МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Государственное бюджетное общеобразовательное учреждение

«Президентский физико-математический лицей №239»

«Приложение для тестирования знаний учащихся по геометрии»

Годовой проект по информатике

Кошкелян Тигран 10-2 класс

2018г.

Постановка задачи:

В целях упрощения системы тестирования знаний учащихся по различным предметам, а также подготовке их к экзаменам в формате ОГЭ и ЕГЕ было предложено создать приложения, помогающие учащимся запоминать важнейшие сведения из школьных предметов, а также позволяющее им пройти пробные тестирования для подготовки к экзаменам.

В рамках годового проекта нам необходимо создать подобное обучающее приложение по предмету «Геометрия», в которое будут включены важнейшие формулы как из курса планиметрии (для 7-9 классов), так и из курса стереометрии (для 10-11 классов).

Этот отчёт посвящен созданию сервера, на котором будут храниться верные ответы на предложенные в приложении задания, а также тексты самих заданий, и прототипа самого приложения.

Уточнение входных и выходных данных:

Учащемуся, использующему наше приложение, будут предложены вопросы и задания, ответы на которые будет необходимо вносить в специально отведённые для этого поля. После того, как учащийся введёт ответ и нажмет кнопку «Ответить», система будет сравнивать введённый ответ с правильным, записанным на сервере (в случае нескольких вариантов будет проверено совпадение хотя бы с одним из них), и выдавать (выводить на экран) результат: «Правильно», если введённый ответ совпадает с ответом на сервере, и «Неправильно» в противном случае.

Таким образом, входным данным является текст, введённый учащимся, а выходным – надпись «Правильно» или «Неправильно».

Анализ используемой структуры данных:

Согласно предыдущему пункту, тип входных данных – строковый (String). Эта строка будет сравниваться с возможными ответами, приведёнными в таблице, записанной в базе данных, созданной заблаговременно.

Основной структурой данных, которая будет нами храниться, является, безусловно, сама база данных, в которую занесены тексты заданий, а также ответы на них. Хранение ее неизбежно ввиду первостепенной важности данных в ней при работе приложения.

Выбор метода решения:

Для создания приложения было принято решение о создании сервера с базой данных, так как это представляется наиболее удобным способом для проверки знаний учащихся, то есть для сравнения введённых ими ответов с верными. Преимущества этого способа заключаются в том, что база данных – объективно наиболее удобный способ хранения больших массивов данных, а сервер позволяет нескольким клиентам одновременно проходить тестирование.

Используемые технологии:

В рамках выполнения данного проекта были использованы следующие технологии:

1. Язык программирования Java

2. Среда программирования Intellij Idea

3. Система управления базами данных PostgreSQL

Выбор языка программирования связан с тем, что он является объектно-ориентированным, что позволяет создать класс, к которому будет обращаться приложение для получения доступа к базе данных, а также к корректировке последней (она будет при этом защищена логином и паролем, которые должен знать пользователь, желающий отредактировать её).

Помимо этого, изучение этого языка является основной частью курса информатики в 10 классе, что позволяет использовать его изучение в рамках школьной программы для реализации проекта. С этим же связан выбор среды программирования. Кроме того, Intellij Idea является весьма удобной для изучения новых технологий, так как она сама предлагает различные конструкции при введении первых символов.

Система управления базами данных (СУБД) выбрана в связи с тем, что она является одновременно интересной для изучения (не настолько простой, как, например, SQLLite) и не слишком сложной для усвоения в течение полугода (в отличие от более усовершенствованных вариантов). Кроме того, информацию об этой системе легко найти в Интернете, что ускоряет и облегчает изучение.

Проект выполняется на компьютере с оперативной системой Windows 7. Для подключения и получения доступа к информации из базы данных используется специальный драйвер Java DataBase Connectivity (JDBC) – платформенно-независимый промышленный стандарт взаимодействия Java-приложений с различными СУБД, так как он является наиболее часто используемым драйвером для работы с СУБД через Java.

Комментированный листинг:

1. Создание базы данных:

```
DROP TABLE IF EXISTS required;

DROP TABLE IF EXISTS formula;
//чтоб не создать базу данных с таким же названием, как уже существующая
```

```
CREATE TABLE required
//таблица, содержащая измеряемые величины
(
num integer PRIMARY KEY,
//номер измеряемой величины, не может быть равен 0, не может повторяться
name text
//измеряемая величина
);
CREATE TABLE formula (
//таблица, содержащая необходимые формулы
num integer REFERENCES required(num),
//номер формулы
value text
//собственно формула
);

INSERT INTO required (num, name) VALUES (1, 'S=');

<другие формулы>

SELECT * FROM required INNER JOIN formula USING (num);

//вывод всех формул для каждой из величин
```

2. Подпрограмма, ищущая формулу по номеру

```
public static String getFormula(int a, int b) {
    String s = "";
    try {
    Class.forName("org.postgresql.Driver"); //подключение к JDBC
    } catch (ClassNotFoundException e) {
        System.out.println("No JBDC");
        e.printStackTrace();
    }
    Connection connection = null;
    try {
        connection = DriverManager.getConnection
                ("jdbc:postgresql://127.0.0.1:5432/Formulas 2",
                   "postgres",    "123456"); //подключение к БД
        PreparedStatement preparedStatement = null;
        preparedStatement = connection.prepareStatement
                ("SELECT required.name, formula.value FROM
required, "
                        + "formula WHERE (required.num =
formula.num) AND " +
                        "(required.num = ?) AND (formula.numb =
```

```
?);"); //выбираем нужную формулу из базы данных
    preparedStatement.setInt(1, a);
    preparedStatement.setInt(2, b); //задаём параметры выбора
    ResultSet result1 = preparedStatement.executeQuery();
    while (result1.next()) {
      s += result1.getString("name")+result1.getString("value");
      //создаём строку в качестве ответа
     }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return s;
}
```

3. Сервер на Java:

```
int port = 6666; //порт
try {
    ServerSocket ss = new ServerSocket(port); //создаем сокет
    Socket socket = ss.accept();
    InputStream sin = socket.getInputStream();
    OutputStream sout = socket.getOutputStream(); // потоки
ввода и вывода

    DataInputStream in = new DataInputStream(sin);
    DataOutputStream out = new DataOutputStream(sout);

    String line = null;
    while(true) {
        line = in.readUTF(); // считываем строку от клиента
        out.writeUTF(getFormula(line.charAt(0) - '0',
line.charAt(2) - '0')); //вызываем подпрограмму, описанную выше
        out.flush(); //закрытие потока
    }
} catch(Exception x) { x.printStackTrace(); }
```

4. Клиент на Java:

```
int serverPort = 6666; // порт сервера
String address = "127.0.0.1"; //IP компьютера с сервером
try {
    InetAddress ipAddress = InetAddress.getByName(address);

    Socket socket = new Socket(ipAddress, serverPort);
// подключение к серверу
    InputStream sin = socket.getInputStream();
    OutputStream sout = socket.getOutputStream();
//входной и выходной потоки
    DataInputStream in = new DataInputStream(sin);
    DataOutputStream out = new DataOutputStream(sout);
```

```
    String line = null;
    Reader r0 = new Reader("Выберите тест: ", 1);
    r0.setVisible(true);
    r0.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    r0.setSize(200,200);
    r0.setResizable(false);
    r0.setLocationRelativeTo(null);
    while (!r0.getT()) {
        Thread.sleep(10);
    }
//создали окно, учащийся выбрал номер теста который хочет пройти
    Scanner fin = null;
    try {
     fin = new Scanner(new File("test" + r0.getF1() + ".txt"));
//считываем из файла, в котором хранятся вопросы этого теста
        int k = 0; //счетчик правильных ответов
        Reader r = new Reader("");
        while (fin.hasNext()) {
            r.setL1(fin.nextLine());
            r.setVisible(true);
            r.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            r.setSize(200,200);
            r.setResizable(false);
            r.setLocationRelativeTo(null);
//открыли тестирующее окно
            out.writeUTF(fin.nextLine());
//отправили номер формулы на сервер
            out.flush();
            line = in.readUTF();
//получили верную формулу
            while (!r.getT()) {
                Thread.sleep(10);
            }
            if(line.equals(r.getF1())) {
                r.setL2("ПРАВИЛЬНО!");
                k++;
            } else {
                r.setL2("НЕПРАВИЛЬНО!");
            }
//сравнили ответ с правильным, вывели результат
            r.setT(false);
            Thread.sleep(2000);
            r.setF1("");
            r.setL2("");
        }
        Reader r1 = new Reader("Ваша оценка: " + k, 0);
        r1.setVisible(true);
        r1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        r1.setSize(200,200);
        r1.setResizable(false);
        r1.setLocationRelativeTo(null);
//вывели отметку, которую заработал учащийся
    }catch (FileNotFoundException e) {
```

```
            System.out.println("Не найден файл" + e);
        }
        finally {
            if (fin != null) {
                fin.close();
            }
        }
    } catch (Exception x) {
        x.printStackTrace();
    }
```

5. Класс окон:

```
public class Reader extends JFrame {
    private JButton b1, b2; //кнопки
    private JLabel l1, l2; //строки
    private JTextField f1; //поле ввода
    private boolean t; //показывает, готов ли учащийся отослать
результат
    eHandler handler = new eHandler(); //слушатель действий
    public Reader (String s) {
        super("Geometry");
        setLayout(new FlowLayout());
        b1 = new JButton("Очистить");
        b2 = new JButton("Проверить");
        l1 = new JLabel(s + "\n");
        l2 = new JLabel("");
        f1 = new JTextField(15);
        t = false;
        add(b1);
        add(b2);
        add(l1);
        add(f1);
        add(l2);
        b1.addActionListener(handler);
        b2.addActionListener(handler);
    }
//описание тестирующего окна
    public Reader (String s, int a) {
        super("Geometry");
        setLayout(new FlowLayout());
        l1 = new JLabel(s + "\n");
        add(l1);
        if(a == 1) {
            b2 = new JButton("Выбрать");
            f1 = new JTextField(6);
            t = false;
            add(f1);
            add(b2);
            b2.addActionListener(handler);
        }
    }
```

```
//описание окна выбора теста и вывода отметки
    public String getF1() {
        return f1.getText();
    }
    public boolean getT() {
        return t;
    }
    public void setL2(String st) {
        l2.setText(st);
    }
    public void setL1(String st) {
        l1.setText(st);
    }
    public void setT(boolean t) {
        this.t = t;
    }
    public void setF1(String st) {
        f1.setText(st);
    }
//геттеры и сеттеры
    public class eHandler implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e) {
            if(e.getSource() == b1) {
                f1.setText(null);
            }
//нажатие на кнопку b1 очищает поле ввода
            if(e.getSource() == b2) {
                t = true;
            }
//нажатие на кнопку b2 показывает готовность отослать ответ на
сервер
        }
    }
//описание слушателя действий
}
```