МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования
«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики
Кафедра современных технологий программирования

КЛИМОВА ЮЛИЯ АЛЕКСАНДРОВНА

Тестирующее приложение «Simple Testing»

Курсовая работа
по дисциплине «Алгоритмизация и программирование»
студентки 2 курса специальности
1-26 03 01 «Управление информационными ресурсами»
дневной формы получения образования

Научный руководитель
Гуща Юлия Вальдемаровна,
старший преподаватель
кафедры современных технологий
программирования

Гродно 2021

РЕЗЮМЕ

Тема курсовой работы

«Тестирующее приложение «Simple Testing»

Работа содержит 16 страниц, 5 рисунков, 8 листингов, 1 приложение, 2 использованных источника.

Ключевые слова: тест, тестирующее приложение, Java, JavaFX, объектно-ориентированное программирование, приложение, классы, методы.

Целью курсовой работы – разработка приложения с тестами, посвященными языкам программирования и концепции объектно-ориентированного программирования.

Объектом курсовой работы выступают тесты с возможностью выбора ответа.

Предметом исследования является разработка приложения с тестами.

Программа служит для проверки своих знаний теории в области программирования.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С каждым годом люди пытаются сделать обучение проще, удобнее и быстрее. Уже придумано множество приложений и программного обеспечения, которое упрощает процесс обучения, в том числе и самостоятельного обучения.

Одним из способов получения полезной информации и знаний является прохождение тестов. С помощью прохождения тестов человек сможет не только проверить свои знания, но заодно запомнить тот или иной термин, понятие и т.д.

Существует несколько видов тестов:

- С открытыми вопросами;
- С закрытыми вопросами;
- С возможностью выбора ответов.

В свою очередь тесты с закрытыми вопросами можно классифицировать так:

- задания альтернативных ответов;
- задания множественного выбора;
- задания на восстановление соответствия;
- задания на установление правильной последовательности.

А тесты с открытыми вопросами могут быть следующих типов:

- задания свободного изложения;
- задания-дополнения.

Тестирование в педагогике выполняет три основные взаимосвязанные функции: диагностическую, обучающую и воспитательную:

- Диагностическая функция заключается в выявлении уровня знаний, умений, навыков учащегося. Это основная и самая очевидная функция тестирования. По объективности, широте и скорости диагностирования, тестирование превосходит все остальные формы педагогического контроля.
- Обучающая функция тестирования состоит в мотивировании учащегося к активизации работы по усвоению учебного материала. Для усиления обучающей функции тестирования могут быть использованы дополнительные меры стимулирования студентов, такие как: раздача преподавателем примерного перечня вопросов для самостоятельной подготовки, наличие в самом тесте наводящих вопросов и подсказок, совместный разбор результатов теста.
- Воспитательная функция проявляется в периодичности неизбежности тестового контроля. Это дисциплинирует, организует и направляет деятельность учащихся, помогает выявить и устранить пробелы в знаниях, формирует стремление развить свои способности.

Также тестирование имеет свои преимущества и недостатки.

Достоинства:

- Тестирование является более качественным и объективным способом оценивания, его объективность достигается путём стандартизации процедуры проведения, проверки показателей качества заданий и тестов целиком.
- Тестирование — более справедливый метод, оно ставит всех учащихся в равные условия, как в процессе контроля, так и в процессе оценки, практически исключая субъективизм преподавателя.
- Тесты — это более объёмный инструмент, поскольку тестирование может включать в себя задания по всем темам курса.

Недостатки:

- Данные, получаемые преподавателем в результате тестирования, хотя и включают в себя информацию о пробелах в знаниях по конкретным разделам, но не позволяют судить о причинах этих пробелов.
- Тест не позволяет проверять и оценивать высокие, продуктивные уровни знаний, связанные с творчеством, то есть вероятностные, абстрактные и методологические знания.
- В тестировании присутствует элемент случайности. Например, учащийся, не ответивший на простой вопрос, может дать правильный ответ на более сложный. Причиной этого может быть, как случайная ошибка в первом вопросе, так и угадывание ответа во втором. Это искажает результаты теста и приводит к необходимости учета вероятностной составляющей при их анализе [1].

Целью курсовой работы является разработка приложения с тестами, посвященными языкам программирования и концепции объектно-ориентированного программирования.

Для достижения казанной цели необходимо решить следующие задачи:

1. Выбрать тип вопросов тестирования;
2. Разработать интерфейс приложения;
3. Разработать классы, необходимые для выполнения необходимых задач;
4. Разработать функционал с учётом всех требований;
5. Протестировать приложение.

1 ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

1.1 Постановка задачи

Приложение «Simple Testing» представляет собой набор тестов. тестов. Изначально здесь представлены 3 теста: «Тест на знание ООП», «Тест на знание Java», «Тест на знание C#». Каждый тест содержит по 10 вопросов, в каждом из которых 4 варианта ответа. Правильный ответ только один. В конце каждого теста демонстрируется результат прохождения теста пользователем. Если пользователь ответил на 1-3 вопроса, то уровень его знаний низкий, если на 4-7, то уровень средний, а если на 8-10, то высокий. Также имеется возможность сгенерировать новый тест из вопросов. [2]

Предполагаемая область применения данного приложения – это проверка собственных знаний в области программирования. Также программа может быть использована преподавателем в качестве проверки знаний студентов.

Приложение должно быть реализовано на языке Java с помощью библиотеки JavaFX. Приложение должно предоставить:

- возможность пройти тесты;
- возможность узнать количество вопросов, на которые был дан правильный ответ;
- возможность пройти тест, в котором вопросы из предыдущих тестов будут сгенерированы в случайном порядке;
- возможность выйти из режима прохождения теста, когда тест ещё не закончен;

Кроме того, приложение должно обладать простым интерфейсом, который не будет отвлекать пользователя от прохождения теста.

1.2 Классы

При реализации было разработано шесть классов: Question, MainController, CsharpController, JavaController, OopController и NewController. Каждый из классов отвечает за свою область в программе и имеет свой набор методов. В совокупности этих классов программа функционирует согласно заданию.

Класс Question был создан для описания формулировки вопроса и ответов. В данном классе присутствует конструктор, который принимает название вопроса типа String и ответы в виде строкового массива.

```
    public Question(String name, String[] answers) {
              this.name = name;
              this.answers = answers;
    }
```

Листинг 1.1 – Конструктор класса Question

Также класс содержит метод correctAnswer(), который возвращает правильный ответ. В свою очередь правильный ответ – тот, который находиться на самом последнем месте в массиве.

```
    //правильный ответ - последний элемент массива
    public String correctAnswer(){
              return this.answers[answers.length-1];
    }
```

Листинг 1.2 – Метод CorrectAnswer() в классе Question

1.3 Формы

Классы MainController, CsharpController, JavaController, OopController и NewController которые хранятся в папке Controllers, отвечают за элементы управления на форме и их функционал.

Класс MainController отвечает за главное меню. В нём расположены обработчики событий, которые включают форму теста, выбранного пользователем.

```
    javaButton.setOnAction(actionEvent -> {
              try {
                   javaButton.getScene().getWindow().hide();
                   FXMLLoader loader = new FXMLLoader();
loader.setLocation(getClass().getResoure("/sample/UI/sampleJava.
fxml"));
                   loader.load();
                   Parent root = loader.getRoot();
                   Stage stage = new Stage();
                   stage.setScene(new Scene(root));
                   stage.show();
                   } catch (Exception e) {
                        e.printStackTrace();
                   }
              });
```

Листинг 1.3 – Обработчик события в классе MainController

Классы CsharpController, JavaController, OopController и NewController предназначены для самих форм с тестами. Данные классы работают одинаково, за исключением класса NewController, где вопросы из предыдущих тестов перемешаны. Все классы содержат метод initialize(), в котором описана то как конкретно работает тест. Помимо этого присутствует метод modalWindow(), который создаёт и выводит модальное окно, если пользователь нажимает на кнопку «Exit» до того как тест завершён.

```
    private static void modalWindow(){
        Stage window = new Stage();
        window.initModality(Modality.APPLICATION_MODAL);
        Pane pane = new Pane();
        Button yesButton = new Button("Yes");
        yesButton.setLayoutX(60);
        yesButton.setLayoutY(100);
        Button noButton = new Button("No");
        noButton.setLayoutX(120);
        noButton.setLayoutY(100);
        noButton.setOnAction(event -> window.close());
        yesButton.setOnAction(event -> {
            try {
                noButton.getScene().getWindow().hide();
    FXMLLoader loader = new FXMLLoader();
loader.setLocation(CsharpController.class.getResource("/sample/U
I/sampleMain.fxml"));
                loader.load();
                Parent root = loader.getRoot();
                Stage stage = new Stage();
                stage.setScene(new Scene(root));
                stage.showAndWait();
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
        pane.getChildren().addAll(yesButton,noButton);
        Scene scene = new Scene(pane,200,200);
        window.setTitle("Are you sure you want to exit?");
        window.setScene(scene);
        window.setResizable(false);
        window.showAndWait();
    }
```

Листинг 1.4 – Метод modalWindow() в классе JavaController

Также в классах определен метод hideAllControls(), который скрывает все радио кнопки и кнопку ответа после того как пользователь ответил на все вопросы в тесте.

```
    private void hideAllControls() {
```

```
        radioButton1.setVisible(false);//если                вопрос
закончились, то скрываем кнопки
        radioButton2.setVisible(false);
        radioButton3.setVisible(false);
        radioButton4.setVisible(false);
        answerButton.setVisible(false);
    }
```

Листинг 1.5 – Метод hideAllControls() в классе JavaController

Как ранее упоминалось в классе NewController вопросы с ответами перемешаны в случайном порядке. За это отвечают следующие строки кода:

```
List<Question>                    questionList                    =
Arrays.asList(questions);//преобразовываем в список
Collections.shuffle(questionList);
```

Листинг 1.6 – Сортировка в случайном порядке в классе NewController

Все вопросы для каждого теста считываются из XML-файлов методом десериализации. Для каждого теста предназначен свой файл, имя которого записано в виде константы вида FILE_NAME_JAVA, где последнее слово идентифицирует конкретный тест. Для записи в файл используется такая технология как сериализация, она реализуется с помощью метода serialize() в классах CsharpController, JavaController, OopController и NewController (ПРИЛОЖЕНИЕ, Листинг 1). За чтение из XML-файлов отвечает метод deserialize().

```
    public Question[] deserialize() throws Exception {
        FileInputStream        fileInputStream        =        new
FileInputStream(FILE_NAME_JAVA);
        XMLDecoder     xmlDecoder     =     new     XMLDecoder(new
BufferedInputStream(fileInputStream));
        xmlDecoder.readObject();
        xmlDecoder.close();
        fileInputStream.close();
        return questions;
    }
```

Листинг 1.7 – Метод deserialize() в классе JavaController

2 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

2.1 Главная форма

В результате успешного запуска приложения появляется основная форма (представлена на рисунке 2.1). На данной форме расположено главное меню. Оно включает в себя четыре кнопки: «C# Test», «Java Test», «OOP Test» и «New Test», а также приветственную надпись.

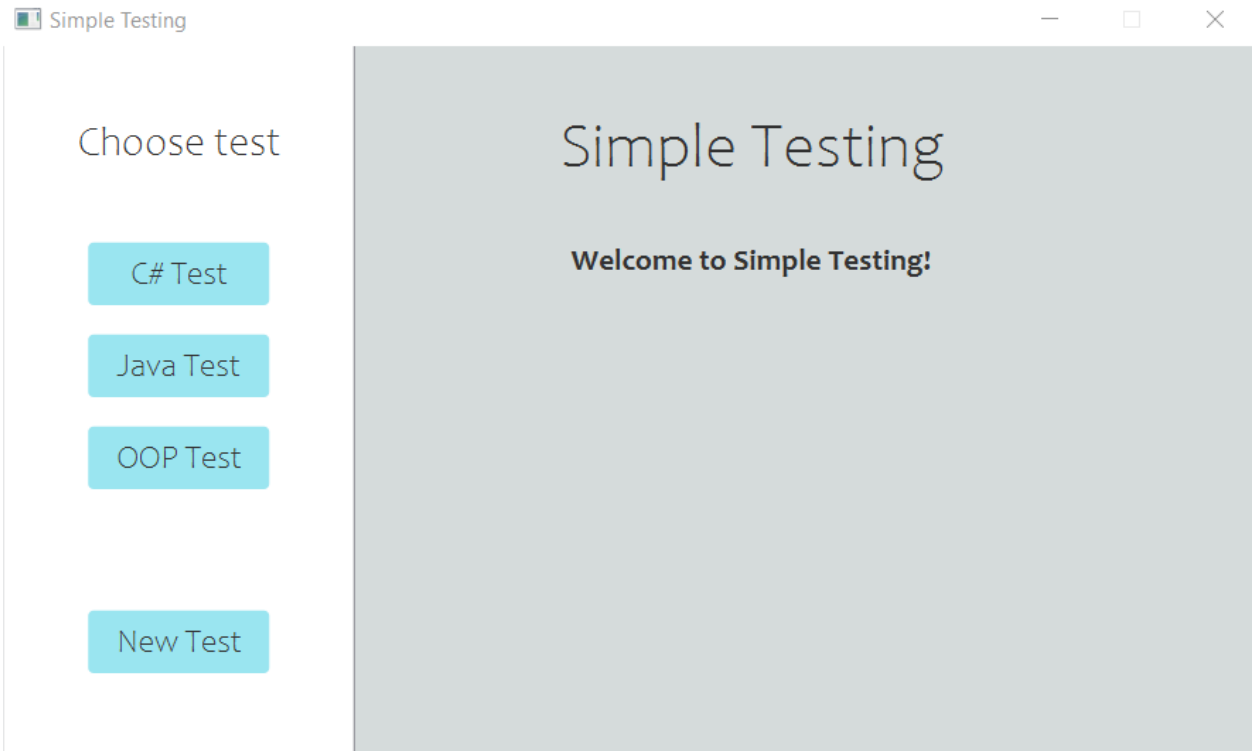

Рисунок 2.1 – Изображение главной формы

2.2 Вспомогательные формы

Для прохождения интересующего теста нажмите кнопку с соответствующей надписью:

- «C# Test» для прохождения теста на знание языка C#;
- «Java Test» для прохождения теста на знание языка Java;
- «OOP Test» для прохождения теста на знание концепции ООП;
- «New Test» для прохождения теста, состоящего из перемешанных вопросов из вышеперечисленных тестов.

При нажатии появятся новые формы, которые работают одинаково, а

именно выводят формулировку вопроса в виде надписи и список ответов в виде радио кнопок, где пользователю предстоит выбрать только один правильный ответ. Кнопка «Answer» переходит на следующий вопрос. Во время прохождения теста кнопки «C# Test», «Java Test», «OOP Test» и «New Test» скрыты. Также на форме располагается кнопка «Exit», которая позволяет выйти в главное меню.

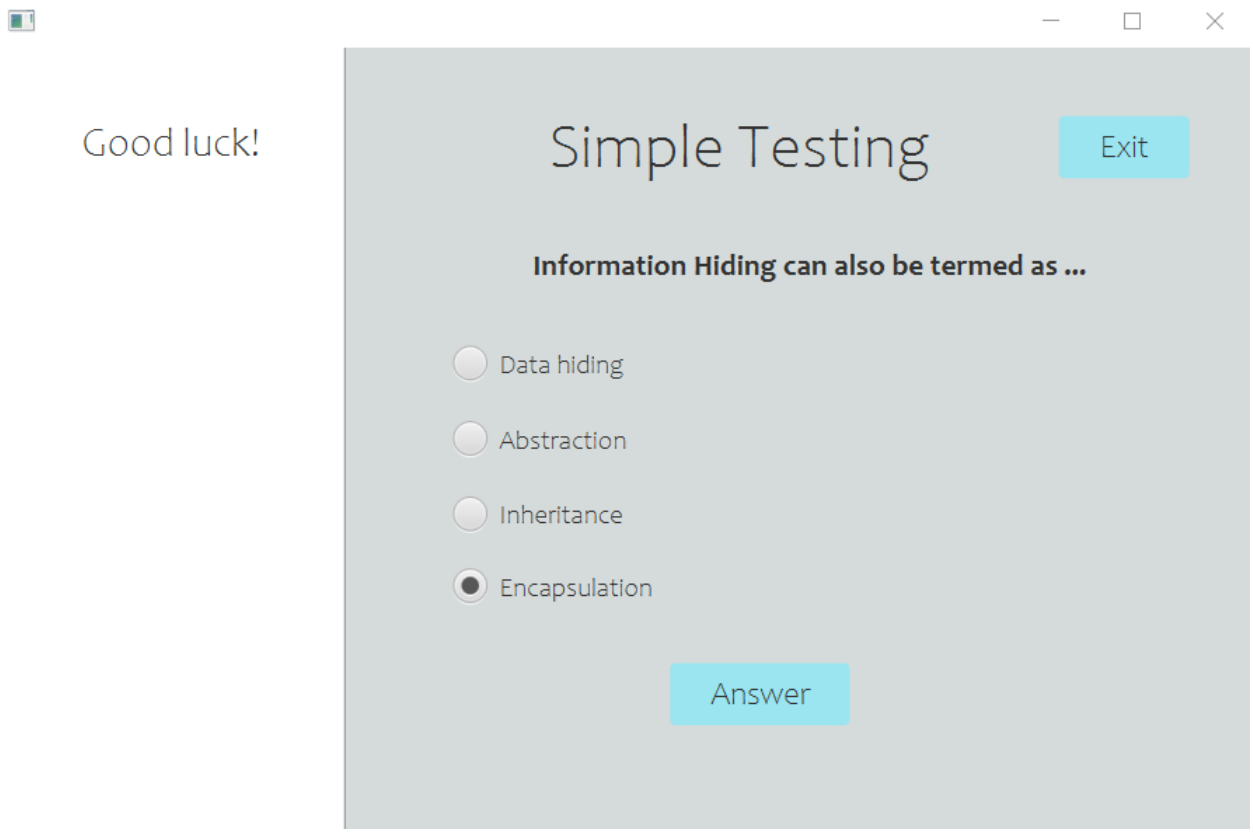

Рисунок 2.2 – Изображение формы «OOP Test»

Если вы желаете прекратить тест, нажмите на кнопку «Exit». В этом случае появится окно, для подтверждения вашего решения выйти в главное меню, не завершив тест (представлено на рисунке 2.3). Нажмите кнопку кнопку «Yes», если желаете вернуться на экран со списком тестов. Если же надо вернуться и продолжить тест, тогда нажмите кнопку «No».

Рисунок 2.3 – Изображение окна на подтверждение решениях о выходе из теста

Если при прохождении теста пользователь не выбрал ответ, но нажал на кнопку «Answer», то появляется окно, сообщающее об ошибке. При закрытии данного окна ничего не изменится.

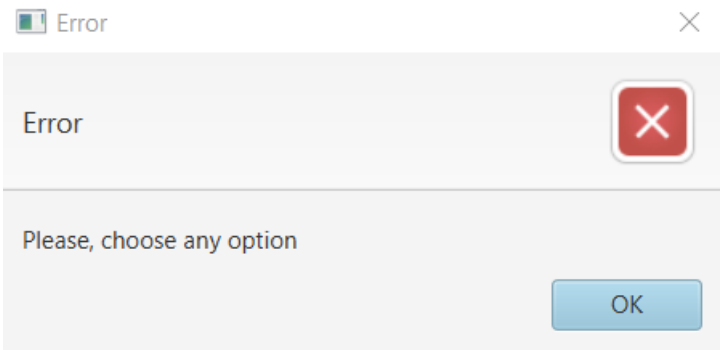

Рисунок 2.4 – Изображение окна, сообщающего об ошибке

По окончанию прохождения теста надпись, которая отвечала за формулировку вопроса меняет свой текст на результат пользователя. Также на форме появляется надпись, которая говорит о том, какими знаниями обладает пользователь в той или иной области. При нажатии на кнопку «Exit» пользователь сразу попадает в главное меню.

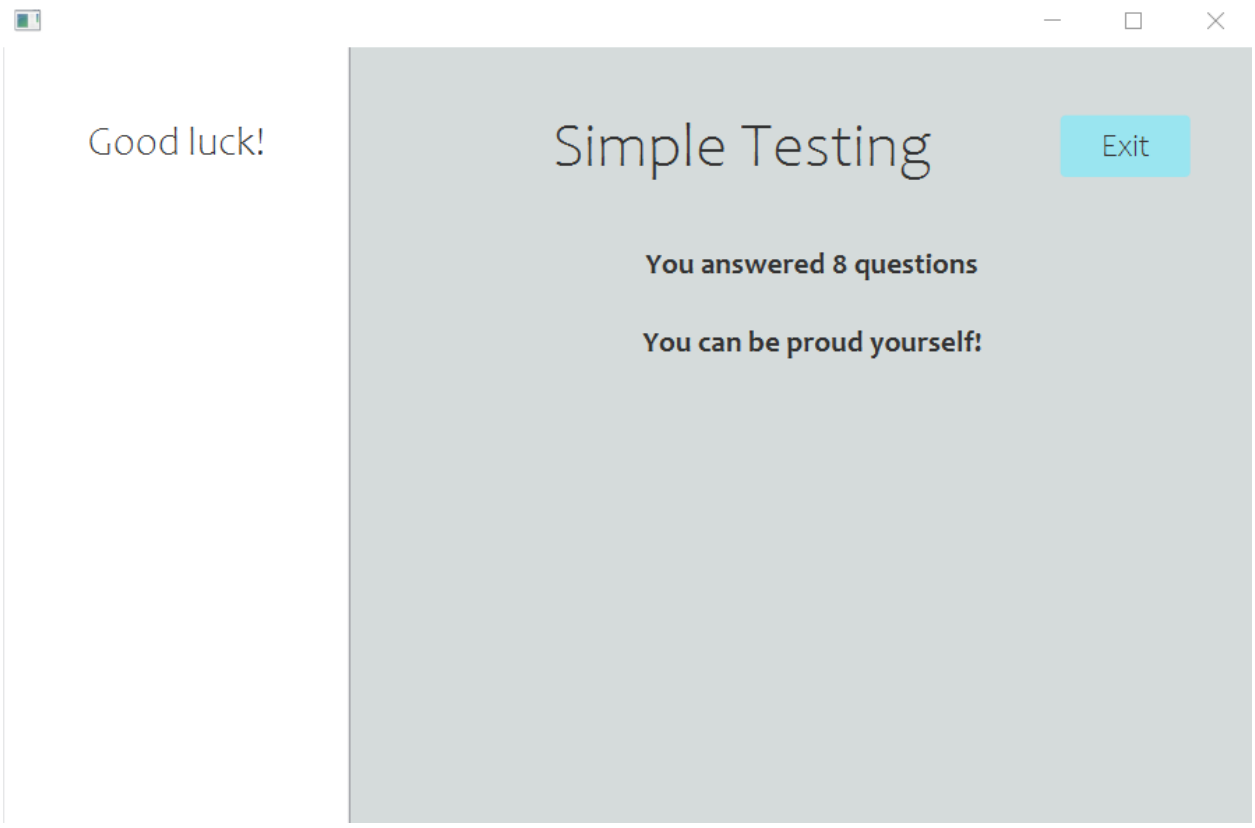

Рисунок 2.5 – Изображение формы после окончания прохождения теста

ЗАКЛЮЧЕНИЕ

Развитие современных информационных технологий значительно повлияло на процесс обучения. Тестирование является одним из способов запоминания информации, что подтверждает актуальность выбранной темы курсовой работы.

При написании курсовой работы было спроектировано и реализовано тестирующее приложение под названием «Simple Testing» в соответствии с техническим заданием.

Программа интуитивно проста и понятна, для ее использования не нужно специального обучения пользователя.

Для этого разработаны алгоритмы и программы для реализации приложения с использованием современных технологий программирования. Программное приложение реализовано с использованием объектно-ориентированной технологии программирования. Тестирование, разрабатываемого приложения показало работоспособность тестирующей программы, целостность и структурированность.

Предполагаемая область применения тестирующего приложения «Simple Testing» – это проверка собственных знаний в области программирования. Также программа может быть использована преподавателем в качестве проверки знаний студентов.

Таким образом, поставленные цели и задачи курсовой работы были решены. Однако данная работа не претендует на глубину теоретического и практического уровней и может быть продолжена или усовершенствована.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Аванесов, В.С. Композиция тестовых заданий. / В.С. Аванесов М.,Центр тестирования, 2002. – 128 – 133 с.

2 Климова, Ю.А. Разработка тестирующего приложения "Simple Testing" с использованием JavaFX [Электронный ресурс] / Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях. Материалы XXIV Республиканской научной конференции студентов и аспирантов, 22-24 марта 2021 г.

ПРИЛОЖЕНИЕ

```
public void serialize(Question[] questions) throws Exception {
        FileOutputStream       fileOutputStream       =       new
FileOutputStream(FILE_NAME_JAVA);
        XMLEncoder           xmlEncoder           =           new
XMLEncoder(fileOutputStream);
        xmlEncoder.writeObject(questions);
        xmlEncoder.close();
        fileOutputStream.close();
    }
```

Листинг 1 – Метод для serialize() в классе JavaController