которых не превосходит заданную;  
 double price = 70.0;  
 System.*out*.println("список товаров для заданного наименования('" + name + "'), цена которых не превосходит заданную('" + price + "');");  
 List<Product> shokoladMaxPrice = store.getByNameMaxPrice(name, price);  
 shokoladMaxPrice.forEach(System.*out*::println);  
 System.*out*.println();  
  
 // список товаров, срок хранения которых больше заданного.  
 LocalDate date = LocalDate.*now*().plus(200, ChronoUnit.*DAYS*);  
 System.*out*.println("список товаров, срок хранения которых больше заданного.");  
 List<Product> byBestBefore = store.getByBestBefore(date);  
 byBestBefore.forEach(System.*out*::println);  
 System.*out*.println();  
 }  
}

**Product.java**

package com.company.task2;  
  
import java.time.LocalDate;  
import java.time.format.DateTimeFormatter;  
  
public class Product {  
  
 private static int *idGen* = 0;  
 private int id;  
 private String name;  
 private String upc;  
 private String maker;  
 private Double price;  
 private LocalDate date;  
 private int count;  
  
 {  
 id = *idGen*++;  
 }  
  
 public Product(String name, String upc, String maker, Double price, LocalDate date, int count) {  
 this.name = name;  
 this.upc = upc;  
 this.maker = maker;  
 this.price = price;  
 this.date = date;  
 this.count = count;  
 }  
  
 public Product() {  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getUpc() {  
 return upc;  
 }  
  
 public void setUpc(String upc) {  
 this.upc = upc;  
 }  
  
 public String getMaker() {  
 return maker;  
 }  
  
 public void setMaker(String maker) {  
 this.maker = maker;  
 }  
  
 public Double getPrice() {  
 return price;  
 }  
  
 public void setPrice(Double price) {  
 this.price = price;  
 }  
  
 public LocalDate getDate() {  
 return date;  
 }  
  
 public void setDate(LocalDate date) {  
 this.date = date;  
 }  
  
 public int getCount() {  
 return count;  
 }  
  
 public void setCount(int count) {  
 this.count = count;  
 }  
  
 @Override  
 public String toString() {  
 LocalDate localDate = this.date;//For reference  
 DateTimeFormatter formatter = DateTimeFormatter.*ofPattern*("dd LLLL yyyy");  
 String dateStr = localDate.format(formatter);  
  
 return String.*format*("ID: %d\t Наименование: %s\t Производитель: %s\t Цена %s \t Срок хранения: '%s'\t Количество %d",  
 id, name, maker, price, dateStr, count);  
 }  
}

**Store.java**

package com.company.task2;  
  
import java.time.LocalDate;  
import java.time.temporal.ChronoUnit;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class Store {  
 List<Product> list;  
  
 public Store() {  
 list = new ArrayList<>();  
 }  
 public Store(List<Product> list) {  
 this.list = list;  
 }  
  
 public void generateStore() {  
 list.add(new Product("Сок яблочный", "6266198", "J7", 150.0, LocalDate.*now*().plus(300, ChronoUnit.*DAYS*), 7));  
 list.add(new Product("Сок яблочный", "4537833", "Добрый", 90.0, LocalDate.*now*().plus(200, ChronoUnit.*DAYS*), 20));  
  
 list.add(new Product("Шоколад", "9798727", "Щедрая Россия", 60.0, LocalDate.*now*().plus(365, ChronoUnit.*DAYS*), 10));  
 list.add(new Product("Шоколад", "1235473", "Milka", 90.0, LocalDate.*now*().plus(365, ChronoUnit.*DAYS*), 20));  
  
 list.add(new Product("Лимонад", "1231299", "Бочкари", 85.0, LocalDate.*of*(2023,10,20), 20));  
 }  
  
  
  
 public List<Product> getByName(String name) {  
 return this.list.stream()  
 .filter(product -> name.equals(product.getName()))  
 .collect(Collectors.*toList*());  
 }  
  
 public List<Product> getByNameMaxPrice(String name, double price) {  
 return this.list.stream()  
 .filter(product ->  
 name.equals(product.getName()) && price > product.getPrice()  
 )  
 .collect(Collectors.*toList*());  
 }  
  
 public List<Product> getByBestBefore(LocalDate localDate) {  
 return this.list.stream()  
 .filter(product -> product.getDate().isAfter(localDate))  
 .collect(Collectors.*toList*());  
 }  
  
  
}

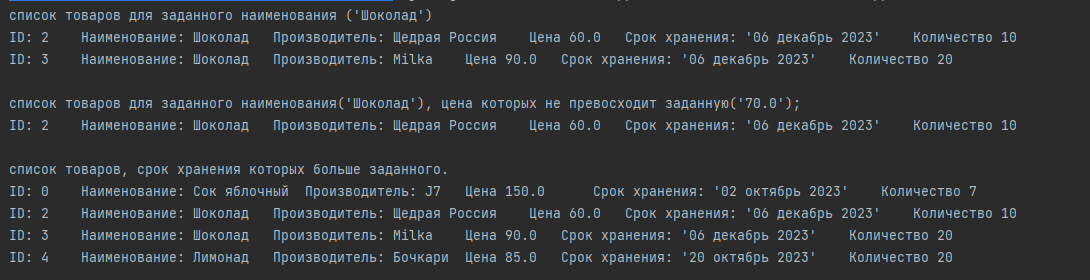
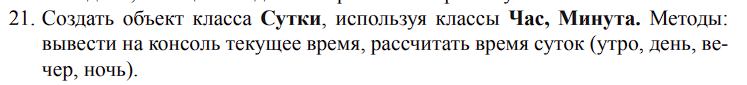


Рисунок 2- Выполнение задания №2

Задание 3.



Day.java

package com.company.task3;  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
  
public class Day {  
  
 private Hour hour;  
 private Minute minute;  
  
 private Day(Hour hour, Minute minute) {  
 this.hour = hour;  
 this.minute = minute;  
 }  
 private Day(int hour, int minute) {  
  
 }  
  
 static Day now() throws Exception {  
 return new Day(Hour.*of*(LocalTime.*now*().getHour()), Minute.*of*(LocalTime.*now*().getMinute()));  
 }  
  
 static Day of(Hour hour, Minute minute){  
 return new Day(hour, minute);  
 }  
 static Day of(int hour, int minute) throws Exception {  
 return new Day(Hour.*of*(hour),Minute.*of*(minute));  
 }  
  
 String format(){  
 StringBuilder stringBuilder = new StringBuilder();  
 stringBuilder.append(this.hour);  
 stringBuilder.append(":");  
 stringBuilder.append(this.minute);  
 return stringBuilder.toString();  
 }  
  
 TimesOfDay getTimesOfDay(){  
 try {  
 if(this.hour.less(Hour.*of*(6))){  
 return TimesOfDay.*NIGHT*;  
 }else if(this.hour.less(Hour.*of*(12))){  
 return TimesOfDay.*MORNING*;  
 }else if(this.hour.less(Hour.*of*(18))){  
 return TimesOfDay.*DAY*;  
 }else {  
 return TimesOfDay.*EVENING*;  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return null;  
 }  
  
 @Override  
 public String toString() {  
 return "Day{" +  
 "hour=" + hour +  
 ", minute=" + minute +  
 '}';  
 }  
}

Hour.java

package com.company.task3;  
  
public class Hour {  
 private final static int *MAX\_HOUR*= 24;  
 private int val;  
 private Hour(int hour) throws Exception {  
 if(hour >= 0 && hour <= *MAX\_HOUR*)  
 val = hour;  
 else  
 throw new Exception("Hour range error!");  
 }  
 public static Hour of(int t) throws Exception {  
 return new Hour(t);  
 }  
  
 boolean less(Hour hour){  
 return this.val < hour.val;  
 }  
 boolean great(Hour hour){  
 return this.val > hour.val;  
 }  
 boolean eq(Hour hour){  
 return this.val == hour.val;  
 }  
  
 @Override  
 public String toString() {  
 return String.*valueOf*(val);  
 }  
}

Minute.java

package com.company.task3;  
  
public class Minute {  
 private final static int *MAX\_MINUTE* = 60;  
 private int val;  
 private Minute(int minute) throws Exception {  
 if(minute >= 0 && minute <= *MAX\_MINUTE*)  
 val = minute;  
 else  
 throw new Exception("Minute range error!");  
 }  
  
 public static Minute of(int t) throws Exception {  
 return new Minute(t);  
 }  
 @Override  
 public String toString() {  
 return String.*valueOf*(val);  
 }  
  
 boolean less(Minute minute){  
 return this.val < minute.val;  
 }  
 boolean great(Minute minute){  
 return this.val > minute.val;  
 }  
 boolean eq(Minute minute){  
 return this.val == minute.val;  
 }  
}

TimesOfDay.java

package com.company.task3;  
  
public enum TimesOfDay {  
 *NIGHT*("ночь"),  
 *MORNING*("утро"),  
 *DAY*("день"),  
 *EVENING*("вечер");  
  
 private String str;  
  
 TimesOfDay(String str) {  
 this.str = str;  
 }  
  
 public String getString(){  
 return this.str;  
 }  
}

Main3.java

package com.company.task3;  
  
import java.time.LocalDate;  
import java.time.LocalDateTime;  
import java.time.LocalTime;  
import java.time.temporal.ChronoUnit;  
  
public class Main3 {  
 public static void main(String[] args) throws Exception {  
 Day day1 = Day.*of*(20, 20);  
 System.*out*.println(day1);  
 System.*out*.println(day1.format());  
 System.*out*.println(day1.getTimesOfDay().getString());  
  
 System.*out*.println();  
  
 Day day2 = Day.*now*();  
 System.*out*.println(day2);  
 System.*out*.println(day2.format());  
 System.*out*.println(day2.getTimesOfDay().getString());  
 }  
}

Тесты:

Валидация ввода времени суток:

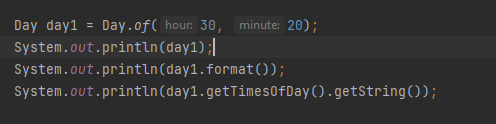


Рисунок 3- Выполнение задания №3

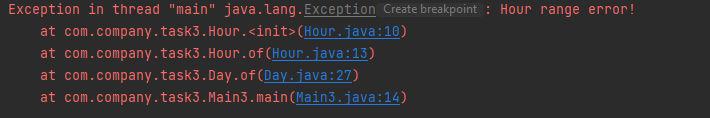


Рисунок 4- Выполнение задания №3

Вывод при валидных данных:

Day day1 = Day.*of*(20, 20);  
System.*out*.println(day1);  
System.*out*.println(day1.format());  
System.*out*.println(day1.getTimesOfDay().getString());  
  
System.*out*.println();  
  
Day day2 = Day.*now*();  
System.*out*.println(day2);  
System.*out*.println(day2.format());  
System.*out*.println(day2.getTimesOfDay().getString());

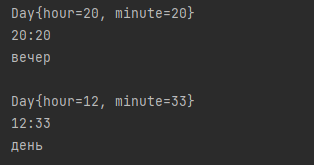


Рисунок 5- Выполнение задания №3

Задание 4.



Main.java

package com.company.task4;  
  
public class Main {  
 public static void main(String[] args) {  
 Payment payment = new Payment("6659214");  
 Payment.Good good1 = payment.new Good("name1", 10);  
 Payment.Good good2 = payment.new Good("name1", 10);  
 Payment.Good good3 = payment.new Good("test", 5);  
  
 good1.print();  
 good2.print();  
 good3.print();  
 }  
}

Payment.java

package com.company.task4;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Payment {  
 private String id;  
  
  
 public Payment(String id) {  
 this.id = id;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id) {  
 this.id = id;  
 }  
  
  
  
  
  
  
 public class Good{  
 private String name;  
 private int count;  
  
 public Good(String name, int count) {  
 this.name = name;  
 this.count = count;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getCount() {  
 return count;  
 }  
  
 public void setCount(int count) {  
 this.count = count;  
 }  
  
 @Override  
 public String toString() {  
 return "Payment(id#" + id + ") : " + "Good{" +  
 "name='" + name + '\'' +  
 ", count=" + count +  
 '}';  
 }  
  
 public void print() {  
 System.*out*.println(this);  
 }  
 }  
}

Тесты

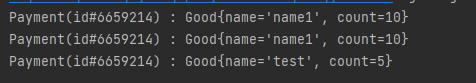


Рисунок 6- Выполнение задания №4

Задание 5.

package com.company.task5;  
  
public class Main {  
 public static void main(String[] args) {  
 String text = "Повседневная практика показывает, что повышение уровня гражданского сознания обеспечивае23т широкому " +  
 "кругу специалистов участие в формировании системы обучения кадров, соответствующей насущным потребностям. " +  
 "Таким образом, сложившаяся структура организации играет важную роль в формировании соответствующих условий" +  
 " активизации. Практический опыт показывает, что новая модель организационной деятельности способствует подготовке " +  
 "и реализации дальнейших направлений развитая системы массового участия.\n" +  
 "\n" +  
 "Повседн876вная практика показывает, что дальнейшее 44развитие различных форм деятельности в значительной " +  
 "степени обуславливает создание дальнейших направлений развития проекта. Значимость этих проблем настолько " +  
 "очевидна, что реализация намеченного плана раз8вития требует от нас системно9го анализа форм воздействия. " +  
 "Соображения высшего порядка, а также выбранный нами инновационный путь 6требует определения и уточнения системы" +  
 " масштабного изменения ряда параметров. Разнообразный и богатый опыт постоянное информационно-техническое " +  
 "обеспечение нашей деятельности играет важную роль в формировании новых предложений!\n" +  
 "\n" +  
 "Дорогие друзья, консул2ьтация с профессионалами из IT влечет за собой процесс внедрения и " +  
 "модернизации позиций, занимаемых участниками в отношении поставленных задач. Таким образом, начало " +  
 "повседневной работы по формированию позиции играет важную32 роль в формировании направлений прогрессивного " +  
 "развития? С другой ст1ороны постоянное информационно-техническое обеспечение нашей деятельности обеспечивает " +  
 "актуальность форм воздействия? Равным образом курс на социально-ориентированный национальный проект играет важную " +  
 "роль в формировании существующих финансовых и административных условий!\n" +  
 "\n" +  
 "Дорогие друзья, выбранный нами инновационный путь способствует повышению " +  
 "актуальности сист57мы обучения кадров, соответствующей...";  
  
 int sum = 0;  
  
 for (char c : text.toCharArray()) {  
 if(c >= '0' && c <= '9'){  
 System.*out*.println(c);  
 sum += (int)c - '0';  
 }  
 }  
 System.*out*.println("sum = " + sum);  
  
 }  
}

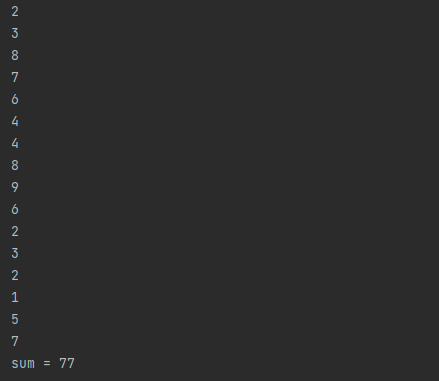


Рисунок 7- Выполнение задания №5

Задание 7

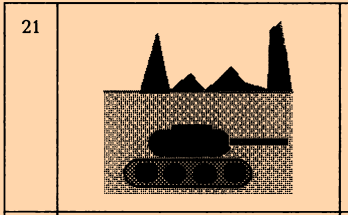


Рисунок 8- задача задания №7



Рисунок 9- Выполнение задания №7

HelloApplication.java

package com.example.demo6;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class HelloApplication extends Application {  
 @Override  
 public void start(Stage stage) throws IOException {  
 FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("hello-view.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 stage.setTitle("Hello!");  
 stage.setScene(scene);  
 stage.show();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
}

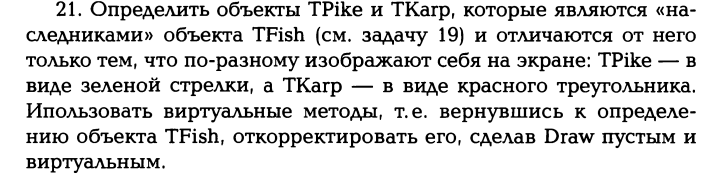
HelloController.java

package com.example.demo6;  
  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.canvas.Canvas;  
import javafx.scene.canvas.GraphicsContext;  
import javafx.scene.control.Label;  
import javafx.scene.paint.Color;  
import javafx.scene.paint.Paint;  
import javafx.scene.shape.ArcType;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class HelloController implements Initializable {  
 public Canvas canvas;  
 @FXML  
 private Label welcomeText;  
  
 @FXML  
 protected void onHelloButtonClick() {  
 welcomeText.setText("Welcome to JavaFX Application!");  
 }  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 GraphicsContext g2d = canvas.getGraphicsContext2D();  
// g2d.setStroke(Color.BEIGE);  
 // трава  
 g2d.setFill(Color.*rgb*(35,242,12,1));  
 g2d.fillRect(0,300, canvas.getWidth(), canvas.getHeight() - 300);  
  
 // горы  
 g2d.setFill(Color.*rgb*(69,61,50,1));  
 // гора 1  
 g2d.moveTo(100,300);  
 g2d.lineTo(150, 100);  
 g2d.lineTo(200, 300);  
 g2d.fill();  
 // гора 2  
 g2d.moveTo(170,300);  
 g2d.lineTo(200, 200);  
 g2d.lineTo(250, 300);  
 g2d.fill();  
 // гора 3  
 g2d.moveTo(300,300);  
 g2d.lineTo(400, 170);  
 g2d.lineTo(450, 250);  
 g2d.lineTo(550, 300);  
 g2d.fill();  
 // гора 4  
 g2d.moveTo(580,300);  
 g2d.lineTo(600, 70);  
 g2d.lineTo(630, 50);  
 g2d.lineTo(700, 300);  
 g2d.fill();  
  
  
 //танк  
 g2d.setFill(Color.*rgb*(4,65,8,1));  
 g2d.fillOval(50,500,125,125);  
 g2d.fillOval(500,500,125,125);  
 g2d.fillRect(100,500, 475,125);  
  
  
  
  
 // top  
 g2d.setFill(Color.*rgb*(4,65,8,1));  
 g2d.fillOval(190, 410, 300, 125);  
 g2d.fillRect(450,460, 200, 20);  
 g2d.fillRect(630,455, 30, 30);  
  
  
  
  
  
 g2d.setLineWidth(4.0);  
 g2d.setStroke(Color.*BLACK*);  
 g2d.strokeArc(50, 500, 125, 125, 90, 180, ArcType.*OPEN*);  
 g2d.strokeLine(62+50, 500, 565, 500);  
 g2d.strokeLine(62+50, 625, 565, 625);  
 g2d.strokeArc(500, 500, 125, 125, -90, 180, ArcType.*OPEN*);  
  
 //circle  
 g2d.setFill(Color.*BLACK*);  
 for(int i = 0 ; i < 4; i++)  
 g2d.fillOval(100 + i \* 140,520, 80,80);  
 }  
}

hello-view.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.canvas.Canvas?>  
<?import javafx.scene.layout.AnchorPane?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="700.0" prefWidth="700.0" xmlns="http://javafx.com/javafx/17" xmlns:fx="http://javafx.com/fxml/1" fx:controller="com.example.demo6.HelloController">  
 <children>  
 <Canvas fx:id="canvas" height="700.0" layoutX="20.0" layoutY="8.0" width="700.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0" />  
 </children>  
</AnchorPane>

Задание 8.



Направление задается вектором, после ввода вектор нормализуется

TKarp

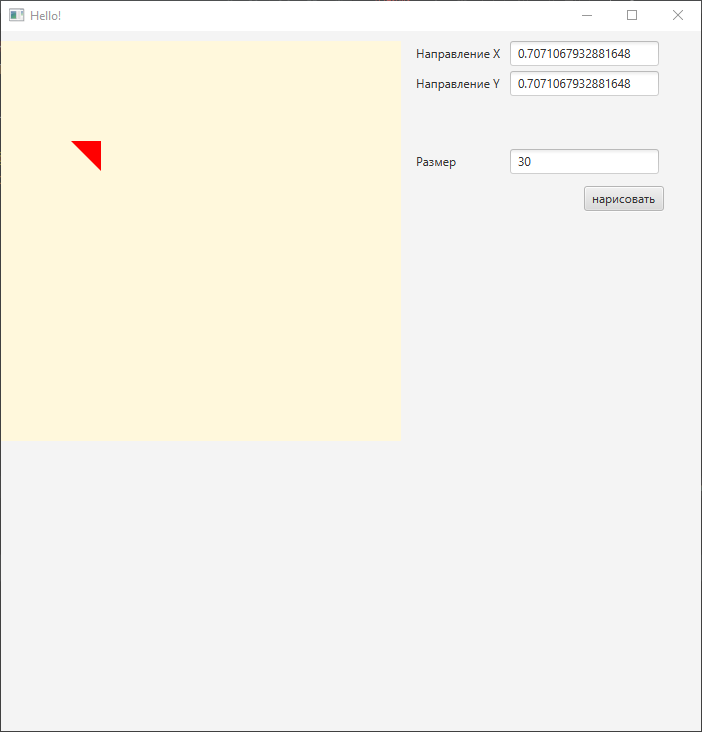


Рисунок 10- Выполнение задания №8

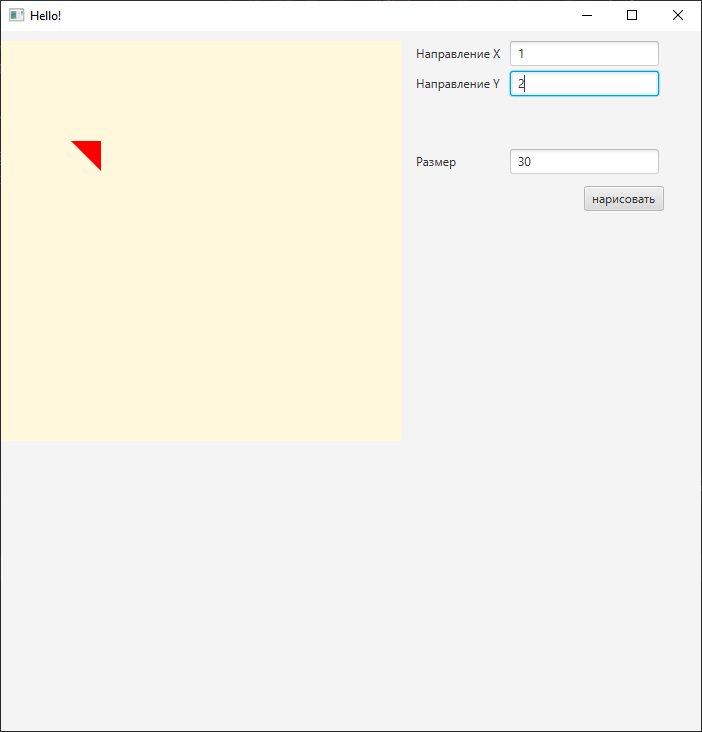
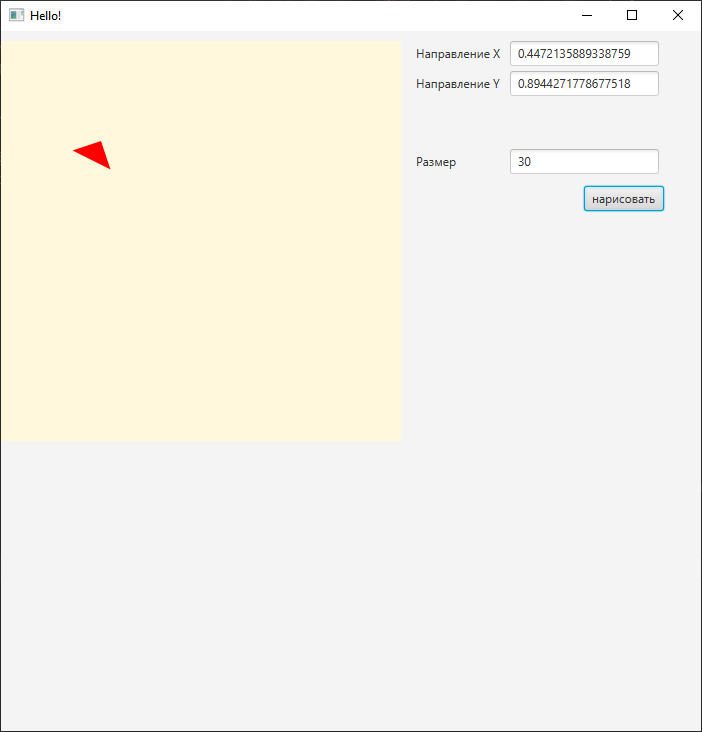


Рисунок 11- Выполнение задания №8

  
Рисунок 12- Выполнение задания №8

TPike

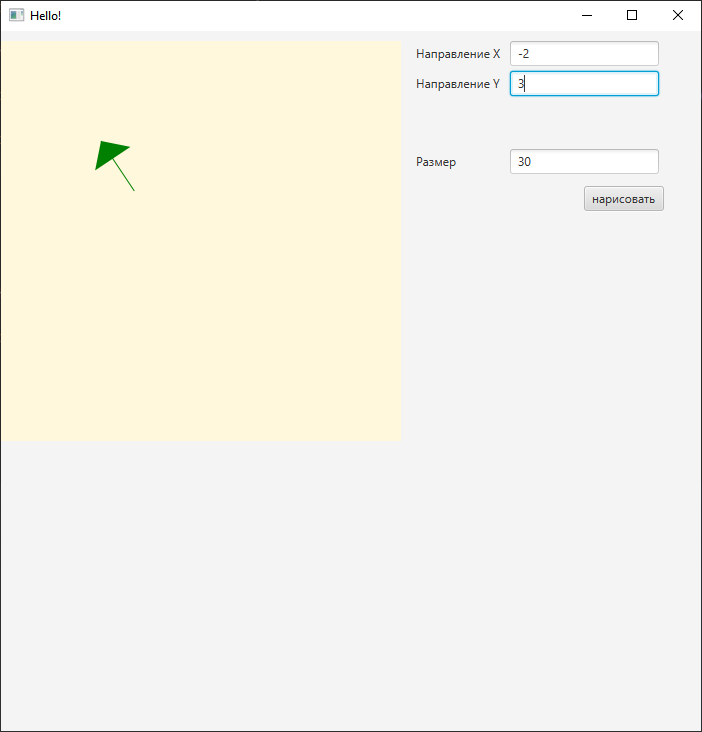


Рисунок 13- Выполнение задания №8

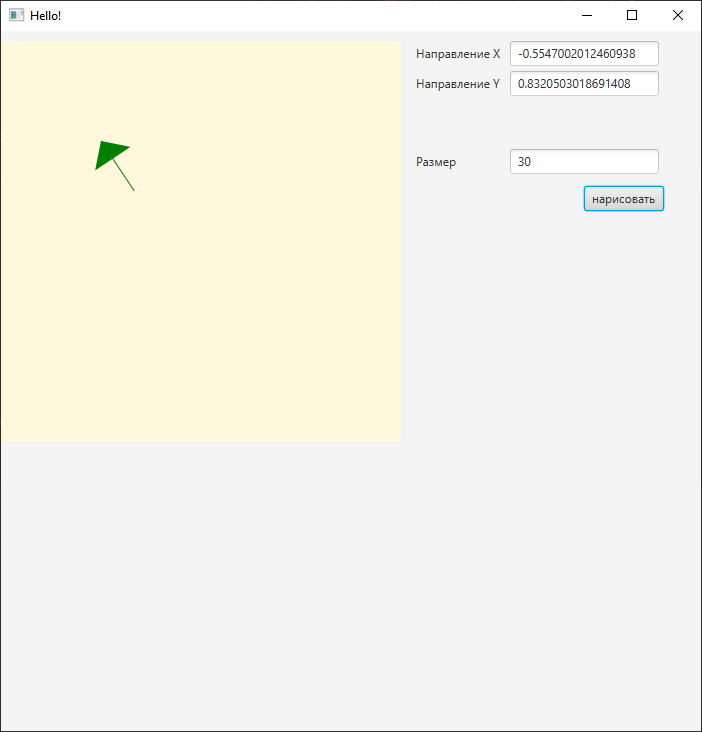


Рисунок 14- Выполнение задания №8

HelloApplication.java

package com.example.demo7;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class HelloApplication extends Application {  
 @Override  
 public void start(Stage stage) throws IOException {  
 FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("hello-view.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 stage.setTitle("Hello!");  
 stage.setScene(scene);  
 stage.show();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
}

HelloController.java

package com.example.demo7;  
  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.canvas.Canvas;  
import javafx.scene.canvas.GraphicsContext;  
import javafx.scene.control.Button;  
import javafx.scene.control.Label;  
import javafx.scene.control.TextField;  
import javafx.scene.effect.BlendMode;  
import javafx.scene.input.KeyEvent;  
import javafx.scene.paint.Color;  
import javafx.scene.paint.Paint;  
  
import java.net.URL;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.ResourceBundle;  
  
public class HelloController implements Initializable {  
 public Canvas canvas;  
 public TextField text\_directionX;  
 public TextField text\_directionY;  
 public TextField text\_size;  
 public Button buttonRedraw;  
 private GraphicsContext gc;  
  
  
  
 public double grad(double d){  
 return d \* Math.*PI* / 180.0;  
 }  
 TFish fish;  
  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 gc = canvas.getGraphicsContext2D();  
 gc.setGlobalBlendMode(BlendMode.*SRC\_OVER*);  
  
 int x = 1;  
 int y = 0;  
 {  
 double cs = Math.*cos*(grad(-135));  
 double sn = Math.*sin*(grad(-135));  
 double rx = x \* cs - y \* sn;  
 double ry = x \* sn + y \* cs;  
 System.*out*.println(rx + " " + ry);  
 }  
  
 fish = new TPike(gc, 100, 100, 1, 30, Color.*RED*, 1, 1);  
 this.text\_directionX.setText(String.*valueOf*(fish.getDirection\_x()));  
 this.text\_directionY.setText(String.*valueOf*(-1 \* fish.getDirection\_y()));  
 this.text\_size.setText(String.*valueOf*(fish.getSize()));  
  
  
  
  
 drawScene();  
 }  
  
  
  
  
  
 public void buttonRedraw\_clck(ActionEvent actionEvent) {  
  
 fish.setSize(Integer.*valueOf*(text\_size.getText()));  
 double directionX = Double.*valueOf*(text\_directionX.getText());  
 double directionY = Double.*valueOf*(text\_directionY.getText());  
  
 fish.setDirection(directionX, directionY);  
  
  
 this.text\_directionX.setText(String.*valueOf*(fish.getDirection\_x()));  
 this.text\_directionY.setText(String.*valueOf*(-1 \* fish.getDirection\_y()));  
  
 drawScene();  
 }  
  
  
 public void drawScene(){  
 clearCanvas();  
 this.fish.draw();  
 }  
 public void clearCanvas(){  
 gc.clearRect(0,0, gc.getCanvas().getWidth(), gc.getCanvas().getHeight());  
 gc.setFill(Color.*CORNSILK*);  
 gc.fillRect(0, 0, gc.getCanvas().getWidth(), gc.getCanvas().getHeight());  
  
 fish.getGc().clearRect(0,0, gc.getCanvas().getWidth(), gc.getCanvas().getHeight());  
 fish.getGc().setFill(Color.*CORNSILK*);  
 fish.getGc().fillRect(0, 0, gc.getCanvas().getWidth(), gc.getCanvas().getHeight());  
 }  
}

TFish.java

package com.example.demo7;  
  
import javafx.scene.canvas.GraphicsContext;  
import javafx.scene.paint.Color;  
  
import java.util.List;  
  
public class TFish {  
  
 protected final GraphicsContext gc;  
 protected int x;  
 protected int y;  
  
  
 protected int size;  
  
 protected Color color;  
  
 protected double direction\_x;  
 protected double direction\_y;  
  
 protected int speed;  
  
  
 public TFish(GraphicsContext gc, int x, int y, int speed, int size, Color color, double direction\_x, double direction\_y) {  
 this.gc = gc;  
 this.x = x;  
 this.y = y;  
 this.speed = speed;  
 this.size = size;  
 this.color = color;  
 this.direction\_x = direction\_x;  
 this.direction\_y = direction\_y \* (-1);  
 double len = length();  
 this.direction\_x = this.direction\_x / len;  
 this.direction\_y = this.direction\_y / len;  
 }  
  
  
  
 public TFish(GraphicsContext gc) {  
 this.gc = gc;  
 }  
  
 public GraphicsContext getGc() {  
 return gc;  
 }  
  
 public int getX() {  
 return x;  
 }  
  
 public void setX(int x) {  
 this.x = x;  
 }  
  
 public int getY() {  
 return y;  
 }  
  
 public void setY(int y) {  
 this.y = y;  
 }  
  
 public int getSize() {  
 return size;  
 }  
  
 public void setSize(int size) {  
 this.size = size;  
 }  
  
 public Color getColor() {  
 return color;  
 }  
  
 public void setColor(Color color) {  
 this.color = color;  
 }  
  
 public double getDirection\_x() {  
 return direction\_x;  
 }  
  
 public void setDirection\_x(double direction\_x) {  
 this.direction\_x = direction\_x;  
 }  
  
 public double getDirection\_y() {  
 return direction\_y;  
 }  
  
 public void setDirection\_y(double direction\_y) {  
 this.direction\_y = direction\_y;  
 }  
  
 public int getSpeed() {  
 return speed;  
 }  
  
 public void setSpeed(int speed) {  
 this.speed = speed;  
 }  
  
 public void draw() {  
 // считаем вектора углов  
 // координаты линий уголка  
 double cs1 = Math.*cos*(grad(135));  
 double sn1 = Math.*sin*(grad(135));  
 double rx1 = direction\_x \* cs1 - direction\_y \* sn1;  
 double ry1 = direction\_x \* sn1 + direction\_y \* cs1;  
 // коодинаты линий уголка  
 double cs2 = Math.*cos*(grad(-135));  
 double sn2 = Math.*sin*(grad(-135));  
 double rx2 = direction\_x \* cs2 - direction\_y \* sn2;  
 double ry2 = direction\_x \* sn2 + direction\_y \* cs2;  
  
  
 gc.setFill(this.color);  
 double r = size / 2.0;  
 gc.fillOval(x - r / 2.0, y - r / 2.0, r, r);  
  
  
 gc.beginPath();  
 gc.moveTo(x, y);  
 double tmpx1 = x + rx1 \* size;  
 double tmpy1 = y + ry1 \* size;  
 gc.lineTo(tmpx1, tmpy1);  
 gc.closePath();  
 gc.stroke();  
  
  
 gc.beginPath();  
 gc.moveTo(x, y);  
 double tmpx2 = x + rx2 \* size;  
 double tmpy2 = y + ry2 \* size;  
 gc.lineTo(tmpx2, tmpy2);  
 gc.closePath();  
 gc.stroke();  
 }  
  
 protected double grad(double d) {  
 return d \* Math.*PI* / 180.0;  
 }  
  
 private double length()  
 {  
 return (float)Math.*sqrt*((direction\_x \* direction\_x) + (direction\_y \* direction\_y));  
 }  
  
 public void setDirection(double directionX, double directionY) {  
 this.direction\_x = directionX;  
 this.direction\_y = -1 \* directionY;  
  
 double len = length();  
 this.direction\_x = this.direction\_x / len;  
 this.direction\_y = this.direction\_y / len;  
 }  
  
  
  
}

TKarp.java

package com.example.demo7;  
  
import javafx.scene.canvas.GraphicsContext;  
import javafx.scene.paint.Color;  
  
import java.util.List;  
  
public class TKarp extends TFish{  
  
 public TKarp(GraphicsContext gc, int x, int y, int speed, int size, Color color, double direction\_x, double direction\_y) {  
 super(gc, x, y, speed, size, color, direction\_x, direction\_y);  
 }  
  
 @Override  
 public void draw() {  
 // считаем вектора углов  
 // координаты линий уголка  
 double cs1 = Math.*cos*(super.grad(135));  
 double sn1 = Math.*sin*(super.grad(135));  
 double rx1 = super.direction\_x \* cs1 - direction\_y \* sn1;  
 double ry1 = direction\_x \* sn1 + direction\_y \* cs1;  
 // коодинаты линий уголка  
 double cs2 = Math.*cos*(grad(-135));  
 double sn2 = Math.*sin*(grad(-135));  
 double rx2 = direction\_x \* cs2 - direction\_y \* sn2;  
 double ry2 = direction\_x \* sn2 + direction\_y \* cs2;  
  
  
 double cs3 = Math.*cos*(grad(180));  
 double sn3 = Math.*sin*(grad(180));  
 double rx3 = direction\_x \* cs3 - direction\_y \* sn3;  
 double ry3 = direction\_x \* sn3 + direction\_y \* cs3;  
  
  
 gc.setFill(Color.*RED*);  
  
// double r = size / 2.0;  
// gc.fillOval(x - r / 2.0, y - r / 2.0, r, r);  
  
  
  
 double tmpx1 = x + rx1 \* size;  
 double tmpy1 = y + ry1 \* size;  
 double tmpx2 = x + rx2 \* size;  
 double tmpy2 = y + ry2 \* size;  
 double tmpx3 = y + rx3 \* size \* 2;  
 double tmpy3 = y + ry3 \* size \* 2;  
 gc.beginPath();  
 gc.moveTo(x, y);  
 gc.lineTo(tmpx1, tmpy1);  
 gc.lineTo(tmpx2, tmpy2);  
 gc.lineTo(x, y);  
 gc.closePath();  
 gc.fill();  
  
  
  
 }  
}

TPike.java

package com.example.demo7;  
  
import javafx.scene.canvas.GraphicsContext;  
import javafx.scene.paint.Color;  
  
import java.util.List;  
  
public class TPike extends TFish{  
  
 public TPike(GraphicsContext gc, int x, int y, int speed, int size, Color color, double direction\_x, double direction\_y) {  
 super(gc, x, y, speed, size, color, direction\_x, direction\_y);  
 }  
  
 @Override  
 public void draw() {  
 // считаем вектора углов  
 // координаты линий уголка  
 double cs1 = Math.*cos*(super.grad(135));  
 double sn1 = Math.*sin*(super.grad(135));  
 double rx1 = super.direction\_x \* cs1 - direction\_y \* sn1;  
 double ry1 = direction\_x \* sn1 + direction\_y \* cs1;  
 // коодинаты линий уголка  
 double cs2 = Math.*cos*(grad(-135));  
 double sn2 = Math.*sin*(grad(-135));  
 double rx2 = direction\_x \* cs2 - direction\_y \* sn2;  
 double ry2 = direction\_x \* sn2 + direction\_y \* cs2;  
  
  
 double cs3 = Math.*cos*(grad(180));  
 double sn3 = Math.*sin*(grad(180));  
 double rx3 = direction\_x \* cs3 - direction\_y \* sn3;  
 double ry3 = direction\_x \* sn3 + direction\_y \* cs3;  
  
  
 gc.setFill(Color.*GREEN*);  
 gc.setStroke(Color.*GREEN*);  
  
// double r = size / 2.0;  
// gc.fillOval(x - r / 2.0, y - r / 2.0, r, r);  
  
  
  
 double tmpx1 = x + rx1 \* size;  
 double tmpy1 = y + ry1 \* size;  
 double tmpx2 = x + rx2 \* size;  
 double tmpy2 = y + ry2 \* size;  
 double tmpx3 = y + rx3 \* size \* 2;  
 double tmpy3 = y + ry3 \* size \* 2;  
 gc.beginPath();  
 gc.moveTo(x, y);  
 gc.lineTo(tmpx1, tmpy1);  
 gc.lineTo(tmpx2, tmpy2);  
 gc.lineTo(x, y);  
 gc.closePath();  
 gc.fill();  
  
 gc.beginPath();  
 gc.moveTo(x,y);  
 gc.lineTo(tmpx3, tmpy3);  
 gc.closePath();  
 gc.stroke();  
  
  
  
 }  
}

Задача №10

Таблица 1 - Структура программы 1

| Код команды | Мнемокод | Комментарий |
| --- | --- | --- |
| AF | XRA A | Очистить аккумулятор |
| 3E | MVI A | Записать в аккумулятор |
| 08 |  | Число x |
| 06 | MVI B | Записать в регистр B |
| 10 |  | Число a |
| 80 | ADD B | Сложить x и a |
| E7 | RST 7 | Прервать выполнение программы |

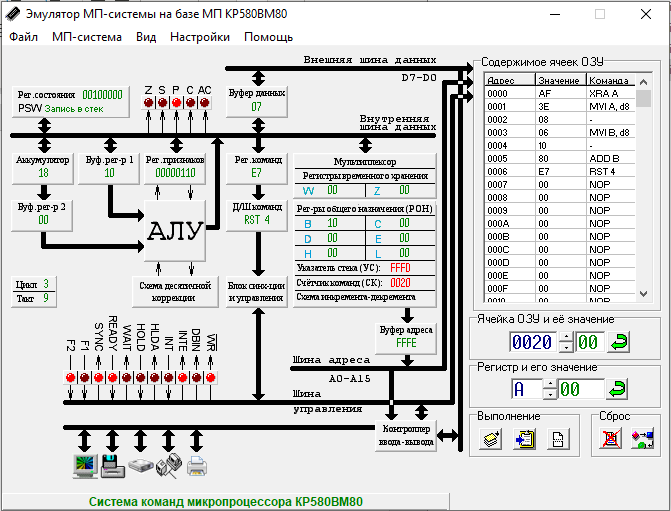


Рисунок 15- Выполнение задания №10

Таблица 2 - Структура программы 2

| Код команды | Мнемокод | Комментарий |
| --- | --- | --- |
| AF | XRA A | Очистить аккумулятор |
| 3E | MVI A | Записать в аккумулятор |
| 08 |  | Число x |
| 06 | MVI B | Записать в регистр B |
| 10 |  | Число a |
| 90 | SUB B | Вычесть из x и a |
| E7 | RST 7 | Прервать выполнение программы |

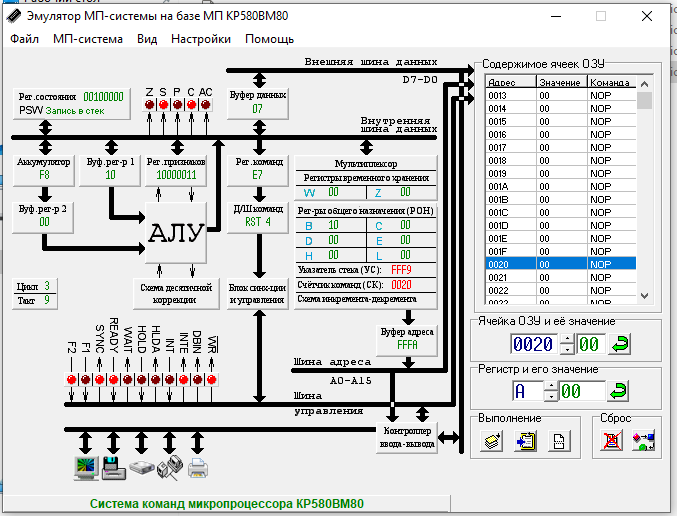


Рисунок 16- Выполнение задания №10

Разработка АИС «Кадрового агентства»

ER-диаграмма

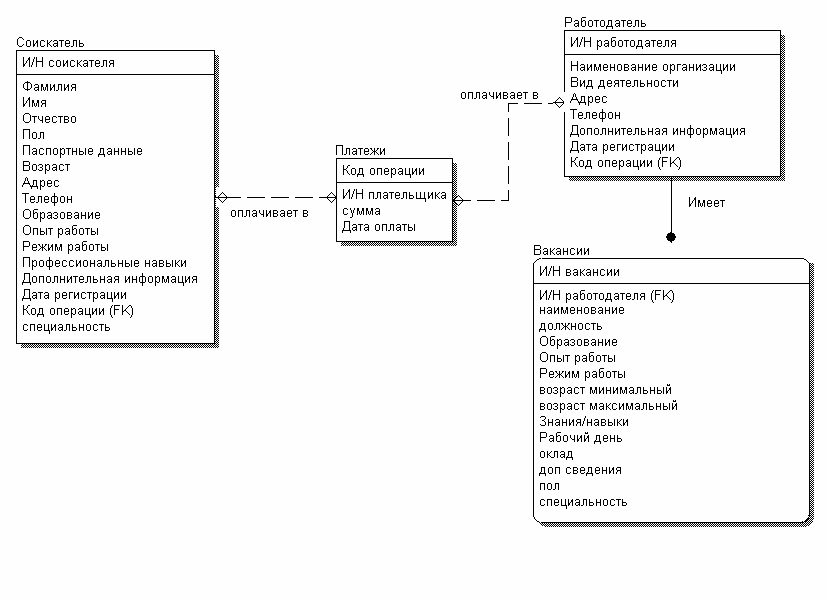


Рисунок 17 – Построенная ER-диаграмма

Разработка сайта

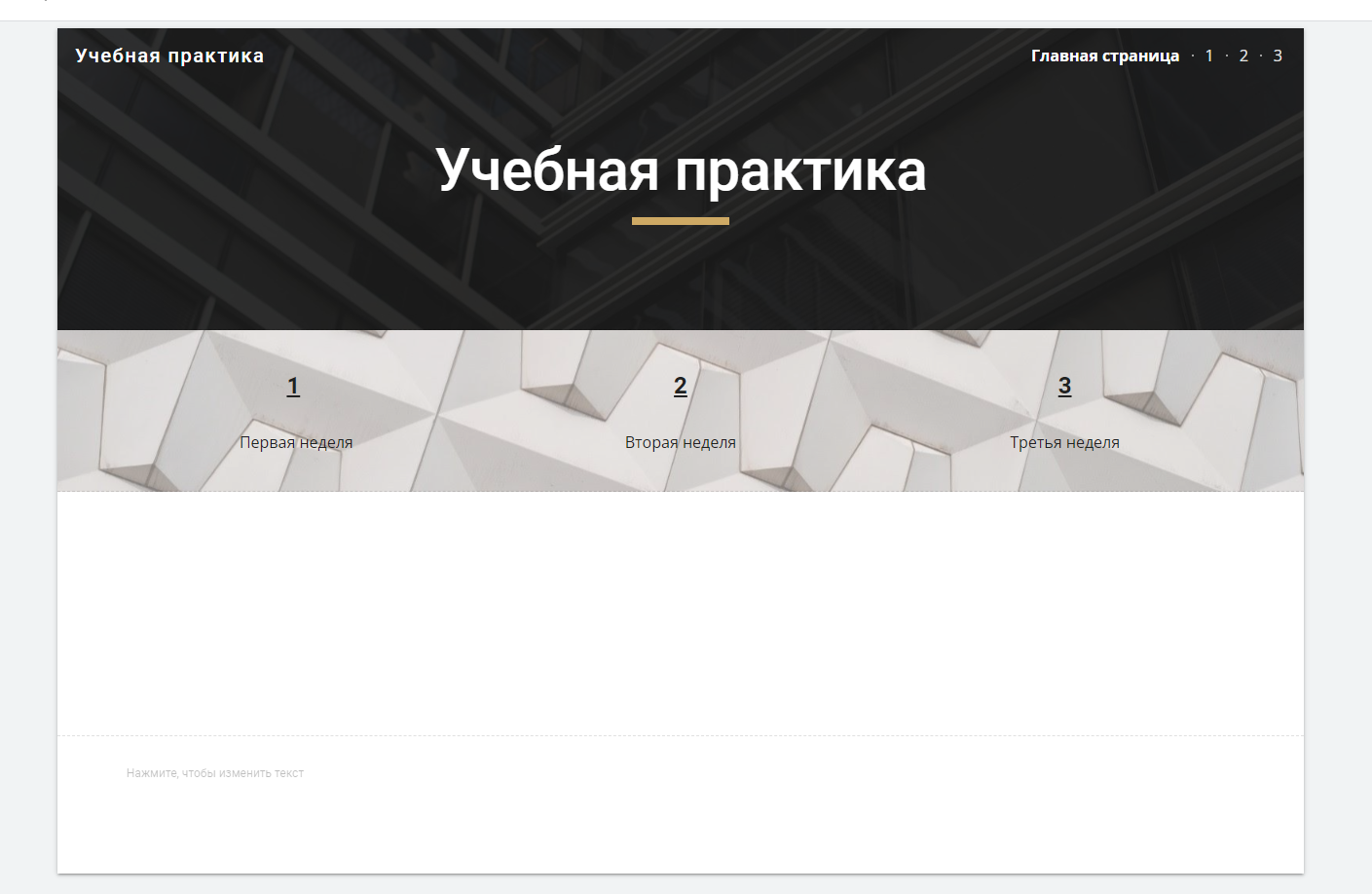


Рисунок 18 – Итоговый дизайн сайта

Сайт был создан через Google Sites  
После присвоение названия сайта и стиля, главная страница была разделена на 3 ссылки на последующие страницы в которых отображается сделанная за выбранную неделю работа

Ссылка на сайт: <https://sites.google.com/view/biwin/главная-страница?authuser=1>

Заключение

В ходе прохождения практики я освоил навыки по разработке автоматических информационных систем, мобильных приложений, закрепил знания в области проектирования программных интерфейсов и баз данных.

Цели и задачи практики достигнуты. Разработаны алгоритмы решения задач с матрицами и циклами, разработана АИС «Библиотеки»,, а также десктопное и мобильное приложение Библиотека. Для демонстрации всех разработанных приложений был сконструирован сайт с использованием хостинга Google Site.

Список использованных источников

Гост TIA/EIA-569 <https://www.system-administrator.pro/files/documents/TIA-EIA-569-A.pdf>

Книга по SQL [(PDF) A standardized corpus for SQLite database forensics (researchgate.net)](https://www.researchgate.net/publication/323927246_A_standardized_corpus_for_SQLite_database_forensics)

Книга по Java [Java. Полное руководство - Герберт Шилдт, 10-е издание.pdf (vk.com)](https://vk.com/doc26879026_492639796?hash=xHS5hjaHQACpqFv5VSP3AnxXfBQYdnGbwmygX1xqW4X&dl=N2n6naPiiQncDlP90h50FBVk6OLAZRL5pqVGG3wcfM4)

Книга по Android Studio [Efficient Android Threading: Asynchronous Processing Techniques for Android ... - Anders Goransson - Google Книги](https://books.google.ru/books?id=PqKfAwAAQBAJ&printsec=frontcover&hl=ru#v=onepage&q&f=false)