Министерство образования и науки Украины

ОДЕССКИЙ НАЦИОНАЛЬНЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Курсовая работа**

По дисциплине

“Объектно-ориентированное программирование”

**Выполнил:**

студент группы АС-071

*Беспечный П. А.*

**Руководитель:**

**Кунгурцев А.Б**

Общая оценка

Дата

Одесса 2009

**Содержание**

1. Задания на разработку
2. Список классов программы
3. Классы программы
4. Инструкция пользователя

Список Литературы

**1 Задания**

**Вариант №1**

Завдання №1.

Учбовий матеріал – елементарні конструкції мови С++.

Треба реалізувати наступні вимоги до програмних класів:1.1, 2.2, 3.2, 4.2, 5.1

**Задания на разработку:**

2. Составные части учебной программы

Учебная программа должна состоять из следующих классов.

Класс представления учебного материала.

Класс обязательно реализует следующие функции.

Чтение учебного материала из файла.

Предоставление студенту порции учебного материала.

Переход к следующей порции материала.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

1. Жесткая, заранее установленная схема подачи материала.

2. Класс контроля знаний студента.

Класс обязательно реализует следующие функции.

Чтение задания к контролю из файла.

Чтение эталонных ответов из файла.

Предоставление студенту задач до контроля знаний.

Оценка каждого ответа.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Задания типа «отметить необходимое».

3. Класс регистрации студента и выдаче итогового документа.

Класс обязательно реализует следующие функции.

Принимает данные студента.

Выдает итоговый документ.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Каждый студент должен пройти регистрацию сам. Он устанавливает себе пароль.

4. Класс введение учебного материала, контрольных заданий, эталонов ответов и другой информации.

Класс обязательно реализует следующие функции.

Введение учебного материала в формате, который задан другими пунктами задания.

Введение контрольных заданий в формате, который задан другими пунктами задания.

Ввод эталонов ответов в формате, который задан другими пунктами задания.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Класс может принимать данные с другого носителя, который не входит в систему.

5. Класс контроля за ходом учебного процесса преподавателем.

Класс обязательно реализует следующие функции.

Просмотр результатов обучения для каждого студента.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Получение данных относительно общего времени обучения студентов

**2 Список классов программы:**

1. Класс входа в систему – авторизация пользователя в системе.
2. Класс регистрации нового студента – по заданию:

Класс регистрации студента и выдаче итогового документа.

Класс обязательно реализует следующие функции.

Принимает данные студента.

Выдает итоговый документ.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Каждый студент должен пройти регистрацию сам. Он устанавливает себе пароль.

1. Класс главного окна программы – реализует:

Класс введение учебного материала, контрольных заданий, эталонов ответов и другой информации.

Класс обязательно реализует следующие функции.

Введение учебного материала в формате, который задан другими пунктами задания.

Введение контрольных заданий в формате, который задан другими пунктами задания.

Ввод эталонов ответов в формате, который задан другими пунктами задания.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Класс может принимать данные с другого носителя, который не входит в систему.

Класс контроля за ходом учебного процесса преподавателем.

Класс обязательно реализует следующие функции.

Просмотр результатов обучения для каждого студента.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

1. Класс проверки знаний пользователя – по заданию класс контроля знаний студента.

Класс обязательно реализует следующие функции.

Чтение задания к контролю из файла.

Чтение эталонных ответов из файла.

Предоставление студенту задач до контроля знаний.

Оценка каждого ответа.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

Задания типа «отметить необходимое».

1. Класс подготовки пользователя – по заданию класс представления учебного материала.

Класс обязательно реализует следующие функции.

Чтение учебного материала из файла.

Предоставление студенту порции учебного материала.

Переход к следующей порции материала.

Класс дополнительно может реализовать следующие функции в соответствии с конкретного задания к курсовой работе.

* 1. Жесткая, заранее установленная схема подачи материала.

Также реализуется пункт 5.1.

**3 Классы программы**

1. Класс входа в систему

Задачей данного класса является проверка логина и пароля пользователя системы, а также предоставление возможности регистрации нового пользователя в системе.

В системе присутствуют два уровня доступа каждый со своими функциями. Класс KLogin проверяет уровни доступа и передает их классу главного окна KMainWin.



Рис. 3.1.1 Алгоритм проверки пары логин и пароль

Исходный код класса:

package KursPackage;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.DataInputStream;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class KLogin extends JFrame implements Consts{

JTextField login;

JPasswordField pass;

JButton enter;

JButton cancel;

JButton reg;

JLabel llogin;

JLabel lpass;

boolean loged;

String ln;

String password;

KLogin () {

super ("Please log in");

JFrame.setDefaultLookAndFeelDecorated(true);

JDialog.setDefaultLookAndFeelDecorated(true);

this.setUndecorated(true);

this.getRootPane().setWindowDecorationStyle(JRootPane.FRAME);

this.setLayout(new FlowLayout ());

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

//--------------------INIT---------------------------------------------

loged = false;

ln = new String ();

password = new String ();

login = new JTextField ();

pass = new JPasswordField ();

llogin = new JLabel ("Логин");

lpass = new JLabel ("Пароль");

enter = new JButton ("Войти");

cancel = new JButton ("Отмена");

reg = new JButton ("Регистрация");

llogin.setPreferredSize(size);

lpass.setPreferredSize(size);

login.setPreferredSize(size);

pass.setPreferredSize(size);

enter.setPreferredSize(size);

cancel.setPreferredSize(size);

reg.setPreferredSize(bigsize);

//-------------------ACTIONLISTENERS-----------------------------------

enter.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

DataInputStream rd = null;

int who = 0;

try {

rd = new DataInputStream(new FileInputStream(Consts.pdatabase));

String buf;

if (rd.available() != 0) {

while (rd.available() != 0) {

buf = rd.readLine();

if (buf.equals(getPass())) {

loged = true;

who = Integer.parseInt(rd.readLine());

getPass();

}

}

if (!loged){

JOptionPane.showMessageDialog(null, "Неправильный логин или пароль");

}

else {

setVisible(false);

new KMainWin (who, ln, password);

}

} else {

JOptionPane.showMessageDialog(null, "Ошибка базы данных");

}

} catch (FileNotFoundException ex) {

Logger.getLogger(KLogin.class.getName()).log(Level.SEVERE, null, ex);

}catch (IOException ex) {

Logger.getLogger(KLogin.class.getName()).log(Level.SEVERE, null, ex);

}

finally {

try {

rd.close();

} catch (IOException ex) {

Logger.getLogger(KLogin.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

cancel.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

reg.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

new KReg();

}

});

//-------------------ADD-----------------------------------------------

this.add(llogin);

this.add(login);

this.add(lpass);

this.add(pass);

this.add(enter);

this.add(cancel);

this.add(reg);

this.setSize(2 \* size.width + 30, size.height \* 4 + 58);

this.setResizable(false);

this.setVisible(true);

}

String getPass(){

ln = login.getText();

password = pass.getText();

return (login.getText() + " " + pass.getText());

}

}

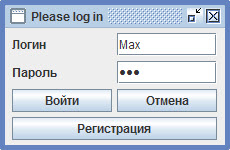


Рис. 3.1.2. Окно входа в систему

1. Класс регистрации нового студента

Задача данного класса регистрация нового пользователя в системе. Такие данные о пользователе как имя, фамилия, группа, ICQ, записываются в файл профиля.



Рис. 3.2.1. Алгоритм регистрации нового профиля

Исходный код класса:

package KursPackage;

import java.awt.FlowLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedWriter;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class KReg extends JFrame implements Consts{

JTextField login;

JPasswordField pass;

JTextField name;

JTextField surname;

JTextField email;

JTextField icq;

JTextField group;

JLabel llogin;

JLabel lpass;

JLabel lname;

JLabel lsurnm;

JLabel lemail;

JLabel licq;

JLabel lgroup;

JButton cancel;

JButton reg;

JButton reset;

boolean flag;

KReg (){

super ("New student");

this.setLayout(new FlowLayout ());

//----------------INIT-------------------------------------------------

login = new JTextField ();

pass = new JPasswordField ();

name = new JTextField ();

surname = new JTextField ();

email = new JTextField ("@");

icq = new JTextField ();

group = new JTextField ();

llogin = new JLabel ("Логин: \*");

lpass = new JLabel ("Пароль: \*");

lname = new JLabel ("Имя: \*");

lsurnm = new JLabel ("Фамилия: \*");

lemail = new JLabel ("Ваш E-Mail: \*");

licq = new JLabel ("Номер ICQ: ");

lgroup = new JLabel ("Группа: \*");

cancel = new JButton ("Отмена");

reg = new JButton ("Регистрация");

reset = new JButton ("Сброс");

login.setPreferredSize(size);

pass.setPreferredSize(size);

name.setPreferredSize(size);

surname.setPreferredSize(size);

email.setPreferredSize(size);

icq.setPreferredSize(size);

group.setPreferredSize(size);

llogin.setPreferredSize(size);

lpass.setPreferredSize(size);

lname.setPreferredSize(size);

lsurnm.setPreferredSize(size);

lemail.setPreferredSize(size);

licq.setPreferredSize(size);

lgroup.setPreferredSize(size);

cancel.setPreferredSize(size);

reg.setPreferredSize(bigsize);

reset.setPreferredSize(size);

flag = false;

//-------------------ACTIONLISTENER------------------------------------

reg.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

try{

FileWriter dtbase = new FileWriter(Consts.pdatabase, true);

dtbase.write(login.getText() + " " + pass.getText() + "\n" + 0 + "\n");

dtbase.close();

BufferedWriter bw = new BufferedWriter(new FileWriter(Consts.pprofiles + login.getText()));

bw.write("Имя : " + name.getText() + "\nФамилия : " + surname.getText() + "\nE-Mail : " + email.getText() + "\nICQ : " + icq.getText() + "\nГруппа : " + group.getText());

flag = true;

bw.close();

dispose ();

} catch (FileNotFoundException ex) {

flag = false;

Logger.getLogger(KReg.class.getName()).log(Level.SEVERE, null, ex);

}

catch (IOException ex) {

flag = false;

Logger.getLogger(KReg.class.getName()).log(Level.SEVERE, null, ex);

}

if (flag){

JOptionPane.showMessageDialog(null, "Регистрация завершена");

}

else {

JOptionPane.showMessageDialog(null, "Ошибка регистрации");

}

}

});

reset.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

login.setText("");

pass.setText("");

name.setText("");

surname.setText("");

email.setText("@");

icq.setText("");

group.setText("");

}

});

cancel.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

dispose ();

}

});

//-------------------ADD-----------------------------------------------

this.add(llogin);

this.add(login);

this.add(lpass);

this.add(pass);

this.add(lname);

this.add(name);

this.add(lsurnm);

this.add(surname);

this.add(lemail);

this.add(email);

this.add(licq);

this.add(icq);

this.add(lgroup);

this.add(group);

this.add(reg);

this.add(reset);

this.add(cancel);

this.setResizable(false);

this.setVisible(true);

this.setSize(2 \* size.width + 30, size.height \* 10 + 60);

}

}

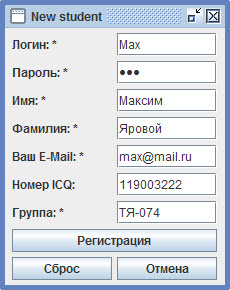


Рис. 3.2.2. Окно регистрации в системе

1. Класс главного окна программы

Задача данного класса – предоставление доступа к остальным функциям системы.

При разных уровнях доступа интерфейс главного окна изменяется, предоставляя разные функции.



Рис. 3.3.1. Алгоритм разделения прав в системе

Исходный код класса:

package KursPackage;

import java.awt.Dimension;

import java.awt.FlowLayout;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

import java.io.\*;

import java.nio.channels.FileChannel;

import java.util.Date;

import javax.swing.border.TitledBorder;

public class KMainWin extends JFrame implements Consts{

KMainWin f;

JButton starttest;

JButton readl;

JButton newt;

JButton newl;

JButton oplog;

JButton exit;

JButton logout;

JButton contr;

JButton time;

File newlec;

File newtest;

JLabel name;

JLabel surnm;

JLabel email;

JLabel icq;

JLabel group;

JPanel info;

JLabel mark;

JPanel button;

KMainWin (int who, final String login, final String password) throws FileNotFoundException, IOException {

super ("Main window");

f = this;

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

this.setLayout(new FlowLayout ());

final BufferedWriter bd = new BufferedWriter(new FileWriter(Consts.log, true));

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " loged in\n");

logout = new JButton ("Log out");

logout.setPreferredSize(size);

exit = new JButton ("Выход");

exit.setPreferredSize(size);

logout.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

try {

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " logged out\n");

bd.close();

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

dispose();

new KLogin();

try {

bd.close();

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

exit.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

try {

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " exited from program\n");

bd.close();

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

System.exit(0);

}

});

if (who != 0){

time = new JButton ("Время обучения");

time.setPreferredSize(bigsize);

newt = new JButton ("Новый тест");

newt.setPreferredSize(bigsize);

newl = new JButton ("Новая лекция");

newl.setPreferredSize(bigsize);

oplog = new JButton ("Открыть лог-файл");

oplog.setPreferredSize(bigsize);

contr = new JButton ("Оценки");

contr.setPreferredSize(bigsize);

//-----------------------------ActionListeners---------------------

time.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

JFrame slog = new JFrame ("Время");

JEditorPane je = new JEditorPane ("rtf;Content-Type = windows-1251", "<html><i><center>Подготовка к тестам</center>");

JScrollPane js = new JScrollPane (je);

slog.add(js);

je.setEditable(false);

BufferedReader rd;

try {

rd = new BufferedReader(new FileReader(Consts.time));

je.read(rd, "rtf;Content-Type = windows-1251");

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

slog.setSize (700, 400);

slog.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

slog.setVisible (true);

slog.setSize(300, 500);

slog.setVisible(true);

}

});

newt.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

JFileChooser fc = new JFileChooser ();

fc.setFileSelectionMode(JFileChooser.FILES\_ONLY);

fc.showDialog(null, "Открыть");

newtest = fc.getSelectedFile();

if (newtest!=null)

{

try {

transfer (curtest, prevtest);

transfer (newtest.getPath(), curtest);

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " added new test\n");

} catch (FileNotFoundException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

JOptionPane.showMessageDialog(null, newlec.getPath());

}

}

});

newl.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

JFileChooser fc = new JFileChooser ();

fc.setFileSelectionMode(JFileChooser.FILES\_ONLY);

fc.showDialog(null, "Открыть");

newlec = fc.getSelectedFile();

if (newlec!=null){

try {

transfer (curlec, prevlec);

transfer (newlec.getPath(), curlec);

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " added new lec\n");

} catch (FileNotFoundException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

JOptionPane.showMessageDialog(null, newlec.getPath());

}

}

});

oplog.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

JFrame slog = new JFrame ("Log");

JEditorPane je = new JEditorPane ("rtf;Content-Type = windows-1251", "<html><i><center>Подготовка к тестам</center>");

JScrollPane js = new JScrollPane (je);

slog.add(js);

je.setEditable(false);

BufferedReader rd;

try {

rd = new BufferedReader(new FileReader(Consts.log));

je.read(rd, "rtf;Content-Type = windows-1251");

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

slog.setSize (700, 400);

slog.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

slog.setVisible (true);

slog.setSize(300, 500);

slog.setVisible(true);

}

});

contr.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

JFrame frame = new JFrame ("Оценки");

JEditorPane je = new JEditorPane ("text/html;Content-Type = windows-1251", "");

JScrollPane js = new JScrollPane (je);

frame.add(js);

je.setEditable(false);

BufferedReader rd;

try {

rd = new BufferedReader(new FileReader(Consts.marks));

je.read(rd, "text/html;Content-Type = windows-1251");

} catch (IOException ey) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ey);

}

frame.setSize (200, 400);

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

frame.setVisible (true);

}

});

button = new JPanel ();

button.setLayout(new FlowLayout());

button.setPreferredSize(new Dimension (210, 180));

button.add (newl);

button.add (newt);

button.add (oplog);

button.add (contr);

button.add(time);

button.add (logout);

button.add (exit);

add(button);

setSize (470, 215);

info = new JPanel();

info.setPreferredSize(new Dimension (230, 170));

}

//who == 0

else {

starttest = new JButton ("Начать тестирование");

starttest.setPreferredSize(bigsize);

readl = new JButton ("Прочитать лекцию");

readl.setPreferredSize(bigsize);

starttest.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

try {

KTestWin n = new KTestWin(login, password, f);

bd.write(new Date().getHours() + ":" + new Date ().getMinutes() + " " + new Date ().getDay() + "/" + new Date ().getMonth() + "/" + new Date ().getYear() + "\n");

bd.write(login + " started test\n");

} catch (FileNotFoundException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

readl.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

try {

new KLec(curlec, "Подготовка к тестам", false, login);

bd.write(new Date().getHours() + ":" + new Date().getMinutes() + " " + new Date().getDay() + "/" + new Date().getMonth() + "/" + new Date().getYear() + " ");

bd.write(login + " began read lec\n");

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

button = new JPanel ();

button.setLayout(new FlowLayout());

button.setPreferredSize(new Dimension (210, 125));

button.add(starttest);

button.add(readl);

button.add(logout);

button.add(exit);

add(button);

setSize (470, 170);

info = new JPanel();

info.setPreferredSize(new Dimension (230, 120));

}

name = new JLabel ();

surnm = new JLabel ();

email = new JLabel ();

icq = new JLabel ();

group = new JLabel ();

mark = new JLabel();

try {

BufferedReader rd = new BufferedReader(new FileReader(Consts.pprofiles + login));

name.setText(rd.readLine());

name.setPreferredSize(size);

surnm.setText(rd.readLine());

surnm.setPreferredSize(size);

email.setText(rd.readLine());

email.setPreferredSize(size);

icq.setText(rd.readLine());

icq.setPreferredSize(size);

group.setText(rd.readLine());

group.setPreferredSize(bigsize);

rd.close ();

BufferedReader rdm = new BufferedReader(new FileReader(Consts.pprofiles + login + ".mrk"));

mark.setText("Оценка: " + rdm.readLine());

rdm.close();

mark.setPreferredSize(size);

}catch (IOException et) {

//BufferedWriter bw = new BufferedWriter(new FileWriter(Consts.pprofiles + login));

//bw.write("Имя : \nФамилия : \nE-Mail : \nICQ : \nГруппа : ");

//bw.close();

}

TitledBorder tb = new TitledBorder ("Info");

info.setLayout(new BoxLayout (info, BoxLayout.Y\_AXIS));

info.add(name);

info.add(surnm);

info.add(email);

info.add(icq);

info.add(group);

info.add(mark);

info.setBorder(tb);

add (info);

this.setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE);

this.setResizable(false);

//this.setIconImage(image);

setVisible (true);

}

boolean transfer (String from, String to){

boolean result = false;

try {

FileChannel ic = new FileInputStream(from).getChannel();

FileChannel oc = new FileOutputStream(to).getChannel();

ic.transferTo(0, ic.size(), oc);

ic.close();

oc.close();

result = true;

} catch (FileNotFoundException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

result = false;

} catch (IOException ex) {

Logger.getLogger(KMainWin.class.getName()).log(Level.SEVERE, null, ex);

result = false;

}

return result;

}

}

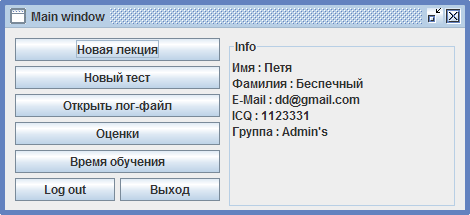


Рис.3.3.2. «Интерфейс для администрирования»

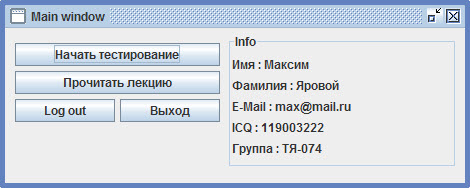


Рис.3.3.3. Интерфейс программы для пользователя

1. Класс проверки знаний пользователя

Задача этого класса – проведение тестирования пользователя, оценка знаний.



Рис.3.4.1. Алгоритм тестирования

Исходный код класса:

package KursPackage;

import java.util.\*;

import javax.swing.\*;

import javax.swing.Timer;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.border.TitledBorder;

public class KTestWin extends JDialog/\*JFrame\*/{

KTestWin g;

int numall;//всего вопросов

int numtrue;//к-во правильных ответов на данной странице

int num;//к-во правильных ответов студента

int time;//время на выполнение теста

int numproi;

Date date;

JLabel lttend;//время до окончания теста

JLabel lrans;//к-во прав. ответов студента

JLabel question;

JButton bnext;

JButton bstop;

JCheckBox cb[] = new JCheckBox[4];

boolean ans[] = new boolean [4];//правильные ответы

JPanel panel;

long megotime;

TitledBorder tv; KTestWin (final String login, String password, KMainWin d) throws FileNotFoundException, IOException{

super(d, "Тест", true);

g = this;

setLayout(new FlowLayout ());

date = new Date();

//узнаем время данное на выполнение теста

BufferedReader tt = new BufferedReader(new

FileReader(Consts.curtesttime));

time = Integer.parseInt(tt.readLine());

tt.close();

//узнали

megotime = date.getTime() + time;

panel = new JPanel ();

panel.setLayout(new FlowLayout ());

//узнаем кво вопросов

BufferedReader test = new BufferedReader(new

FileReader(Consts.curtest));

if (test.ready())

{

while (test.ready()){

if (test.readLine().equals("<q>"))

numall++;

}

}

else {

JOptionPane.showMessageDialog(null, "Не найден файл \" " + Consts.curtest + " \"");

dispose();

}

test.close();

//узнали

//------------------------------init------------------------------------

numproi = 1;

num = 0;

tv = new TitledBorder ("Вопрос");

lrans = new JLabel ("Количество правильных ответов: 0");

lttend = new JLabel ("Время осталось: " + String.valueOf((megotime - date.getTime())/1000) + " sec");

question = new JLabel();

bnext = new JButton ("Далее");

bstop = new JButton ("Стоп");

final BufferedReader tst = new BufferedReader(new

FileReader(Consts.curtest));

add(lttend);

add(lrans);

panel.setLayout(new BoxLayout (panel, BoxLayout.Y\_AXIS));

panel.setPreferredSize(new Dimension (580, 180));

tst.readLine();

question.setText(tst.readLine());

panel.add(question);

for (int i = 0; i<4 ; i++){

cb[i] = new JCheckBox (tst.readLine(), false);

ans[i] = Boolean.parseBoolean(tst.readLine());

panel.add(cb[i]);

}

if (numproi == numall)

{

bnext.hide();

}

if (((megotime - date.getTime())/1000)<0){

bnext.hide();

}

bnext.addActionListener(new ActionListener () {

public void actionPerformed(ActionEvent e) {

numtrue = 0;

for (int i = 0; i < 4; i++){

if (cb[i].isSelected() == ans[i]){

numtrue++;

}

}

if (numtrue == 4){

num++;

}

numproi++;

lrans.setText("Количество правильных ответов: " + num);

lttend.setText("Время осталось: " + String.valueOf((megotime - new Date().getTime())/1000) + " sec");

if (numproi == numall)

{

bnext.hide();

}

try {

tst.readLine();

question.setText(tst.readLine());

for (int i = 0; i<4 ; i++){

cb[i].setLabel(tst.readLine());

cb[i].setSelected(false);

ans[i] = Boolean.parseBoolean(tst.readLine());

}

} catch (IOException ex) {

Logger.getLogger(KTestWin.class.getName()).log(Level.SEVERE, null, ex);

}

if (((megotime - date.getTime())/1000)<0){

bnext.hide();

}

}

});

final ActionListener listener = new ActionListener (){

public void actionPerformed(ActionEvent e) {

numtrue = 0;

for (int i = 0; i < 4; i++){

if (cb[i].isSelected() == ans[i]){

numtrue++;

}

}

if (numtrue == 4){

num++;

}

bstop.hide();

JButton close = new JButton ("Close");

numproi++;

tv.setTitle("Статистика");

lrans.hide();

lttend.hide();

bnext.hide();

question.setText("Количество правильных ответов: " + num);

panel.add(new JLabel("Запись результатов в файл..."));

try{

String buf[] = new String [3];

BufferedWriter mrk = new BufferedWriter(new FileWriter(Consts.pprofiles + login + ".mrk"));

mrk.write(num + "/" + numall);

mrk.close();

BufferedWriter marks = new BufferedWriter(new FileWriter(Consts.marks));

BufferedReader prof = new BufferedReader (new FileReader (Consts.pprofiles + login));

buf[0] = prof.readLine();

buf[1] = prof.readLine();

prof.readLine();

prof.readLine();

buf[2] = prof.readLine();

marks.write(buf[0] + " " + buf[1] + " " + buf[2] + " has " + num + "/" + numall + "\n");

prof.close();

marks.close();

}catch(Exception ey){

System.out.print(ey);

}

panel.add(new JLabel("Запись успешно завершена"));

for (int i = 0; i<4 ; i++)

cb[i].hide();

try {

tst.close();

} catch (IOException ex) {

Logger.getLogger(KTestWin.class.getName()).log(Level.SEVERE, null, ex);

}

close.addActionListener(new ActionListener (){

public void actionPerformed(ActionEvent e) {

dispose();

}

});

g.setSize (600, 255);

add(close);

g.repaint();

}

};

Timer timer = new Timer(time, listener);

timer.setRepeats(false);

timer.start();

bstop.addActionListener(listener);

panel.setBorder(tv);

add(panel);

add(new JLabel ("Всего вопросов : " + numall));

add(bnext);

add(bstop);

setSize (600, 280);

setVisible(true);

}

}

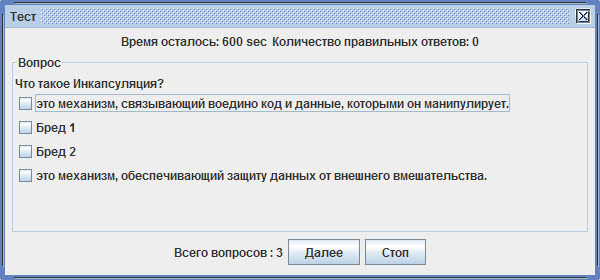


Рис.3.4.2. Начало теста

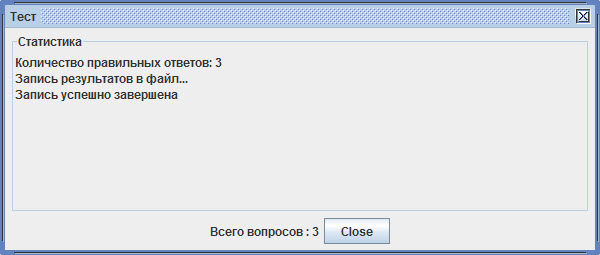


Рис3.4.3. Окончание теста:

1. Класс подготовки пользователя

Задача класса – подготовка пользователя к тестированию.



Рис. 3.5.1. Алгоритм записи времени чтения

Исходный код класса:

package KursPackage;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Date;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class KLec extends JFrame {

JEditorPane je;

JScrollPane js;

long time;

long curtime;

KLec (String lec, String title, boolean iseditable, final String login) throws IOException{

super (title);

je = new JEditorPane ("text/html;Content-Type = windows-1251", "<html><i><center>Подготовка к тестам</center>");

js = new JScrollPane (je);

je.setEditable(iseditable);

if (title.equals("Подготовка к тестам")){

BufferedReader tt = new BufferedReader(new FileReader(Consts.pprofiles + login + ".tme"));

time = Long.parseLong(tt.readLine());

curtime = new Date().getTime();

tt.close();

}

BufferedReader rd = new BufferedReader(new FileReader(lec));

je.read(rd, "text/html;Content-Type = windows-1251");

rd.close();

setSize (700, 400);

add(js);

this.addWindowListener(new WindowListener (){

public void windowClosing(WindowEvent e) {

BufferedWriter tm = null;

try {

BufferedWriter ctt;

ctt = new BufferedWriter(new FileWriter(Consts.pprofiles + login + ".tme"));

curtime = new Date().getTime() - curtime;

time += curtime;

ctt.write(time + "");

ctt.close();

String[] buf = new String[3];

tm = new BufferedWriter(new FileWriter(Consts.time));

BufferedReader prof = new BufferedReader(new FileReader(Consts.pprofiles + login));

buf[0] = prof.readLine();

buf[1] = prof.readLine();

prof.readLine();

prof.readLine();

buf[2] = prof.readLine();

tm.write("общее время обучения " + buf[0] + " " + buf[1] + " " + buf[2] + " : " + time + "\n");

prof.close();

tm.close();

dispose();

} catch (IOException ex) {

Logger.getLogger(KLec.class.getName()).log(Level.SEVERE, null, ex);

} finally {

try {

tm.close();

} catch (IOException ex) {

Logger.getLogger(KLec.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

public void windowClosed(WindowEvent e) {}

public void windowIconified(WindowEvent e) {}

public void windowDeiconified(WindowEvent e) {}

public void windowActivated(WindowEvent e) {}

public void windowDeactivated(WindowEvent e) {}

public void windowOpened(WindowEvent e) {}

});

setVisible (true);

}

}

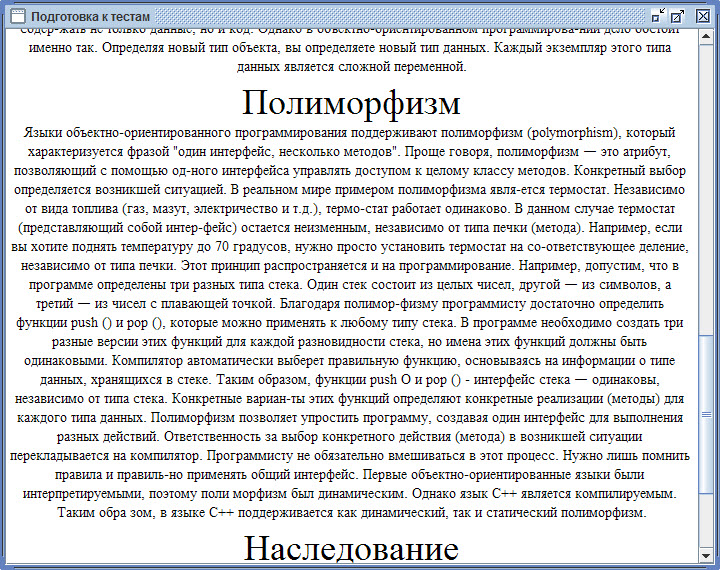


Рис.3.5.2. Интерфейс окна чтения лекции

1. Интерфейс констант

Интерфейс содержит константы.

package KursPackage;

import java.awt.Dimension;

import java.awt.Image;

import java.awt.Toolkit;

public interface Consts {

public String pdatabase = new String ("d:\\database.dat");//путь к базе данных

public String pprofiles = new String ("d:\\profiles\\");//путь к профайлам

public String prevlec = new String ("d:\\lec\\prev.dat");//путь к пред. лекции

public String curlec = new String ("d:\\lec\\cur.dat");//путь к текущей лекции

public String curtest = new String ("d:\\tests\\cur.dat");//путь к текущим тестам

public String prevtest = new String ("d:\\tests\\prev.dat");//путь к пред. тестам

public String curtesttime = new String ("d:\\tests\\cur.conf");//путь к файлу времени пров. тек. теста

public String prevtesttime = new String ("d:\\tests\\prev.conf");//путь к файлу времени пров. пред. теста

public String marks = new String ("d:\\marks.dat");//путь к файлу оценок

public String log = new String ("d:\\log.log");//путь к логу

public String time = new String ("d:\\time.dat");//путь к логу времени

public Dimension size = new Dimension (100, 23);

public Dimension bigsize = new Dimension(size.width \* 2 + 5, size.height);

public Image image = Toolkit.getDefaultToolkit().getImage("d:\\icon.jpg");

}

**4 Инструкция пользователя**

После запуска программы появляется окно (рис. 4.1) где нужно ввести логин и пароль и нажать на кнопку «Войти».

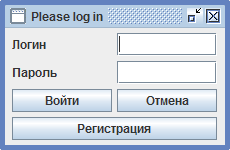


Рис.4.1. Окно входа в систему

Если Вы еще не зарегистрированы в системе, то Вам нужно пройти регистрацию. Чтобы зарегистрироваться необходимо в окне входа в систему (рис.4.1) нажать на кнопку «Регистрация». После этого появится окно регистрации нового пользователя системы (рис. 4.2). В этом окне Вам будет предложено ввести регистрационные данные. После заполнения всех полей нажмите на кнопку «Регистрация».

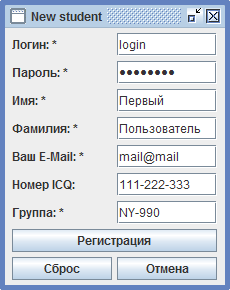


Рис.4.2. Регистрация нового пользователя

Если регистрация в системе прошла успешно, то появится уведомление об успешном окончании регистрации (рис.4.3).

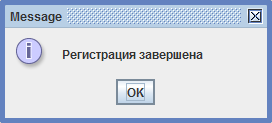


Рис.4.3. Успешное окончание регистрации

После регистрации Вы можете войти в систему(рис.4.4.).

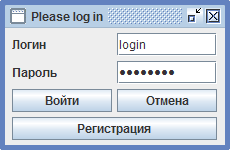


Рис.4.4. Окно входа в систему под логином login и паролем password.

После нажатия на кнопку «Войти» Вы войдете в систему как login с правами пользователя. После входа в систему появится главное окно программы (рис. 4.5.).

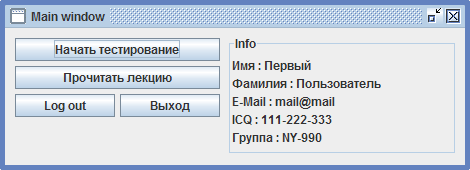


Рис. 4.5. Главное окно системы пользователя

При нажатии на кнопку «Начать тестирование» появится окно тестирования (рис.4.6)где Вам будет дан вопрос и варианты ответа. На выполение теста будет дано определенное преподавателем.

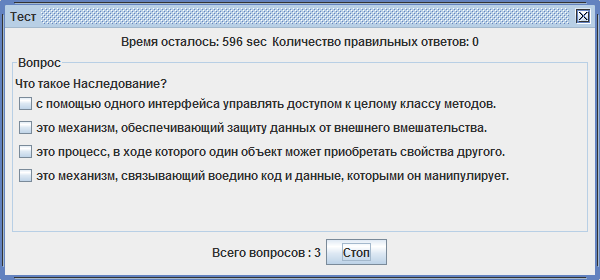


Рис.4.6. Окно тестирования пользователя

После окончания теста Вам будет выдана оценка(рис.4.7.)

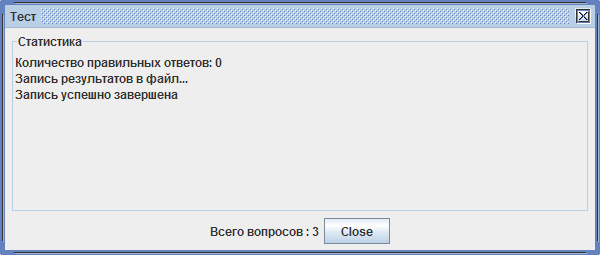


Рис.4.7. Статистика прохождения тестов.

Также в главном окне(рис.4.5) можно получить доступ к лекциям с помощью кнопки «Прочитать лекцию». После нажатия на эту кнопку появится окно (рис.4.8) с лекцией.

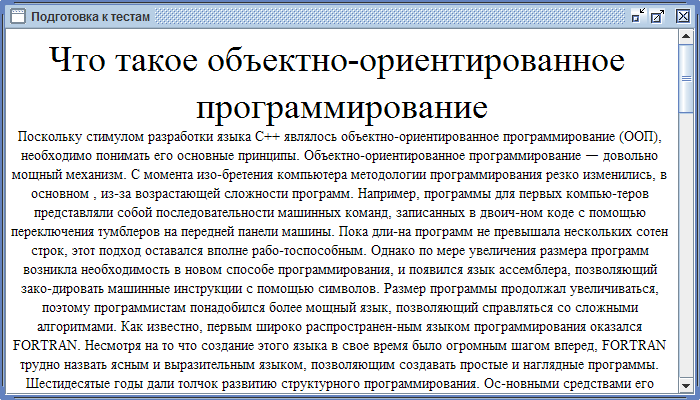


Рис.4.8. «Подготовка к тестам»

При авторизации в системе с правами администратора или с правами преподавателя появляется окно со следующим интерфейсом (рис.4.9.).

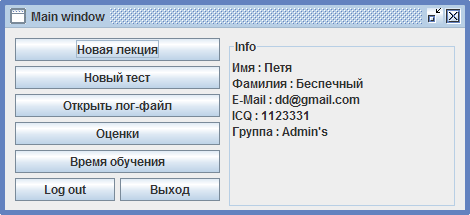


Рис.4.9. Главное окно системы пользователя с правами администратора

При нажатии на кнопку «Новая лекция» появится окно выбора файла лекции(рис.4.10.)

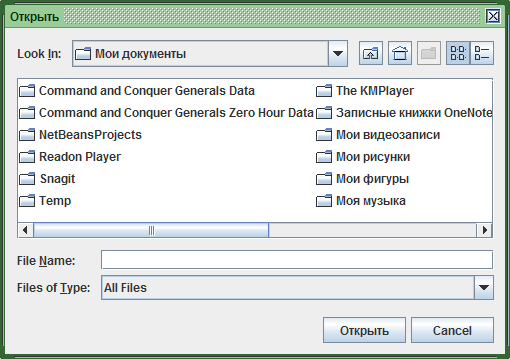


Рис. 4.10. Выбор файла лекции

При нажатии на кнопку «Новый тест» появляется окно выбора файла (рис.4.11)

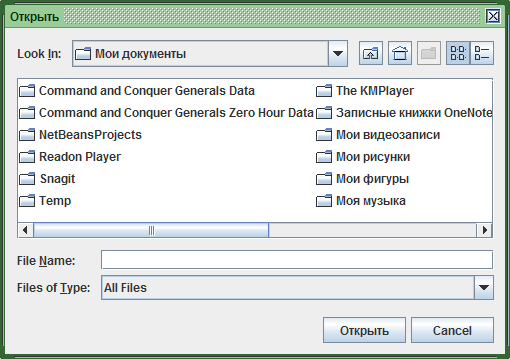


Рис.4.11. Окно выбора файла теста.

При нажатии на кнопку «Открыть лог-файл» откроется окно просмотра лог-файла(рис.4.12.)

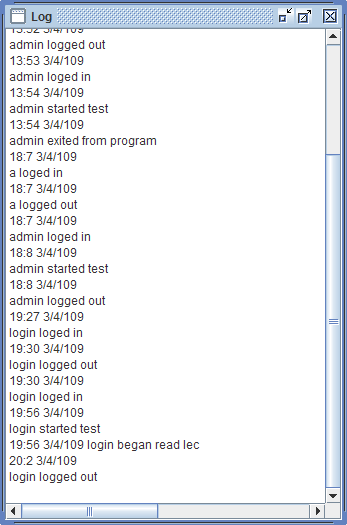


Рис.4.12. Просмотр лог-файла.

При нажатии на кнопку «Оценки» появляется окно с результатами тестирования пользователей(рис.4.13.)

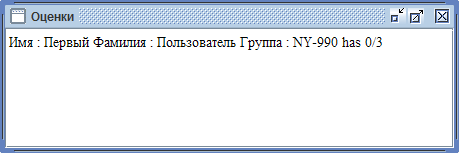


Рис.4.13. Окно просмотра результатов тестирования

При нажатии на кнопку «Время обучения» появляется окно просмотра общего времени обучения пользователей в миллисекундах (рис.4.14.).

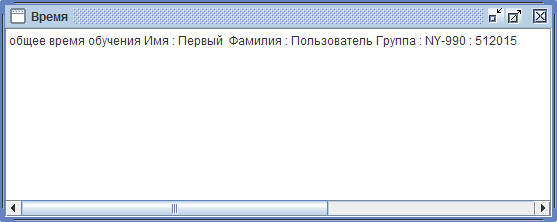


Рис.4.14. Окно просмотра общего времени обучения пользователя.

При нажатии на кнопку «Выход» программа завершает свою работу.

**Список литературы**

1.Давыдов В.И. Кунгурцев А.Б. Объектно-ориентированное программирование. Одесса: Автограф, 2003. - 124с.

2.Кунгурцев О.Б. Основы программирования на языке Java.Середовище Net Beans. Одесса. : ВМВ, 2006. -182с.

2.Грады Буч. Объектно-ориентированный анализ и проектирование с программа приложений на С + +. СПБ: БИНОМ, 2000. 560 с.

3.Дейтел Х. М., Дейтел П. Дж. БСантары С.И. Технология программирования на Java 2. Кн. 1. Как программировать на С + +. М.: БИНОМ, 2003. 560 с.

4.Грегоры К. Использование Visual C + + 6. М-СПБ: Дом «Вильямс», 2001. 864 с.

5.Дейтел Х. М., Дейтел П. Дж. Как программировать на С + +. М.: БИНОМ, 2001. 1152с.