**Министерство образования и науки Республики Казахстан**

**КГУ «Костанайский казахско-турецкий лицей-интернат для одаренных детей»**

**Выполнили:** Шайкемелов Теймур 11Б класс

Ким Владимир 10Б класс

**Тема: «Онлайн тестирования в школах»**

**Секция**: Информатика

**Руководитель**: Маратов А.

**Костанай, 2014**

**Отзыв**

**на научную работу ученика 11 класса казахско-турецкого**

**лицей-интерната для одаренных детей на тему:**

**«Онлайн тестирования в школах»**

Авторов работы Шайкемелова Теймура и Ким Владимира интересует программирование, поэтому они избрали темой своей научной работы «Онлайн тестирование в школах».

Они в течение нескольких месяцев самостоятельно и глубоко изучали Java и искали способы решения этой проблемы.

В исследовательской части они грамотно описали актуальность данной темы и выбрали подходящий язык программирования для разработки данной программы.

Также они достоверно описали главные классы программы, что доказывает то, что они делали эту программу самостоятельно.

В заключении ученики самостоятельно и грамотно сделали выводы, полученные в процессе исследования, определили актуальность данной темы, также привели новые идеи по улучшению проекта.

Авторами проекта проделана серьезная и большая работа по изучению Java и разработки программы.

Цели и задачи, поставленные учениками по данной теме выполнены. Материалы работы и выводы исследования можно использовать для дальнейшего использования в учебных заведениях. Данная работа могла бы помочь преподавателям с проверкой знаний учащихся.

Учитель информатики: Маратов А.М.

**Содержание**

Введение.................................................................................................................1

Исследовательская часть………………………………………………......................................2

Описание серверной части (Для преподавателя)................................................3

Описание программного кода серверной части……………………………………………….4-6

Клиентская часть программы(Для учащегося).....................................................7

Описание программного кода клиентской части……………………………………………..8

Демонстрация программы…………………………………………………………………………………9

Заключение.............................................................................................................10

Инструкция по использованию для преподавателя…………………………………………11

Инструкция по использованию для учащегося…………………………………………………12

Использованная литература………………………………………………………………………………13

**Аннотация**

Целью данной работы является задача онлайн тестирования в школах, а также проверка текущих знаний учеников и проведение обычных контрольных работ.

**Annotation**

The aim of this work is the task of online testing in schools, as well as verification of current knowledge of the students and conduct routine examinations.

**Введение**

В наше время очень сильно развиты информационные технологии. Все стараются перейти на использование этих технологий. Все модернизируется. Теперь можно не тратить время на ту или иную задачу.

XXI век – век информационных технологий. У всех сейчас есть планшеты, смартфоны, ультрабуки и т.д. У нас появилась идея сделать программу, благодаря которой можно будет проводить тестирования в школах, не тратя при этом много ресурсов и времени.

В частности, наша программа поможет преподавателю проверить насколько был усвоен материал изученного предмета.

**Исследовательская часть**

В интернете очень много разных способов проверить свои знания. Достаточно зайти в Google и ввести онлайн тестирования, но вы получите лишь возможность написать пробный ЕНТ. Там вы не найдете сайт, который предоставит вам возможность пройти тест по определенному материалу с нужным уровнем заданий.

Программа была написана на объектно-ориентированным языке программирования - *Java.* Почему именно *Java*? Java является основой ряда платформ, предназначенных для разработки программного обеспечения и обеспечивающих эффективную работу на системах с разной чип-архитектурой. Java помогает решать задачи разработчикам серверных, клиентских и встраиваемых систем. Одно из главных преимуществ Java – его независимость от платформы, на которой выполняются программы. Таким образом, один и тот же код можно запускать на разных *ОС*. Средой разработки был выбран *Eclipse*.

*Цель нашей программы* – проведение упрощённого тестирования по пройденному материалу в помощь преподавателю.

**Описание серверной части (Для преподавателя).**

Запустив программу преподавателя, вы увидите небольшое окно, с разными вкладками.

**Первая вкладка** – это “News”, в этой вкладке вы найдете новые функции программы, а также ссылку, чтобы проверить доступность новой версии программы.

**Вторая** – основная вкладка, в которой создается тест для учащихся, а также задаются критерии оценивания в 100 бальной системе. Указывается в каких интервалах будет определённая оценка и установлено время для выполнения тестовых заданий. Также можно сохранить созданный тест для дальнейших использований и открыть уже сохранённый.

**В третьей вкладке** преподаватель может посмотреть результаты уже закончивших тестирования учащихся, с подсчетом правильных и не правильных ответов.

**Четвертая вкладка** содержит в себе информацию о сервере. В ней есть IP адрес, с помощью которого учащиеся будут соединяться.

**В пятой вкладке** указана краткая инструкция преподавателю по использованию данной программы.

**В шестой вкладке** указаны контактные данные разработчика данной программы, с помощью которых вы можете отправить сообщение с пожеланиями и предложениями по улучшению программы.

**В седьмой вкладке** указана информация о разработчиках.

**Описание программного кода серверной части.**

В каждой программе есть главный класс, в моей этот класс тоже существует. Он называется *Main*. В этом классе написан код который реализует большую часть визуального оформления. Также в этом классе вызываются другие классы таки как *Network*.

В классе *Network* задана реализация сетевого соединения. Рассмотрим код ниже.

**public** **class** Network **implements** Runnable{

Ниже объявляются переменные

**protected** **static** ServerSocket *serverSocket* = **null**;

**protected** **static** Socket *clientSocket* = **null**;

**protected** **static** clientThread *t*[] = **new** clientThread[10];

**protected** **static** InetAddress *ipAddress*;

@SuppressWarnings("static-access")

@Override

**public** **void** run() {

System.***out***.println("Network.java/run()");

**try** {

Создается сервер на порте 5678

*serverSocket* = **new** ServerSocket(5678);

Переменной ipAddress задается значение IP адреса сервера

*ipAddress* = *serverSocket*.getInetAddress().*getLocalHost*();

Через метод передается переменная

Stats.*setIpAddress*(*ipAddress*);

}

**catch** (IOException e)

{System.***out***.println(e);}

**while**(**true**){

**try** {

Слушает подключения

*clientSocket* = *serverSocket*.accept();

**for**(**int** i=0; i<=9; i++){

**if**(*t*[i]==**null**){

Создается новый отдельный поток для каждого соединения

(*t*[i] = **new** clientThread(*clientSocket*,*t*,i)).start();

**break**;

}

}

}

**catch** (IOException e) {

System.***out***.println(e);}

}

}

}

Как вы поняли в классе *Network* создается новый поток для каждого соединения. Рассмотрим часть кода из класса *clientThread*.

**public** **void** run() {

System.***out***.println("clientThread.java/run()");

**try** {

Отправляется тест клиенту

*output* = **new** ObjectOutputStream(clientSocket.getOutputStream());

*output*.writeObject(CreateTest.*getLabPanel*());

Принимается ответ от клиента

input = **new** ObjectInputStream(clientSocket.getInputStream());

Создается новый поток, который проверяет ответы

**new** Thread(**new** Checker((String[])input.readObject(),*t*,k)).start();

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

}

По аналогии из класса *clientThread* вызывается класс *Checker*. Этот класс проверяет ответы учеников и добавляет результат в таблицу, которую можно увидеть во вкладке *Result*. Также рассмотрим часть кода.

В этом фрагменте кода проверяется неправильное количество ответов

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

**if**(teacherCharAnswer[i]!=userCharAnswer[i]){

*incorrect*++;

}

}

Здесь подсчитывается правильное количество ответов.

*correct* = teacherCharAnswer.length - *incorrect*;

Ниже результат добавляется в таблицу, а затем таблица перерисовывается

Result.*help\_to*.add(**new** HelpToResult(userName, userSurname, *correct*, *incorrect*,

xPercent, mark));

Result.*jtab*.revalidate();

Ну и конечно же самый важный класс - *CreateTest*, в котором создается сам тест. Этот класс очень большой и содержит в себе много кода, который в тоже время и не такой уж и сложный для объяснения, но важный. Каждый вопрос в тесте – это панель. *Тест* – массив панелей.

В этом методе добавляются панели в *ArrayList*

**public** **static** **void** addPanel(**int** newNumber, JPanel newPanel){

System.***out***.println("CreateTest.java/setPanel()");

JPanel addpanel = newPanel;

*panels*.add(addpanel);

addpanel.setAlignmentX(JLabel.***CENTER\_ALIGNMENT***);

addpanel.setFont(*font*);

addpanel.setPreferredSize(**new** Dimension(200,300));

*labPanel*.add(addpanel);

*scrollPane*.revalidate();

}

**Клиентская часть программы**

Запустив программу преподавателя, вы увидите небольшое окно, с полями ввода данных – это имя, фамилия и IP адрес, а также кнопку *Connect*. После того как вы нажмете кнопку *Connect* перед вами появится окно, схожее с серверной частью программы.

**Первая вкладка** – это “News”, в этой вкладке вы найдете новые функции программы, а также ссылку, чтобы проверить доступность новой версии программы.

**Вторая** – основная вкладка, в которой учащийся может по нажатию кнопки *Open* начать тест.

**В третьей вкладке** учащийся может посмотреть свой результат, с подсчетом правильных и не правильных ответов, и выставленной оценкой.

**В четвертой вкладке** указана краткая инструкция учащемуся по использованию данной программы.

**В пятой вкладке** указаны контактные данные разработчика данной программы, с помощью которых вы можете отправить сообщение с пожеланиями и предложениями по улучшению программы.

**В шестой вкладке** указана информация о разработчиках.

**Описание программного кода клиентской части.**

Главным классом здесь является – *Login*. В этом классе реализуется маленькое окно, в котором будет вводиться имя, фамилия и IP адрес сервера, чтобы подключиться. Из класса *Login* вызывается конструктор класса *Menu*, и создается новый поток и конструктор класса *Network*.

В классе *Menu* реализована визуальная часть программы. Как вы уже поняли класс *Network* отвечает за соединение с сервером. Рассмотрим код ниже.

Этот код реализует подключение к серверу, через порт *5678*. После принимается тест, который уже расположен на *JPanel*. Затем вызывается конструктор класса *Test*, оттуда отправляется уже решенный тест. Потом после того как решение было отправлено, клиент принимает результат теста и добавляет его в таблицу.

*connection* = **new** Socket(inetAddress,5678);

*input* = **new** ObjectInputStream(*connection*.getInputStream());

*output* = **new** ObjectOutputStream(*connection*.getOutputStream());

*labPanel* = (JPanel)*input*.readObject();

Menu.*changeTask*();

**new** Test(*labPanel*);

**int**[] n = (**int**[]) *input*.readObject();

Result.*help\_to*.add(**new** HelpToResult(n[0], n[1], n[2], n[3]));

Result.*jtab*.revalidate();

Рассмотрим класс *Test*. В этом классе нет ничего сложного, код очень прост в написании и понимании. После того как учащийся выполнит тест. Программа забьет отмеченные ответы, с именем и фамилией в массив строк и отправит на сервер для проверки, а окно с выполнением теста закроется.

**for**(**int** i=0;i<*panels*.size();i++){

JPanel jp = *panels*.get(i);

JRadioButton jrb1 = (JRadioButton)jp.getComponent(1);

JRadioButton jrb2 = (JRadioButton)jp.getComponent(2);

JRadioButton jrb3 = (JRadioButton)jp.getComponent(3);

JRadioButton jrb4 = (JRadioButton)jp.getComponent(4);

JRadioButton jrb5 = (JRadioButton)jp.getComponent(5);

**if**(jrb1.isSelected() == **true**){*answer*.append("A");}

**if**(jrb2.isSelected() == **true**){*answer*.append("B");}

**if**(jrb3.isSelected() == **true**){*answer*.append("C");}

**if**(jrb4.isSelected() == **true**){*answer*.append("D");}

**if**(jrb5.isSelected() == **true**){*answer*.append("E");}

}

String[] send = **new** String[3];

send[0] = Login.*getUserName*();

send[1] = Login.*getSurname*();

send[2] = *answer*.toString();

Network.*sendData*(send);

*jframe*.dispose();

}

**Демонстрация программы**

В качестве примеры мы проведем тест по русскому языку на тему фонетика.

1. В разделе «Фонетика» изучается …   
   A) словосочетание и предложение   
   B) состав слова   
   C) правила, определяющие написание слов   
   D) словарный состав языка   
   E) звуки речи   
   Ответ: Е
2. Сколько согласных звуков в русском языке?

A) 36

B) 34

C) 33

D) 37

E) 38

Ответ: A

1. Определите слово, в котором больше букв, чем звуков   
   A) Сосна   
   B) Бобр   
   C) Липа   
   D) Дождь   
   E) Смородина   
   Ответ: D
2. Укажите слово, в котором нет звонких согласных звуков   
   A) Беглянка   
   B) Мольба   
   C) Обзор   
   D) Копать   
   E) Кашевар   
   Ответ: D
3. В каком слове все согласные звуки мягкие?   
   A) Широкий   
   B) Веселье   
   C) Встреча   
   D) Суровый   
   E) Мишень   
   Ответ: В
4. Каким фонетическим признаком объединены следующие слова: подсечь, вязкий, ложка, обтесать?   
   A) Оглушение согласных на конце слова   
   B) Оглушение согласных перед другими согласными   
   C) Выпадение согласных   
   D) Смягчение согласных перед другими согласными   
   E) Озвончение согласных   
   Ответ: В
5. Сколько гласных звуков в русском языке?

A) 7   
B) 5   
C) 6   
D) 8   
E) 4   
Ответ: С

**Заключение**

Как вы увидели мы рассмотрели основную часть кода программы. Программа работает, но она все еще не совершенна.

У нас есть много идей по улучшению нашей программы. Например, написать версию программы на *Android*. Так как, у всех есть планшеты и сенсорные телефоны, с помощью которых учащиеся смогут использовать данную программу.

Также в панели *Stats* необходимо сделать мониторинг подключенных клиентов, чтобы преподаватель мог удостовериться, что все учащиеся были подключены.

Планируется создать возможность сохранения теста и открытие уже сохранённого.

Существует идея по улучшению самого теста, то есть динамическое создание разных вариантов, с модулями.

Программа нуждается в базе данных, с помощью которой учащийся и преподаватель мог посмотреть предыдущие тестирования. Кроме того, с базой данных можно добавить много новых возможностей.

Программа была протестирована среди учащихся и успешно прошла тестирование. Учащимся понравилась программа, так как она проста в использовании.

**Инструкция по использованию для преподавателя**

1. Вы должны запустить программу.
2. Открыть вкладку *Test*.
3. В этой вкладке вы должны ввести критерии оценивания и время для выполнения теста.
4. Затем нажать кнопку *Create*, чтобы создать тест.
5. В появившемся окне, чтобы добавить новый вопрос нужно нажать кнопку *Add* new.
6. Во вкладке Add *new* question есть поля. В первое поле нужно ввести вопрос, также по желанию можно добавить картинку к этому вопросу. После нужно ввести варианты ответа, среди которых должен быть один правильный ответ. Ниже необходимо выбрать правильный ответ.
7. Нажать *Add*. Все теперь вопрос добавлен к тесту. Также можно добавить новые вопросы.
8. Если вы хотите удалить последний добавленный вопрос нажмите *Remove* last.
9. Когда добавили все вопросы нажмите *Done*. Теперь, когда учащиеся будут подключаться к вам через IP адрес, который вы найдете во вкладке *Stats*. Они получат тест и сразу же смогут начать его выполнение.
10. Когда кто-нибудь закончит тест, вы сможете увидеть результат во вкладке *Result*.

**Инструкция по использованию для учащегося**

1. Вы должны запустить программу.
2. Ввести свои данные – имя, фамилия. Также ввести IP адрес, который даст вам преподаватель.
3. Перейти во вкладку *Test* и нажать *Open*, чтобы начать выполнение задания.
4. Открыв тест, вы можете увидеть первый вопрос, чтобы увидеть другие вопросы нажмите *Next*, чтобы вернуться к предыдущему вопросы нажмите *Prev*.
5. Когда закончите тест нажмите *Done*.
6. Чтобы увидеть свой результат откройте вкладку *Result*.

**Примечание:** *Если вы не успеете ответить на все вопросы до того, как истечёт время, тестовое окно закроется и все не отвеченные вопросы будут засчитаны как неправильные.*

**Использованная литература**

1. Герберт Шилдт, Полный справочник по JAVA. Java SE 6 Edition, 2007.
2. Герберт Шилдт, Полное руководство, 2012.
3. H. M. Deitel, Java™ How to Program, Sixth Edition, August 04, 2004.
4. Герберт Шилдт, SWING руководство для начинающих, 2007.
5. Так же были использованы видео уроки в интернете.