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 «09» сентября 2024 г. по «09» ноября 2024 г.

Место практики ГБУ КО ПОО «КИТиС»

|  |  |
| --- | --- |
| Выполнил: | студент 4 курса,  группы ИСп 21-2к  Соколов Ростислав Александрович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Проверила: | Большакова-Стрекалова Анна Викторовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Калининград, 2024

Содержание

[Министерство образования Калининградской области 1](#_Toc180235566)

[Введение 3](#_Toc180235567)

[1 Решение задач 4](#_Toc180235568)

[1.1 Задача 1 4](#_Toc180235569)

[1.2 Задача 2 6](#_Toc180235570)

[1.3 Задача 3 13](#_Toc180235571)

[1.4 Задача 4 18](#_Toc180235572)

[1.5 Задачи 5 21](#_Toc180235573)

[1.6 Задача 6 23](#_Toc180235574)

[1.7 Задача 7 29](#_Toc180235575)

[1.8 Задача 8 32](#_Toc180235576)

[1.9 Задача9 32](#_Toc180235577)

[1.10 Задача 10 37](#_Toc180235578)

[2.1 Задача 1 41](#_Toc180235579)

Введение

Данный отчет по УП ПМ01, включающий в себя дисциплины: разработка программных модулей; разработка мобильных приложений; программирование 1С; поддержка и тестирование программных модулей; системное программирование.

Цели практики:

* Укрепить и углубить свои знания в изучаемых дисциплинах путем выполнения поставленного технического задания.
* Разработать автоматизированную информационную систему, предоставленную преподавателем практики.
* Провести тестирование разработанного программного модуля.
* Разработать мобильное приложение.
* Разработать сайт.
* Разработать и оформить руководство пользователя.

1. Решение задач
   1. Задача 1

Ввести с консоли n-размерность матрицы a [n] [n]. Задать значения элементов матрицы в интервале значений от -n до n с помощью датчика случайных чисел. Перестроить матрицу, переставляя в ней строки так, чтобы сумма элементов в строках полученной матрицы возрастала

import java.util.Random;  
import java.util.Scanner;  
  
public class Matrix {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 System.out.print("Введите размер матрицы: ");  
 int n = scanner.nextInt();  
  
 int[][] matrix = new int[n][n];  
 Random random = new Random();  
  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 matrix[i][j] = random.nextInt(2 \* n + 1) - n;  
 }  
 }  
  
 System.out.println("Обычная матрица:");  
 printMatrix(matrix);  
  
 sortMatrixRows(matrix);  
  
 System.out.println("Сортированная матрица:");  
 printMatrix(matrix);  
 }  
  
 private static void sortMatrixRows(int[][] matrix) {  
 for (int i = 0; i < matrix.length; i++) {  
 for (int j = i + 1; j < matrix.length; j++) {  
 if (getSum(matrix[i]) > getSum(matrix[j])) {  
 int[] temp = matrix[i];  
 matrix[i] = matrix[j];  
 matrix[j] = temp;  
 }  
 }  
 }  
 }  
  
 private static int getSum(int[] row) {  
 int sum = 0;  
 for (int element : row) {  
 sum += element;  
 }  
 return sum;  
 }  
  
 private static void printMatrix(int[][] matrix) {  
 for (int i = 0; i < matrix.length; i++) {  
 for (int j = 0; j < matrix[i].length; j++) {  
 System.out.print(matrix[i][j] + " ");  
 }  
 System.out.println();  
 }  
 }  
}

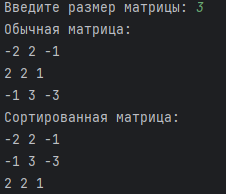


Рисунок 1 - Результат

|  |  |
| --- | --- |
| Название переменной | Тип |
| scan | Scanner |
| random | Random |
| n | int |
| i | int |
| j | int |
| sum | int |

Таблица 1

* 1. Задача 2

Определить класс Четырехугольник на плоскости. Определить площадь и периметр четырехугольника. Создать массив/список/множество объектов и подсчитать количество четырехугольников разного типа (квадрат, прямоугольник, ромб, произвольный). Определить для каждой группы наибольший и наименьший по площади (периметру) объект.

import java.util.\*;

public class Main {

public static void main(String[] args) {

List<Quadrilateral> quadrilaterals = new ArrayList<>();

quadrilaterals.add(new Square(5));

quadrilaterals.add(new Rectangle(5, 10));

quadrilaterals.add(new Rhombus(6, 8));

quadrilaterals.add(new Square(7));

quadrilaterals.add(new Rectangle(3, 6));

quadrilaterals.add(new Rhombus(4, 6));

quadrilaterals.add(new Square(8));

quadrilaterals.add(new Rectangle(4, 8));

quadrilaterals.add(new Rhombus(5, 7));

Map<String, List<Quadrilateral>> groups = groupByType(quadrilaterals);

for (Map.Entry<String, List<Quadrilateral>> entry : groups.entrySet()) {

String type = entry.getKey();

List<Quadrilateral> group = entry.getValue();

Quadrilateral minByArea = Collections.min(group, Comparator.comparingDouble(Quadrilateral::getArea));

Quadrilateral maxByArea = Collections.max(group, Comparator.comparingDouble(Quadrilateral::getArea));

Quadrilateral minByPerimeter = Collections.min(group, Comparator.comparingDouble(Quadrilateral::getPerimeter));

Quadrilateral maxByPerimeter = Collections.max(group, Comparator.comparingDouble(Quadrilateral::getPerimeter));

System.out.println(type + ":");

System.out.println("\tМинимальная площадь: " + minByArea.getArea());

System.out.println("\tМаксимальная площадь: " + maxByArea.getArea());

System.out.println("\tМинимальный периметр: " + minByPerimeter.getPerimeter());

System.out.println("\tМаксимальный периметр: " + maxByPerimeter.getPerimeter());

}

}

public static Map<String, List<Quadrilateral>> groupByType(List<Quadrilateral> quadrilaterals) {

Map<String, List<Quadrilateral>> groups = new HashMap<>();

for (Quadrilateral quadrilateral : quadrilaterals) {

String type = quadrilateral.getType();

if (!groups.containsKey(type)) {

groups.put(type, new ArrayList<>());

}

groups.get(type).add(quadrilateral);

}

return groups;

}

}

abstract class Quadrilateral {

protected double side1;

protected double side2;

protected double side3;

protected double side4;

public Quadrilateral(double side1, double side2, double side3, double side4) {

this.side1 = side1;

this.side2 = side2;

this.side3 = side3;

this.side4 = side4;

}

public double getPerimeter() {

return side1 + side2 + side3 + side4;

}

public abstract double getArea();

public String getType() {

if (side1 == side2 && side2 == side3 && side3 == side4) {

return "Квадрат";

} else if (side1 == side3 && side2 == side4) {

return "Прямоугольник";

} else if (side1 == side3 && side2 == side4 && side1 != side2) {

return "Ромб";

} else {

return "Произвольный четырехугольник";

}

}

}

class Square extends Quadrilateral {

public Square(double side) {

super(side, side, side, side);

}

@Override

public double getArea() {

return side1 \* side1;

}

}

class Rectangle extends Quadrilateral {

public Rectangle(double width, double height) {

super(width, height, width, height);

}

@Override

public double getArea() {

return side1 \* side2;

}

}

class Rhombus extends Quadrilateral {

private double diagonal1;

private double diagonal2;

public Rhombus(double diagonal1, double diagonal2) {

super(Math.sqrt(Math.pow(diagonal1 / 2, 2) + Math.pow(diagonal2 / 2, 2)),

Math.sqrt(Math.pow(diagonal1 / 2, 2) + Math.pow(diagonal2 / 2, 2)),

Math.sqrt(Math.pow(diagonal1 / 2, 2) + Math.pow(diagonal2 / 2, 2)),

Math.sqrt(Math.pow(diagonal1 / 2, 2) + Math.pow(diagonal2 / 2, 2)));

this.diagonal1 = diagonal1;

this.diagonal2 = diagonal2;

}

@Override

public double getArea() {

return diagonal1 \* diagonal2 / 2;

}

}

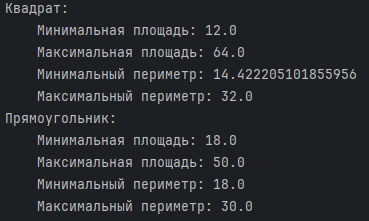


Рисунок 2 - Результат

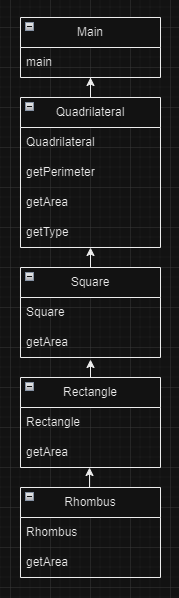


Рисунок 3 - Диаграмма классов

* 1. Задача 3

Создать приложение, удовлетворяющее требованиям, приведенным в задании. Наследование применять только в тех заданиях, в которых это логически обосновано. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString(). Создать объект класса Год, используя классы Месяц, День. Методы: задать дату, вывести на консоль день недели по заданной дате, рассчитать количество дней, месяцев в заданном временном промежутке.

import java.time.DayOfWeek;  
import java.time.LocalDate;  
import java.time.temporal.ChronoUnit;  
import java.util.Objects;  
import java.util.Scanner;  
  
class Month {  
 private int month;  
  
 public Month(int month) {  
 this.month = month;  
 }  
  
 public int getMonth() {  
 return month;  
 }  
}  
  
class Day {  
 private int day;  
  
 public Day(int day) {  
 this.day = day;  
 }  
  
 public int getDay() {  
 return day;  
 }  
}  
  
class Year {  
 private int year;  
  
 public Year(int year) {  
 this.year = year;  
 }  
  
 public int getYear() {  
 return year;  
 }  
}  
  
class Date {  
 private Year year;  
 private Month month;  
 private Day day;  
  
 public Date(Year year, Month month, Day day) {  
 this.year = year;  
 this.month = month;  
 this.day = day;  
 }  
  
 public void setDate(int year, int month, int day) {  
 this.year = new Year(year);  
 this.month = new Month(month);  
 this.day = new Day(day);  
 }  
  
 public void printDayOfWeek() {  
 LocalDate localDate = LocalDate.of(year.getYear(), month.getMonth(), day.getDay());  
 DayOfWeek dayOfWeek = localDate.getDayOfWeek();  
 System.out.println("День недели: " + dayOfWeek);  
 }  
  
 public long calculateDaysBetweenDates(Date startDate, Date endDate) {  
 LocalDate startLocalDate = LocalDate.of(startDate.year.getYear(), startDate.month.getMonth(), startDate.day.getDay());  
 LocalDate endLocalDate = LocalDate.of(endDate.year.getYear(), endDate.month.getMonth(), endDate.day.getDay());  
 return ChronoUnit.DAYS.between(startLocalDate, endLocalDate);  
 }  
  
 public long calculateMonthsBetweenDates(Date startDate, Date endDate) {  
 LocalDate startLocalDate = LocalDate.of(startDate.year.getYear(), startDate.month.getMonth(), startDate.day.getDay());  
 LocalDate endLocalDate = LocalDate.of(endDate.year.getYear(), endDate.month.getMonth(), endDate.day.getDay());  
 return ChronoUnit.MONTHS.between(startLocalDate, endLocalDate);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Date date = (Date) o;  
 return year.equals(date.year) && month.equals(date.month) && day.equals(date.day);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(year, month, day);  
 }  
  
 @Override  
 public String toString() {  
 return "Дата{" +  
 "Год=" + year +  
 ", Месяц=" + month +  
 ", День=" + day +  
 '}';  
 }  
}  
  
public class DateApp {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 System.out.print("Введите год: ");  
 int year = scanner.nextInt();  
 System.out.print("Введите месяц: ");  
 int month = scanner.nextInt();  
 System.out.print("Введите день: ");  
 int day = scanner.nextInt();  
  
 Date date = new Date(new Year(year), new Month(month), new Day(day));  
 date.printDayOfWeek();  
  
 System.out.print("Введите год: ");  
 int startYear = scanner.nextInt();  
 System.out.print("Введите месяц: ");  
 int startMonth = scanner.nextInt();  
 System.out.print("Введите день: ");  
 int startDay = scanner.nextInt();  
  
 System.out.print("Введите год: ");  
 int endYear = scanner.nextInt();  
 System.out.print("Введите месяц: ");  
 int endMonth = scanner.nextInt();  
 System.out.print("Введите день: ");  
 int endDay = scanner.nextInt();  
  
 Date startDate = new Date(new Year(startYear), new Month(startMonth), new Day(startDay));  
 Date endDate = new Date(new Year(endYear), new Month(endMonth), new Day(endDay));  
  
 System.out.println("Дней между датами: " + date.calculateDaysBetweenDates(startDate, endDate));  
 System.out.println("Месяцей между датами: " + date.calculateMonthsBetweenDates(startDate, endDate));  
 }  
}

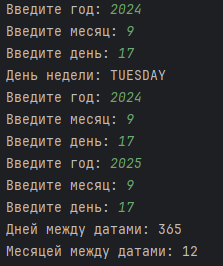


Рисунок 4 - Результат

* 1. Задача 4

Создать класс Notepad с внутренним классом или классами, с помощью объектов которого могут храниться несколько записей на одну дату.

import java.time.LocalDate;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Notepad {  
 private List<Note> notes;  
  
 public Notepad() {  
 this.notes = new ArrayList<>();  
 }  
  
 public void addNote(String text, LocalDate date) {  
 Note note = new Note(text, date);  
 notes.add(note);  
 }  
  
 public void printNotes() {  
 for (Note note : notes) {  
 System.out.println(note);  
 }  
 }  
  
 public void printNotesByDate(LocalDate date) {  
 for (Note note : notes) {  
 if (note.getDate().equals(date)) {  
 System.out.println(note);  
 }  
 }  
 }  
  
 public class Note {  
 private String text;  
 private LocalDate date;  
  
 public Note(String text, LocalDate date) {  
 this.text = text;  
 this.date = date;  
 }  
  
 public String getText() {  
 return text;  
 }  
  
 public LocalDate getDate() {  
 return date;  
 }  
  
 @Override  
 public String toString() {  
 return "Дата: " + date + ", Текст: " + text;  
 }  
 }  
}  
class Main {  
 public static void main(String[] args) {  
 Notepad notepad = new Notepad();  
 notepad.addNote("Сходить в магазин", LocalDate.of(2022, 3, 15));  
 notepad.addNote("Позвонить другу", LocalDate.of(2022, 3, 15));  
 notepad.addNote("Сделать домашнее задание", LocalDate.of(2022, 3, 16));  
  
 System.out.println("Все записи:");  
 notepad.printNotes();  
  
 System.out.println("Записи на 15 марта:");  
 notepad.printNotesByDate(LocalDate.of(2022, 3, 15));  
 }  
}

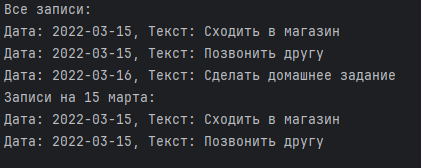


Рисунок 5 - Результат

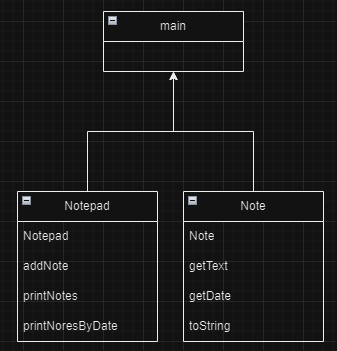


Рисунок 6 - Диаграмма классов

* 1. Задачи 5

Подсчитать количество содержащихся в данном тексте знаков препинания.

public class Main {  
 public static void main(String[] args) {  
 String text = "Подсчитать количество содержащихся в данном тексте знаков препинания.";  
  
 int punctuationCount = 0;  
 for (int i = 0; i < text.length(); i++) {  
 char c = text.charAt(i);  
 if (Character.getType(c) == Character.CONNECTOR\_PUNCTUATION ||  
 Character.getType(c) == Character.CURRENCY\_SYMBOL ||  
 Character.getType(c) == Character.DASH\_PUNCTUATION ||  
 Character.getType(c) == Character.END\_PUNCTUATION ||  
 Character.getType(c) == Character.FINAL\_QUOTE\_PUNCTUATION ||  
 Character.getType(c) == Character.INITIAL\_QUOTE\_PUNCTUATION ||  
 Character.getType(c) == Character.OTHER\_PUNCTUATION ||  
 Character.getType(c) == Character.START\_PUNCTUATION) {  
 punctuationCount++;  
 }  
 }  
  
 System.out.println("Количество знаков препинания в тексте: " + punctuationCount);  
 }  
}



Рисунок 7 - Результат

|  |  |
| --- | --- |
| Название переменной | Тип |
| i | int |
| c | char |
| punctuationCount | int |

Таблица 2

* 1. Задача 6

Выполнить задания на основе задачи № 3, контролируя состояние потоков ввода/вывода. При возникновении ошибок, связанных с корректностью выполнения математических операций, генерировать и обрабатывать исключительные ситуации. Предусмотреть обработку исключений, возникающих при нехватке памяти, отсутствии требуемой записи (объекта) в файле, недопустимом значении поля и т. д.

Выполнить задания из задачи № 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

import java.time.DayOfWeek;  
import java.time.LocalDate;  
import java.time.temporal.ChronoUnit;  
import java.util.InputMismatchException;  
import java.util.Objects;  
import java.util.Scanner;  
  
class Month {  
 private int month;  
  
 public Month(int month) {  
 this.month = month;  
 }  
  
 public int getMonth() {  
 return month;  
 }  
}  
  
class Day {  
 private int day;  
  
 public Day(int day) {  
 this.day = day;  
 }  
  
 public int getDay() {  
 return day;  
 }  
}  
  
class Year {  
 private int year;  
  
 public Year(int year) {  
 this.year = year;  
 }  
  
 public int getYear() {  
 return year;  
 }  
}  
  
class Date {  
 private Year year;  
 private Month month;  
 private Day day;  
  
 public Date(Year year, Month month, Day day) {  
 this.year = year;  
 this.month = month;  
 this.day = day;  
 }  
  
 public void setDate(int year, int month, int day) {  
 this.year = new Year(year);  
 this.month = new Month(month);  
 this.day = new Day(day);  
 }  
  
 public void printDayOfWeek() {  
 LocalDate localDate = LocalDate.*of*(year.getYear(), month.getMonth(), day.getDay());  
 DayOfWeek dayOfWeek = localDate.getDayOfWeek();  
 System.*out*.println("День недели: " + dayOfWeek);  
 }  
  
 public long calculateDaysBetweenDates(Date startDate, Date endDate) {  
 LocalDate startLocalDate = LocalDate.*of*(startDate.year.getYear(), startDate.month.getMonth(), startDate.day.getDay());  
 LocalDate endLocalDate = LocalDate.*of*(endDate.year.getYear(), endDate.month.getMonth(), endDate.day.getDay());  
 return ChronoUnit.*DAYS*.between(startLocalDate, endLocalDate);  
 }  
  
 public long calculateMonthsBetweenDates(Date startDate, Date endDate) {  
 LocalDate startLocalDate = LocalDate.*of*(startDate.year.getYear(), startDate.month.getMonth(), startDate.day.getDay());  
 LocalDate endLocalDate = LocalDate.*of*(endDate.year.getYear(), endDate.month.getMonth(), endDate.day.getDay());  
 return ChronoUnit.*MONTHS*.between(startLocalDate, endLocalDate);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Date date = (Date) o;  
 return year.equals(date.year) && month.equals(date.month) && day.equals(date.day);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(year, month, day);  
 }  
  
 @Override  
 public String toString() {  
 return "Дата{" +  
 "Год=" + year +  
 ", Месяц=" + month +  
 ", День=" + day +  
 '}';  
 }  
}  
  
public class DateApp {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 while (true) {  
 try {  
 System.*out*.print("Введите год: ");  
 int year = scanner.nextInt();  
 System.*out*.print("Введите месяц: ");  
 int month = scanner.nextInt();  
 System.*out*.print("Введите день: ");  
 int day = scanner.nextInt();  
  
 if (month < 1 || month > 12) {  
 throw new InvalidMonthException("Недопустимый месяц");  
 }  
  
 if (day < 1 || day > 31) {  
 throw new InvalidDayException("Недопустимый день");  
 }  
  
 Date date = new Date(new Year(year), new Month(month), new Day(day));  
 date.printDayOfWeek();  
  
 System.*out*.print("Введите год: ");  
 int startYear = scanner.nextInt();  
 System.*out*.print("Введите месяц: ");  
 int startMonth = scanner.nextInt();  
 System.*out*.print("Введите день: ");  
 int startDay = scanner.nextInt();  
  
 System.*out*.print("Введите год: ");  
 int endYear = scanner.nextInt();  
 System.*out*.print("Введите месяц: ");  
 int endMonth = scanner.nextInt();  
 System.*out*.print("Введите день: ");  
 int endDay = scanner.nextInt();  
  
 Date startDate = new Date(new Year(startYear), new Month(startMonth), new Day(startDay));  
 Date endDate = new Date(new Year(endYear), new Month(endMonth), new Day(endDay));  
  
 System.*out*.println("Дней между датами: " + date.calculateDaysBetweenDates(startDate, endDate));  
 System.*out*.println("Месяцей между датами: " + date.calculateMonthsBetweenDates(startDate, endDate));  
  
 break;  
 } catch (InputMismatchException e) {  
 System.*out*.println("Недопустимый ввод. Пожалуйста, введите целое число.");  
 scanner.next();  
 } catch (InvalidMonthException e) {  
 System.*out*.println(e.getMessage());  
 } catch (InvalidDayException e) {  
 System.*out*.println(e.getMessage());  
 }  
 }  
 }  
}  
  
class InvalidMonthException extends Exception {  
 public InvalidMonthException(String message) {  
 super(message);  
 }  
}  
  
class InvalidDayException extends Exception {  
 public InvalidDayException(String message) {  
 super(message);  
 }  
}

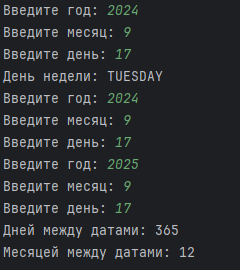


Рисунок 8 - Результат

* 1. Задача 7

Нарисовать кита

import java.awt.\*;

import javax.swing.\*;

public class WhaleDrawing extends JFrame {

public WhaleDrawing() {

super("Кит");

setSize(300, 300);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

add(new WhalePanel());

setVisible(true);

}

public static void main(String[] args) {

new WhaleDrawing();

}

}

class WhalePanel extends JPanel {

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

// Тело кита

g.setColor(Color.BLACK);

g.fillArc(50, 100, 150, 120, 0, 180);

// Глаз

g.setColor(Color.WHITE);

g.fillOval(90, 120, 10, 10);

g.setColor(Color.BLACK);

g.drawOval(90, 120, 5, 5);

// Хвост

g.fillRect(180, 140, 50, 20);

g.setColor(Color.BLACK);

int[] xPoints1 = {270, 268, 218};

int[] yPoints1 = {150, 100, 140};

g.fillPolygon(xPoints1, yPoints1, 3);

// Хвост

g.setColor(Color.BLACK);

int[] xPoints2 = {270, 268, 218};

int[] yPoints2 = {170, 120, 160};

g.fillPolygon(xPoints2, yPoints2, 3);

// Фонтан

g.setColor(Color.LIGHT\_GRAY);

int startX = 120;

int startY = 50;

// Основная струя

for (int i = 0; i < 10; i++) {

g.fillRect(startX - 2, startY + i \* 5, 5, 5);

}

// Верхние струи

for (int i = 0; i < 5; i++) {

int offsetX = i \* 6;

int offsetY = i \* 2;

g.drawLine(startX - offsetX, startY + i \* 5, startX - offsetX, startY + i \* 5 + 2 + offsetY);

g.drawLine(startX + offsetX, startY + i \* 5, startX + offsetX, startY + i \* 5 + 2 + offsetY);

}

}

}

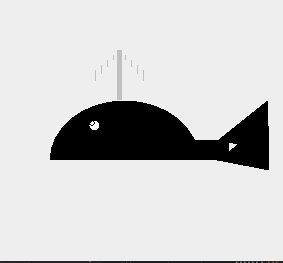


Рисунок 9 - Результат

|  |  |
| --- | --- |
| Название переменной | Тип |
| startX | Int |
| startY | Int |
| xPoints1 | Int |
| xPoints2 | Int |
| yPoints1 | Int |
| yPoints2 | Int |
| offsetX | Int |
| offsetY | int |
| i | int |

Таблица 3

* 1. Задача 8

Определить объект TAquarium, который является местом обитания рыбок и представляет собой область экрана, заполненную водой. Рыбки живут в аквариуме, поэтому экземпляры объекта TFish должны быть полями объекта TAquarium.

* 1. Задача9

Разработать программу, которая позволит имитировать игру человека с компьютером: случайным образом извлекается какая-либо из имеющихся карточек и выдается записанное на ней число. Каждый игрок заносит это число в одну из клеток квадрата, и так продолжается до тех пор, пока не будут заполнены все клетки квадрата.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Random;

public class Matematiko extends JFrame {

private JLabel[][] cells = new JLabel[5][5];

private JButton drawCardButton;

private JLabel drawnCardLabel;

private JLabel PSL;

private JLabel CSL;

private int PS = 0;

private int CS = 0;

private int[][] field = new int[5][5];

private Random random = new Random();

public Matematiko() {

setLayout(new BorderLayout());

JPanel fieldPanel = new JPanel();

fieldPanel.setLayout(new GridLayout(5, 5));

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

cells[i][j] = new JLabel("");

cells[i][j].setBorder(BorderFactory.createLineBorder(Color.BLACK));

fieldPanel.add(cells[i][j]);

}

}

add(fieldPanel, BorderLayout.CENTER);

JPanel controlPanel = new JPanel();

controlPanel.setLayout(new FlowLayout());

drawCardButton = new JButton("Вытянуть карту");

drawCardButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

drawCard();

}

});

controlPanel.add(drawCardButton);

drawnCardLabel = new JLabel("Вытянутая карта: ");

controlPanel.add(drawnCardLabel);

PSL = new JLabel("Счет игрока: 0");

controlPanel.add(PSL);

CSL = new JLabel("Счет компьютера: 0");

controlPanel.add(CSL);

add(controlPanel, BorderLayout.SOUTH);

drawCard();

}

private void drawCard() {

int card = random.nextInt(13) + 1;

drawnCardLabel.setText("Вытянутая карта: " + card);

int playerRow = random.nextInt(5);

int playerCol = random.nextInt(5);

while (field[playerRow][playerCol] != 0) {

playerRow = random.nextInt(5);

playerCol = random.nextInt(5);

}

field[playerRow][playerCol] = card;

cells[playerRow][playerCol].setText(String.valueOf(card));

PS += card;

PSL.setText("Счет игрока: " + PS);

int computerRow = random.nextInt(5);

int computerCol = random.nextInt(5);

while (field[computerRow][computerCol] != 0) {

computerRow = random.nextInt(5);

computerCol = random.nextInt(5);

}

field[computerRow][computerCol] = card;

cells[computerRow][computerCol].setText(String.valueOf(card));

CS += card;

CSL.setText("Счет компьютера: " + CS);

boolean gameOver = true;

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

if (field[i][j] == 0) {

gameOver = false;

break;

}

}

}

if (gameOver) {

drawCardButton.setEnabled(false);

if (PS > CS) {

JOptionPane.showMessageDialog(this, "Игрок выиграл!");

} else if (PS < CS) {

JOptionPane.showMessageDialog(this, "Компьютер выиграл!");

} else {

JOptionPane.showMessageDialog(this, "Ничья!");

}

}

}

public static void main(String[] args) {

Matematiko game = new Matematiko();

game.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

game.setSize(400, 400);

game.setVisible(true);

}

}

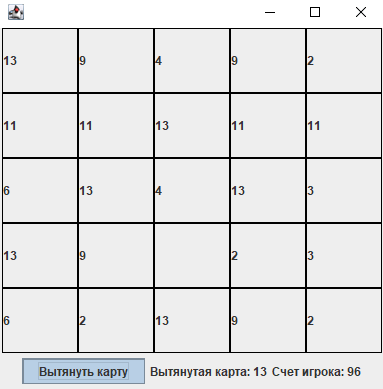


Рисунок 10 - Результат

|  |  |
| --- | --- |
| Название переменной | Тип |
| field | Int[][] |
| PS | Int |
| CS | Int |
| playerRow | Int |
| playerCol | Int |
| computerRow | Int |
| computerCol | Int |
| card | Int |
| i | Int |
| j | Int |

Таблица 4

* 1. Задача 10

АССЕМБЛЕР (рекомендация – использовать «Ассемблер учебник»)

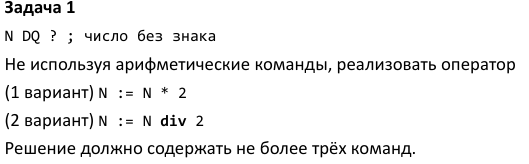


Рисунок 11 - Задача

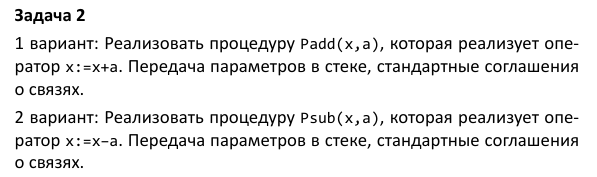


Рисунок 12 - Задача

section .data

N dd 5

result db 'Result: ', 0

newline db 10, 0

section .bss

buf resb 10

section .text

global \_start

\_start:

mov eax, [N]

shl eax, 1

mov [N], eax

mov edi, buf

call int\_to\_string

mov eax, 1

mov edi, 1

mov rsi, result

mov edx, 8

syscall

mov eax, 1

mov edi, 1

mov rsi, buf

mov edx, 10

syscall

mov eax, 1

mov edi, 1

mov rsi, newline

mov edx, 1

syscall

mov eax, 60

xor edi, edi

syscall

int\_to\_string:

mov ecx, 10

xor ebx, ebx

convert\_loop:

xor edx, edx

div ecx

add dl, '0'

mov [rdi], dl

inc rdi

test eax, eax

jnz convert\_loop

ret

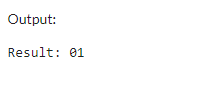


Рисунок 13 - Результат

section .data

msg db "Result: ", 0

result db 0, 0, 0, 0, 0

section .text

global \_start

\_start:

mov rdi, 10

mov rsi, 5

call Padd

mov rbx, result

call int\_to\_str

mov rax, 1

mov rdi, 1

mov rsi, msg

mov rdx, 8

syscall

mov rax, 1

mov rdi, 1

mov rsi, result

mov rdx, 5

syscall

mov rax, 60

xor rdi, rdi

syscall

Padd:

add rdi, rsi

mov rax, rdi

ret

int\_to\_str:

mov rcx, 0

mov rbx, result

convert\_loop:

xor rdx, rdx

div rsi

add dl, '0'

mov [rbx + rcx], dl

inc rcx

test rax, rax

jnz convert\_loop

mov rdi, result

mov rsi, rbx

dec rsi

reverse\_loop:

cmp rdi, rsi

jge reverse\_done

mov al, [rdi]

mov bl, [rsi]

mov [rdi], bl

mov [rsi], al

inc rdi

dec rsi

jmp reverse\_loop

reverse\_done:

ret

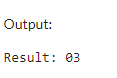


Рисунок 14 - Результат

2.1 Задача 1

1. Анализ предметной области. Разработка ER-диаграммы «сущность-связь» (ErWin). Обязательна 3 нормальная форма с обеспечением ссылочной целостности. При разработке диаграммы обратите внимание на согласованную осмысленную схему именования, создайте необходимые первичные и внешние ключи, определите ограничения внешних ключей, отражающие характер предметной области.
2. ER - диаграмма должна быть представлена в формате удобном для просмотра и содержать таблицы, связи между ними, атрибуты и ключи (типами данных на данном этапе можно пренебречь) проведение анализа поставленной задачи и проектирования базы данных (ERD модели) с применением case-средств;
3. Создайте все необходимые сущности, определите отношения, создайте ограничения на связи между сущностями (при наличии всех связей), приведите базу данных к 3НФ (при наличии всех сущностей и связей).

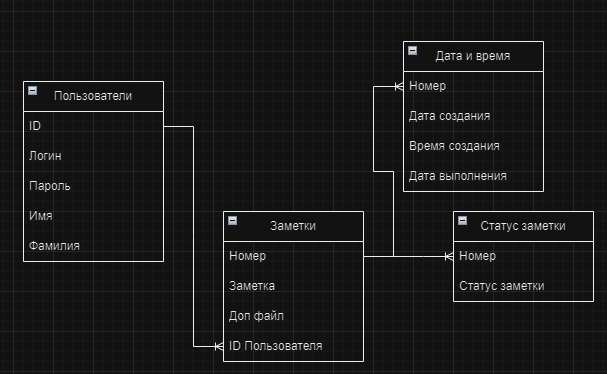


Рисунок 15 - Диаграмма

1. Разработка диаграммы вариантов использования, диаграммы классов, диаграммы последовательностей, диаграммы кооперации, диаграммы состояний, диаграммы деятельности (MS Visio, Rational Rose и другие) – в процессе
2. На основании анализа предметной области сформировать «Техническое задание», оформить в соответствии с требованиями стандартов ЕСПД
3. Введение
   1. Наименование системы: Система «Органайзер».
   2. Основание для разработки: Анализ предметной области «Органайзер» (Заметки).
4. Цель разработки

Целью разработки АИС является создание программы для управления заметками, их статусом и датой выполнения.

1. Область применения

Система будет использоваться для управления данными заметок пользователей.

1. Требования к системе
   1. Функциональные требования:

* Управление данными пользователей
* Управление статусом заметки (выполнено/не выполнено)
  1. Нефункциональные требования:
* Интерфейс должен быть интуитивно понятным и удобным пользователю
* Высокая скорость обработки и обновление статуса заметок

1. Условия эксплуатации

Система должна работать на Windows и быть доступна через приложение, написанное на Java, с использованием JavaFX для графического интерфейса и PostgreSQL для управления базой данных.

1. Требования к программному обеспечению
   1. Язык программирования: Java
   2. Среда разработки: IntelliJ IDEA
   3. Среда выполнения: Windows 7/10/11
   4. База данных: MySQL
2. Требования к документированию

К системе должны быть представлены следующие документы:

* Руководство пользователя
* Техническое руководство по установке системы

1. Этапы разработки
   1. Анализ требований и подготовка технического задания
   2. Проектирование системы и базы данных
   3. Разработка интерфейса и функциональности системы
   4. Тестирование системы и исправление ошибок
   5. Внедрение и обучение персонала
2. Порядок контроля и приемки
   1. Контроль качества: регулярная проверка и тестирование системы перед внедрением.
   2. Приемка: система будет считаться принятой, если она соответствует всем требованиям ТЗ и успешно проходит все этапы тестирования.

package org.example;

import java.awt.\*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import javax.swing.\*;

import java.awt.\*;

import java.sql.\*;

import java.util.Vector;

public class NotesDatabaseGUI extends JFrame {

private JTable usersTable;

private JTable notesTable;

private JTable datesTable;

private JTable statusTable;

private DefaultTableModel usersTableModel;

private DefaultTableModel notesTableModel;

private DefaultTableModel datesTableModel;

private DefaultTableModel statusTableModel;

private Connection connection;

public NotesDatabaseGUI() {

super("База данных заметок");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(800, 600);

setLocationRelativeTo(null);

// Создаем таблицы

usersTable = new JTable();

notesTable = new JTable();

datesTable = new JTable();

statusTable = new JTable();

// Создаем модели таблиц

usersTableModel = new DefaultTableModel();

notesTableModel = new DefaultTableModel();

datesTableModel = new DefaultTableModel();

statusTableModel = new DefaultTableModel();

// Устанавливаем модели для таблиц

usersTable.setModel(usersTableModel);

notesTable.setModel(notesTableModel);

datesTable.setModel(datesTableModel);

statusTable.setModel(statusTableModel);

// Создаем вкладки

JTabbedPane tabbedPane = new JTabbedPane();

tabbedPane.addTab("Пользователи", new JScrollPane(usersTable));

tabbedPane.addTab("Заметки", new JScrollPane(notesTable));

tabbedPane.addTab("Дата и время", new JScrollPane(datesTable));

tabbedPane.addTab("Статус заметки", new JScrollPane(statusTable));

// Создаем кнопку "Обновить"

JButton updateButton = new JButton("Обновить");

updateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

updateTables();

}

});

// Добавляем кнопку в панель инструментов

JToolBar toolBar = new JToolBar();

toolBar.add(updateButton);

// Добавляем вкладки и кнопку в фрейм

getContentPane().add(toolBar, BorderLayout.NORTH);

getContentPane().add(tabbedPane, BorderLayout.CENTER);

// Устанавливаем соединение с базой данных

connectToDatabase();

// Заполняем таблицы

updateTables();

setVisible(true);

}

private void connectToDatabase() {

try {

connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/notes", "root", "122333444455555");

System.out.println("Соединение с базой данных установлено.");

} catch (SQLException e) {

JOptionPane.showMessageDialog(this, "Ошибка при подключении к базе данных: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

private void updateTables() {

try {

// Обновляем данные в таблице "Пользователи"

updateUsersTable();

// Обновляем данные в таблице "Заметки"

updateNotesTable();

// Обновляем данные в таблице "Дата и время"

updateDatesTable();

// Обновляем данные в таблице "Статус заметки"

updateStatusTable();

} catch (SQLException e) {

JOptionPane.showMessageDialog(this, "Ошибка при обновлении данных: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

// Методы для обновления таблиц

private void updateUsersTable() throws SQLException {

Statement stmt = connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, ResultSet.CONCUR\_UPDATABLE);

ResultSet rs = stmt.executeQuery("SELECT \* FROM users");

usersTableModel.setDataVector(getResultSetData(rs), new String[]{"ID", "Логин", "Пароль", "Имя", "Фамилия"});

rs.close();

stmt.close();

}

private void updateNotesTable() throws SQLException {

Statement stmt = connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, ResultSet.CONCUR\_UPDATABLE);

ResultSet rs = stmt.executeQuery("SELECT \* FROM notes");

notesTableModel.setDataVector(getResultSetData(rs), new String[]{"Номер", "Заметка", "Доп. файл", "ID пользователя"});

rs.close();

stmt.close();

}

private void updateS tatusTable() throws SQLException {

Statement stmt = connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, ResultSet.CONCUR\_UPDATABLE);

ResultSet rs = stmt.executeQuery("SELECT \* FROM status");

statusTableModel.setDataVector(getResultSetData(rs), new String[]{"Номер", "Статус заметки"});

rs.close();

stmt.close();

}

private void updateDatesTable() throws SQLException {

Statement stmt = connection.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, ResultSet.CONCUR\_UPDATABLE);

ResultSet rs = stmt.executeQuery("SELECT \* FROM dates");

datesTableModel.setDataVector(getResultSetData(rs), new String[]{"Номер", "Дата создания", "Время создания", "Дата выполнения"});

rs.close();

stmt.close();

}

private Object[][] getResultSetData(ResultSet rs) throws SQLException {

// Получаем количество столбцов

ResultSetMetaData metaData = rs.getMetaData();

int columnCount = metaData.getColumnCount();

// Получаем количество строк

rs.last();

int rowCount = rs.getRow();

rs.beforeFirst();

// Создаем двумерный массив для хранения данных

Object[][] data = new Object[rowCount][columnCount];

// Заполняем массив данными

int row = 0;

while (rs.next()) {

for (int col = 0; col < columnCount; col++) {

data[row][col] = rs.getObject(col + 1);

}

row++;

}

return data;

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

new NotesDatabaseGUI();

}

});

}

}



Рисунок 16 - таблицы

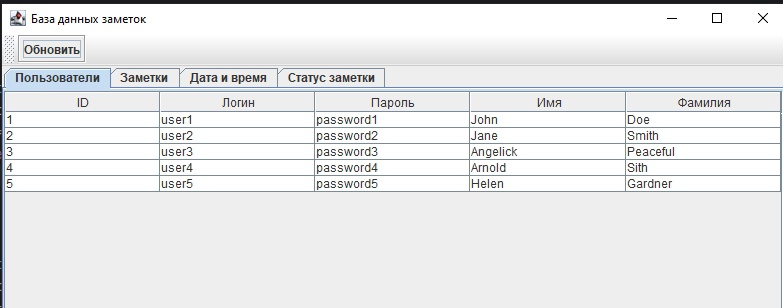


Рисунок 17 – Таблица пользователи

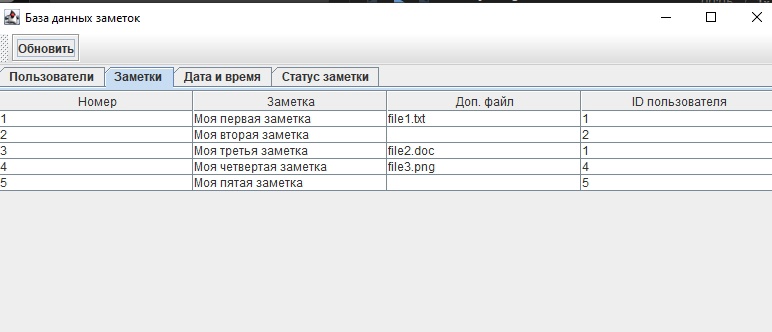


Рисунок 18 – Таблица заметки

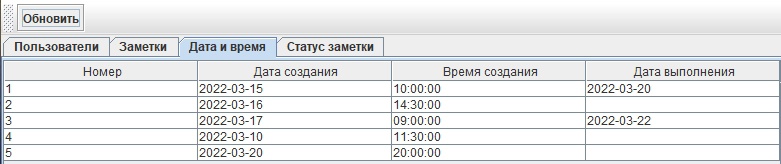


Рисунок 19 – Таблица дата и время

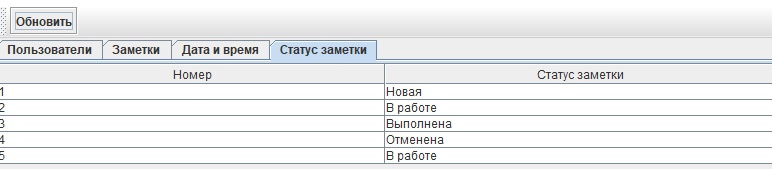


Рисунок 20 – Таблица статус заметки

Разработано мобильное приложение, взаимодействующее с базой данных MySQL и предоставляющее доступ к информации о заметках пользователей. Интерфейс организован в несколько разделов, каждый из которых отображает извлеченные из базы данные:

1. Пользователи – отображает список пользователей с их ID, фамилией, именем, логином и паролем.
2. Заметки – список номера заметки, самой заметки, доп файла и ID пользователя.
3. Дата и время – список с номером заметки, датой создания заметки, временем создания и датой выполнения (если заметка выполнена).
4. Статус заметки – список с номером заметки и статусом (Новая, В работе, Выполнена или Отменена).

Для реализации функционала использована база данных MySQL, из которой с помощью SQL-запросов извлекаются необходимые данные. Эти данные отображаются в структурированном виде, что делает приложение удобным инструментом для работы с заметками пользователей, их статусом и дополнительными файлами.



Рисунок 21 – Мобильное приложение «Главное меню»

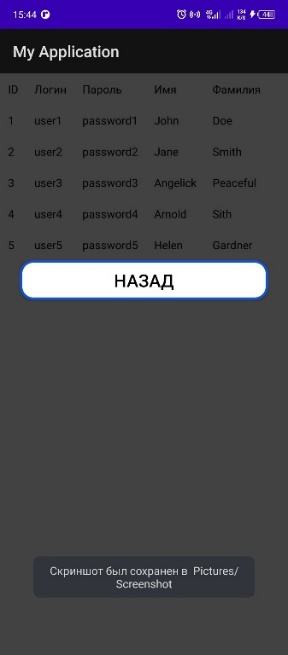


Рисунок 21 – Мобильное приложение «Пользователи»

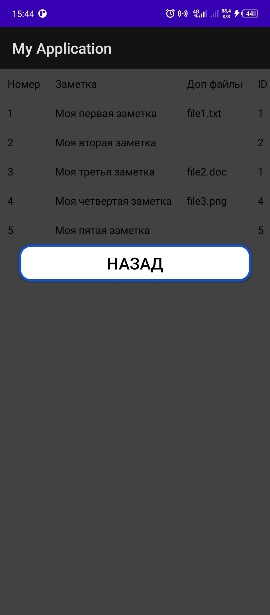


Рисунок 22 – Мобильное приложение «Заметки»

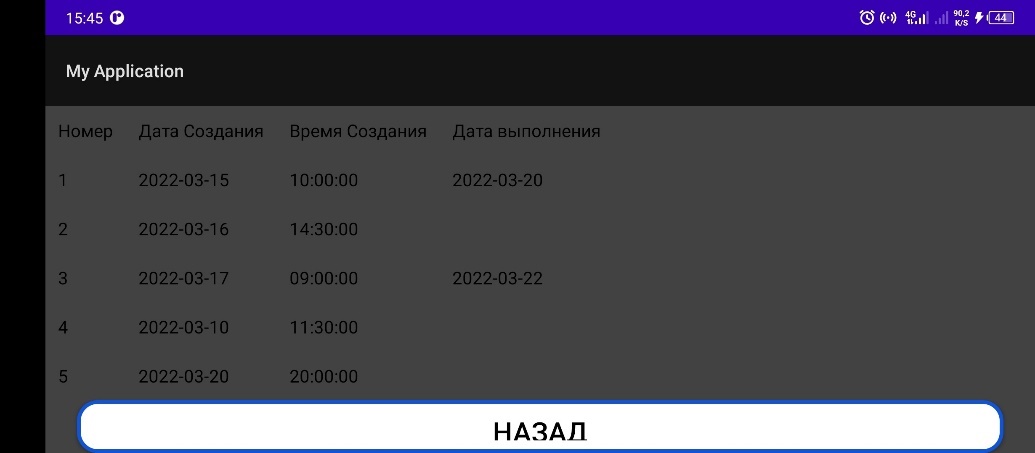


Рисунок 23 – Мобильное приложение «Дата и время»

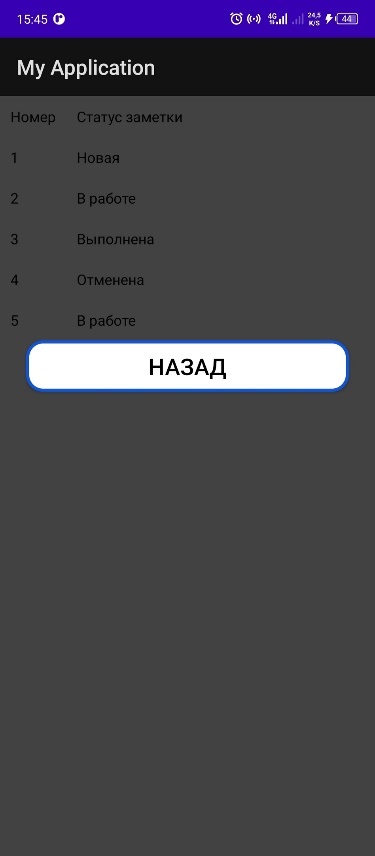


Рисунок 24 – Мобильное приложение «Статус заметки»

# 5. Разработка сайта

Сайт разработан для предоставления информации и услуг пользователям с акцентом на удобство и функциональность. В процессе создания сайта использовались современные веб-технологии и подходы к разработке, чтобы обеспечить его быстродействие, удобный интерфейс и адаптивность для различных устройств.

Основные этапы разработки:

1. Планирование и дизайн — проведен анализ пользовательских потребностей, разработана структура сайта и спроектирован интерфейс. Было важно создать интуитивно понятную навигацию, обеспечивающую легкий доступ к основным разделам.
2. Верстка и адаптация — сайт сверстан с учетом кроссбраузерности и адаптивности, чтобы корректно отображаться на различных устройствах, включая смартфоны и планшеты. Использованы HTML, CSS и JavaScript для создания отзывчивого и приятного интерфейса.
3. Интеграция с серверной частью — для работы с данными реализованы серверные компоненты, обеспечивающие взаимодействие сайта с базой данных или API, что позволяет динамически подгружать и отображать актуальную информацию.
4. Оптимизация и тестирование — выполнена оптимизация кода и графики для повышения скорости загрузки страниц, а также тестирование сайта для выявления и исправления возможных ошибок или проблем с производительностью.

Сайт представляет собой удобный и функциональный ресурс, соответствующий современным стандартам качества и безопасности, обеспечивая пользователям простой доступ к нужной информации и услугам.

Сайт - https://demo-2675870043.smart-engine.ru/

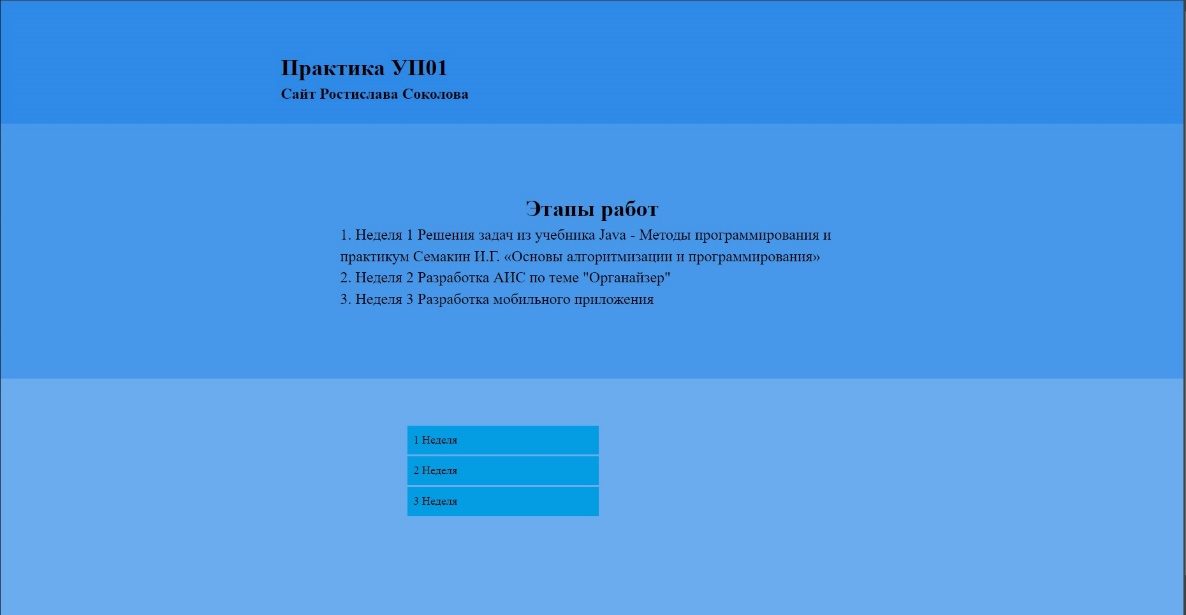
****

Рисунок 25 – Скриншот главного экрана сайта

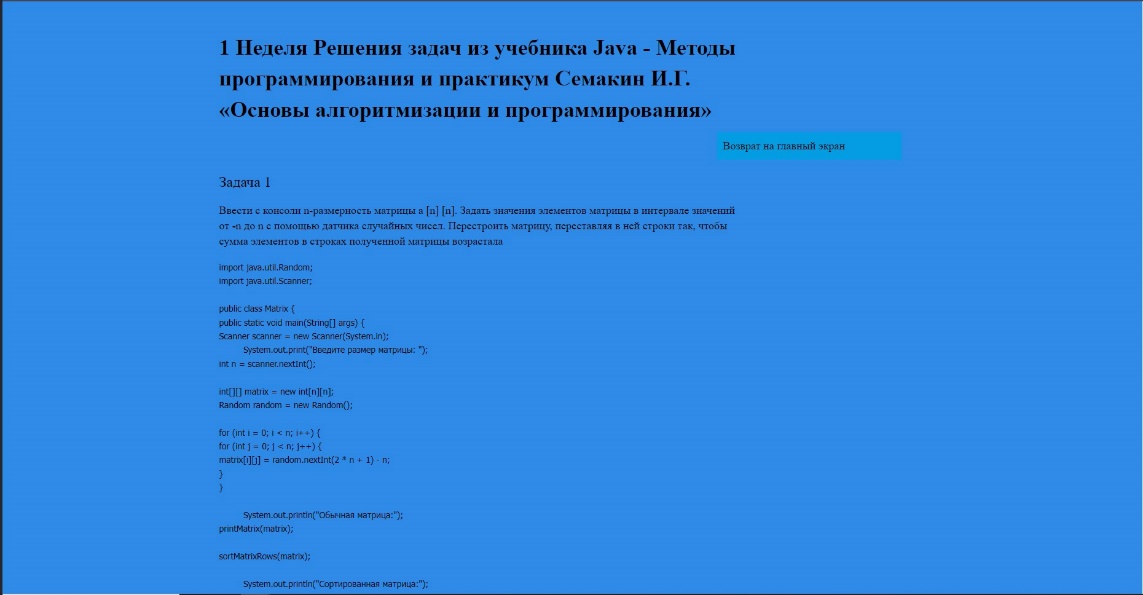


Рисунок 26 – Скриншот заданий 1 недели на сайте

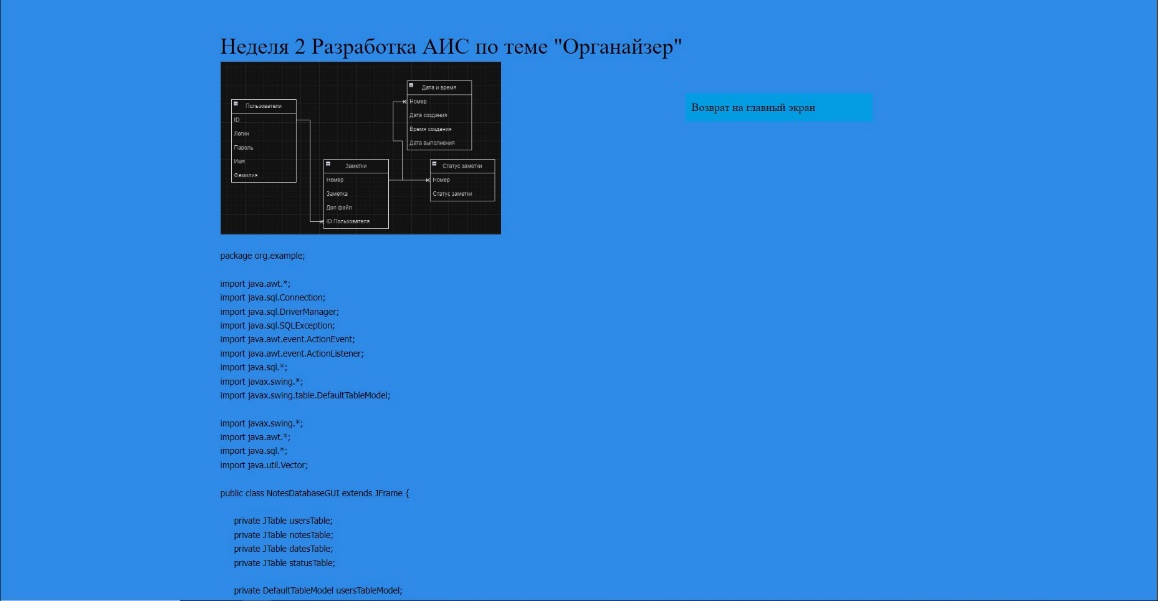
****

Рисунок 27 – Скриншот заданий 2 недели на сайте

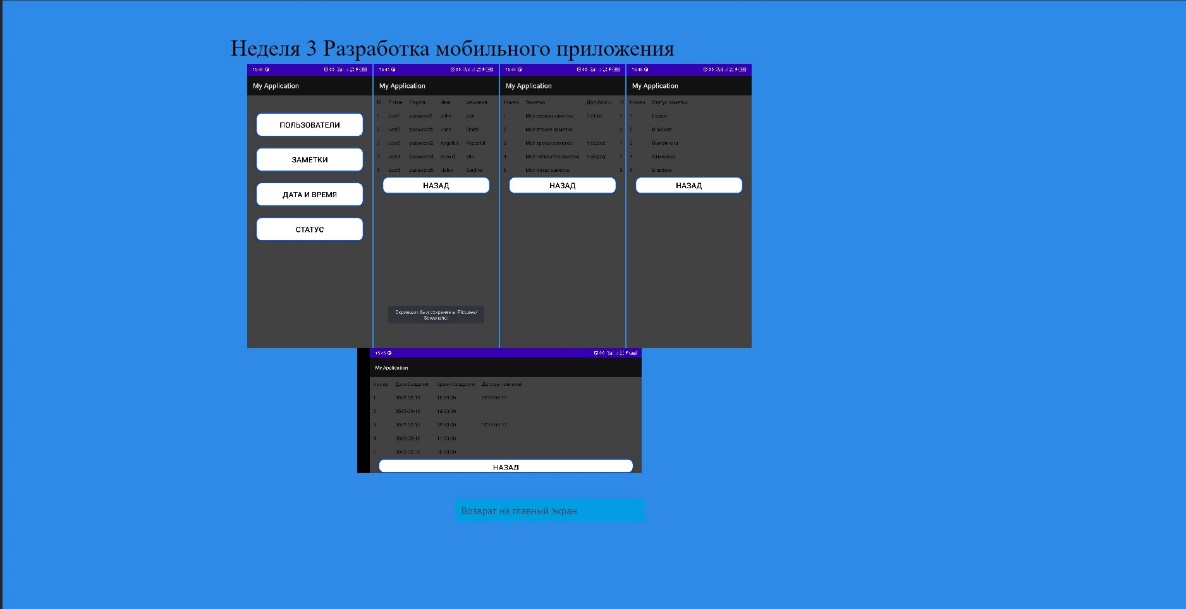
****

Рисунок 28 – Скриншот заданий 3 недели на сайте

# Заключения

Полное выполнение программы — Практикант успешно завершил все поставленные задачи, охватывающие темы от системного программирования до разработки мобильных приложений и сайтов. Каждое задание имеет ясное описание, кодовые реализации, схемы и результаты выполнения программ.

Разработка Автоматизированной информационной системы (АИС) — Проект по созданию системы управления телефонными номерами и тарифами был выполнен в полном объеме. В отчёте подробно описан процесс анализа предметной области, разработка ER-диаграммы и структуры базы данных с соблюдением 3НФ, а также разработка документации.

Качественная реализация кода и использование OOP — В отчёте представлены классы и методы, отражающие основное понимание ООП. Сложные программы, такие как "Математико" и система управления счётом с историей операций, выполнены с обработкой исключений и соответствующими проверками, что улучшает надёжность кода.

Проектирование и тестирование — В процессе создания программы уделено внимание тестированию, проверке ошибок и обработке исключений, что указывает на тщательный подход к разработке качественного программного обеспечения.

Профессиональные навыки — Отчёт демонстрирует владение практическими навыками, которые будут полезны в дальнейшей профессиональной деятельности.

Список используемой литературы:

1. Вагин, Д. В. Современные технологии разработки веб-cайтов / Д. В. Вагин, Р. В. Петров. – Новосибирск: НГТУ, 2023
2. Белугина, С. В. Разработка программных модулей программного обеспечения для компьютерных систем. Прикладное программирование / С. В. Белугина. – 4-е изд., стер. – Санкт-Петербург: Лань, 2022
3. Java 9 для начинающих / Патрик Нимейер
4. Разработка сайта: эффективные Java-приложения: разработка, тестирование и производительность / Кей С. Хорстманн
5. Блох, Д. Java. Эффективное программирование/Д. Блох. – 3-е

изд., – Диалектика , 2022

1. Аблязов, Р. Программирование на ассемблере на платформе х86-64 / Р. Аблязов. – Москва: ДМК Пресс, 2020
2. Рысин, М. Л. Введение в современную Android-разработку на языке Java: учебное пособие / М. Л. Рысин. – Москва: РТУ МИРЭА, 2023 – Часть 1 – 2023.
3. Курбатова, И. В. Основы программирования на языке Java: учебное пособие для вузов / И. В. Курбатова, А. В. Печкуров. – Санкт-Петербург: Лань, 2024
4. Курбатова, И. В. Основы программирования на языке Java: учебное пособие для спо / И. В. Курбатова, А. В. Печкуров. – Санкт-Петербург: Лань, 2024
5. Фомичева, С. Г. Основы программирования на языке низкого уровня Assembler: учебное пособие / С. Г. Фомичева, О. С. Варига, А. А. Попкова. – Норильск: ЗГУ им. Н.М. Федоровского, 2022