МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра автоматизированных систем управления

Отчет

по курсовой работе

по программированию

**Система тестирования знаний**

Выполнил Проверил

Студент группы АВТ-913 Ассистент кафедры АСУ

Волков Богдан Попов Е.А.

г. Новосибирск

2020

**Оглавление**

1. Постановка задачи………………………………………………………….3
2. Описание классов…………………………………………………………..4
3. Взаимосвязь классов……………………………………………………….6
4. Описание разработанного приложения…………………………………...7
5. Заключение………………………………………………………………..10
6. Список использованной литературы…………………………………….10
7. Приложение. Листинг разработанной программы……………………...11

**Постановка задачи**

Разработать программу системы тестирования знаний. Элементами системы являются модели авторизации, списка вопросов, прохождения тестирования, результаты тестирования.

Пользователь проходит авторизацию и тестирование, где составитель создал вопросы (выводятся в случайном порядке) для его прохождения, в итоге человек получает ответ на сколько вопросов он правильно ответил, по окончанию тестирования.

Программа должна составлять вопросы и варианты ответов, которые должны браться из файла, который делает составитель тестирования.

Всё вышеперечисленное программа вычисляет с учётом количества вопросов и правильных ответов.

При запуске программа должна запрашивать следующие параметры для инициализации работы системы:

1. Логин и пароль для авторизации.
2. Количество вопросов и вариантов ответа.
3. Количество верных ответов.
4. Максимальное количество вопросов.
5. Результат

На выходе программа должна отображать рекомендуемые показатели тестирования знаний.

При реализации программы использовать концепции ООП парадигмы: инкапсуляцию, наследование и полиморфизм.

Использовать язык программирования C# и Windows forms.

**Описание классов**

**authorization**

Первичный конструктор инициализирует поля данных объекта данными, введенными пользователем:

1. Логин.

2. Пароль.

Содержит поля данных: login, password, log, pas.

Публичные методы form2.Show(), Hide() используются для показа второй формы и скрытия первой формы.

Приватная функция button1\_click\_1(object sender, EventArgs e) используется для проверки корректности ввода данных и вывода второй формы, в которой происходит тестирование.

Приватная функция button2\_Сlick(object sender, EventArgs e) используется для вывода информации о программе и действий для корректной работы.

**testing**

Класс имитирует сам процесс тестирования:

1. Вывод вопросов.

2. Вывод варианта ответов.

3. Подсчёт количества вопросов.

4. Подсчёт верных ответов.

5. Завершение тестирования.

Содержит поля данных: allqlist, currentqlist, currentquestionnumber, correctanswers.

Приватная функция Form2\_Load(object sender, EventArgs e) используется для загрузки txt файла с вопросами для тестирования.

Публичная функция ParseQuestionsFromList(IList<string> stringlist) используется для загрузки txt файла с вопросами для тестирования.

Публичная функция NextQuestion() используется для вывода вопросов, вариантов ответа, проверки правильности ответа, подсчёта верных ответов, подсчёта полного количества вопросов, вывод итога тестирования.

Публичная функция Restart() используется для повторного прохождения тестирования.

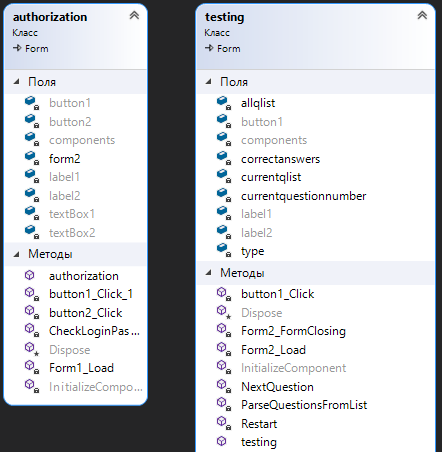
Приватная функция button1\_Click(object sender, EventArgs e) используется для вывода следующего вопроса.

**BaseQuestion**

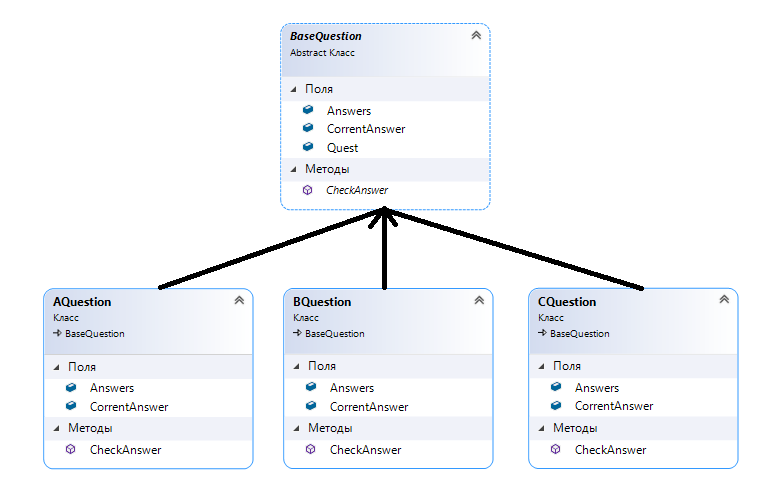
Класс выводит вопрос, варианты ответа и запоминает верный ответ.

Содержит поля данных: Quest, Answers, CorrentAnswer.

**Взаимосвязь классов**

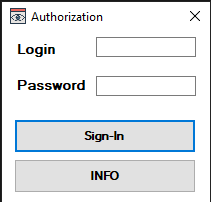


*Рисунок 1. Элементы программы*

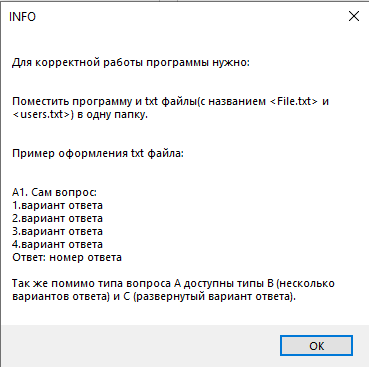


*Рисунок 2. Реализация интерфейса BaseQuestion*

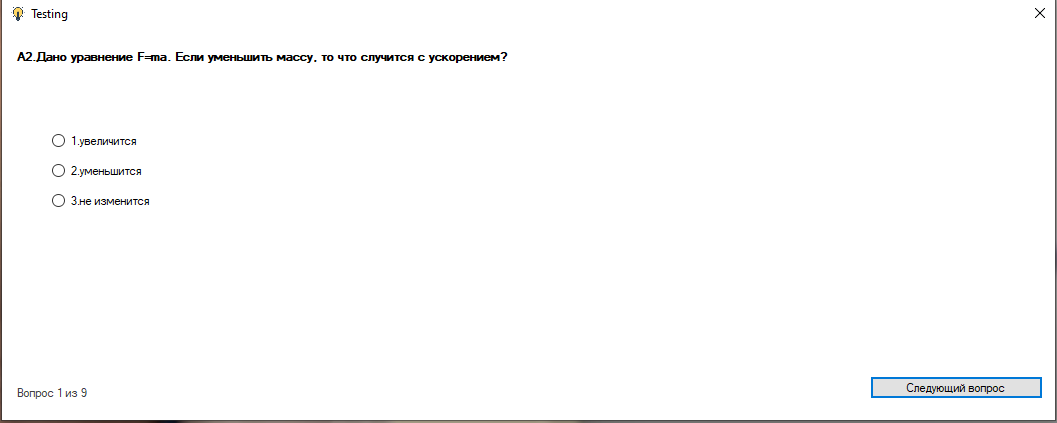
**Описание разработанного приложения**

**

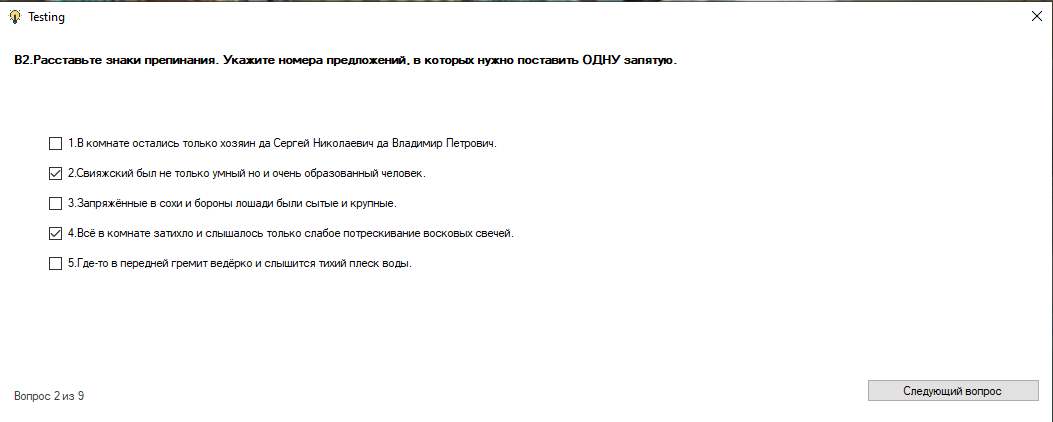
*Рисунок 3. Окно авторизации.*

**

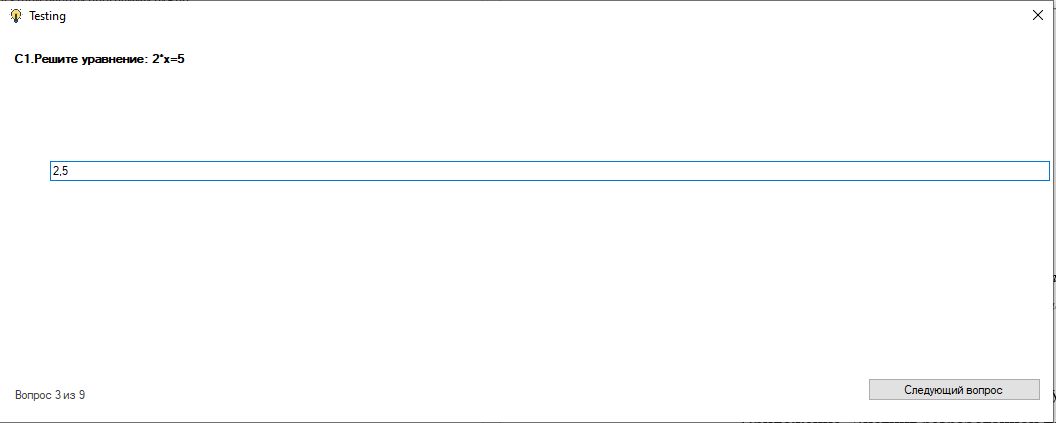
*Рисунок 4. Окно, которое открывается при нажатии кнопки «INFO».*

**

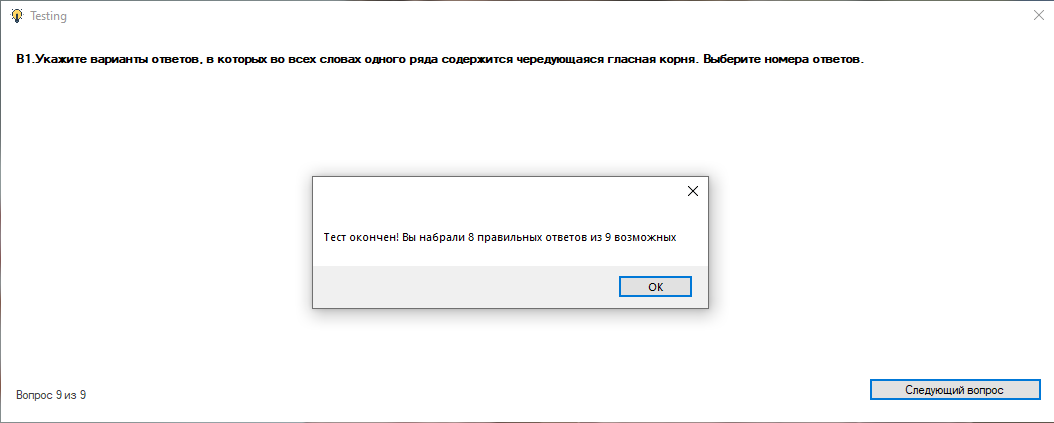
*Рисунок 5. Окно, которое открывается при нажатии кнопки «Sign-In». Появляется форма с тестированием. Выбирается вариант ответа и нажимается кнопка «Следующий вопрос».*

**

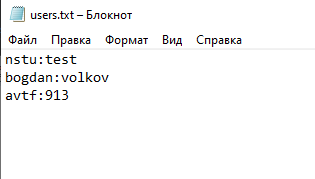
*Рисунок 6. Окно, которое открывается при нажатии кнопки «Следующий вопрос». Выбирается несколько вариантов ответа и нажимается кнопка «Следующий вопрос»*

**

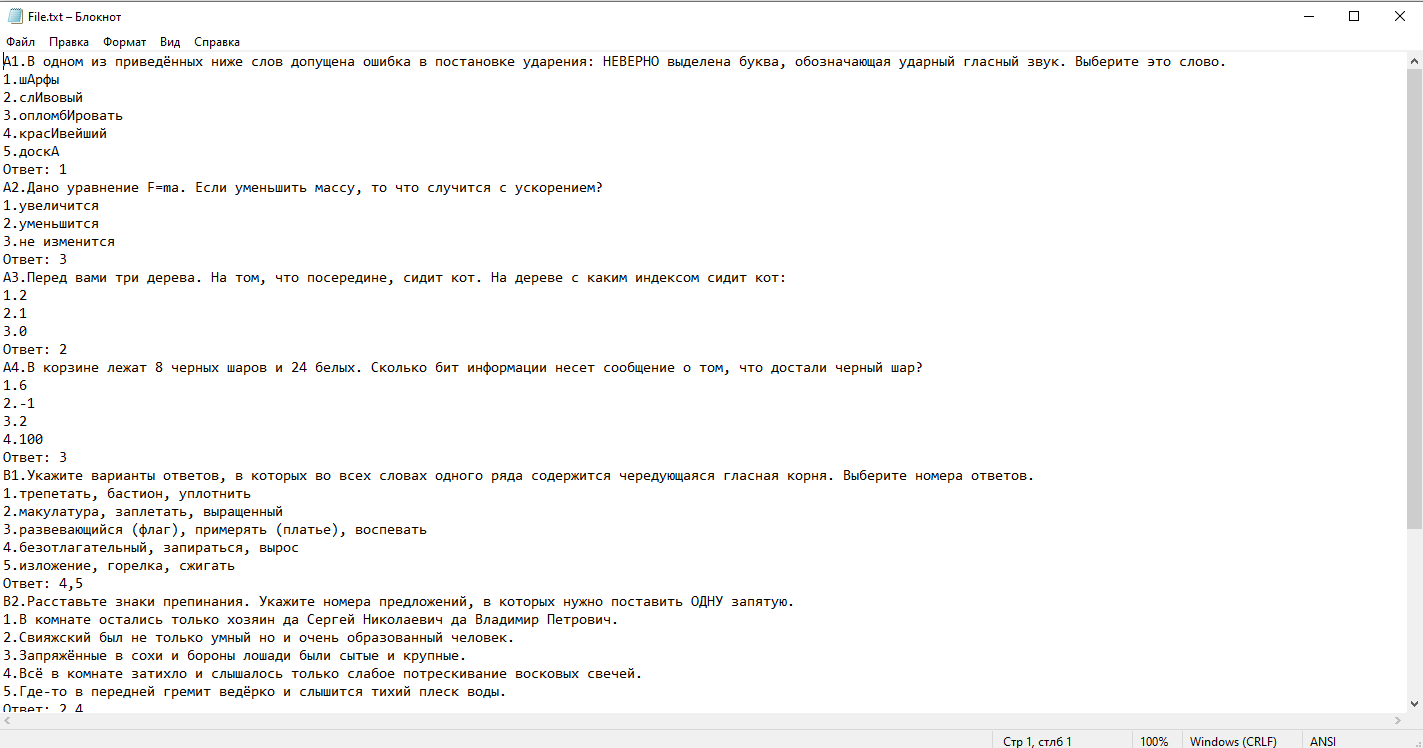
*Рисунок 7. Окно, которое открывается при нажатии кнопки «Следующий вопрос». Вводится свой вариант ответа и нажимается кнопка «Следующий вопрос».*

**

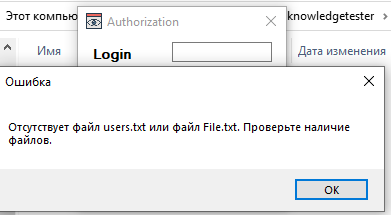
*Рисунок 8. Окно, которое открывается при нажатии кнопки «Следующий вопрос». После ответа на последний вопрос выводится окно с результатом. При нажатии на кнопку «ОК» программа завершит свою работу.*

**

*Рисунок 9. Из этого файла с расширением <txt> берется список пользователей для авторизации.*

**

*Рисунок 10. Из этого файла с расширением <txt> берутся вопросы для прохождения тестирования.*

**

*Рисунок 11. Программа показывает ошибку из-за отсутствия одного из файлов.*

**Заключение**

Программа, разработанная в процессе курсовой работы, представляет собой инструмент для создания и прохождения тестирования.

В ходе разработки программы были использованы архитектурные приемы, раскрывающие реальный потенциал парадигмы объектно-ориентированного программирования, без которых она не имеет смысла. Они позволяют упростить многие моменты разработки, добавляют возможность масштабирования элементов системы и облегчают сопровождение в будущем. Все поставленные цели из постановки задачи успешно достигнуты.

Однако, в процессе работы возникло некоторое количество идей, которые не были реализованы в итоговом проекте в силу недостатка знаний, что оставляет задел на будущее и мотивирует продолжить изучение объектно-ориентированного программирования как дисциплины.

**Список использованной литературы**

1. Язык программирования C# 7 и платформы .NET и .NET Core / Джепикс Филипп, Троелсен Эндрю, 2018 – 1328 c.
3. Современные технологии программирования: разработка Windows-приложений на языке С# / С.В. Горелов, 2019 – 363 с.
4. Курс «Программирование» <https://dispace.edu.nstu.ru/didesk/course/show/8261>

/ Д.Н. Достовалов, Н.В. Баранова, О.В. Лауферман, Е.А. Попов,

И.В. Эстрайх, 2020.

4. Сайт «Решу ЕГЭ» <https://ege.sdamgia.ru> / Д.Д. Гущин, 2011-2020

**Приложение. Листинг разработанной программы**

Файл **«authorization.cs»**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.IO;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace knowledgetester

{

public partial class authorization : Form

{

private static bool CheckLoginPassword(string login, string password)

{

string line;

using (var sr = new StreamReader("users.txt"))

{

while ((line = sr.ReadLine()) != null)

{

if (line == $"{login}:{password}") return true;

}

}

return false;

}

public authorization()

{

InitializeComponent();

form2 = new testing(); // привязка первой формы ко второй, чтобы они видели друг друга

}

testing form2; //привязали форму2 к первой форме

private void button1\_Click\_1(object sender, EventArgs e)

{

if (File.Exists("users.txt") && File.Exists("File.txt"))

{

string Log = textBox1.Text; // что пользователь ввёл в login

string Pas = textBox2.Text; // что пользователь ввёл в password

if (CheckLoginPassword(Log, Pas)) //проверка правильности ввода данных

{

form2.Show(); //Показ второй формы(testing)

Hide(); //Скрытие текущей формы(authorization)

}

else

{

MessageBox.Show("Не удаётся войти.\nПожалуйста, проверьте правильность написания логина и пароля.", "Ошибка.");

}

}

else

{

MessageBox.Show("Отсутствует файл users.txt или файл File.txt. Проверьте наличие файлов.", "Ошибка");

Environment.Exit(0);

}

}

private void button2\_Click(object sender, EventArgs e)

{

MessageBox.Show("Для корректной работы программы нужно:\n\n\nПоместить программу и " +

"txt файлы(с названием <File.txt> и <users.txt>) в одну папку.\n\n\nПример оформления txt файла:\n\n\nА1. " +

"Сам вопрос:\n1.вариант ответа\n2.вариант ответа\n3.вариант ответа\n4.вариант ответа\n" +

"Ответ: номер ответа\n\nТак же помимо типа вопроса A доступны типы B (несколько вариантов ответа) и С (развернутый вариант ответа).", "INFO");

}

private void Form1\_Load(object sender, EventArgs e)

{

}

}

}

Файл **«testing.cs»**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Text.RegularExpressions;

namespace knowledgetester

{

public partial class testing : Form

{

abstract class BaseQuestion

{

public string Quest;

public abstract bool CheckAnswer(string answer);

public object Answers;

public object CorrentAnswer;

}

class AQuestion : BaseQuestion

{

public new List<string> Answers = new List<string>();

public new int CorrentAnswer;

public override bool CheckAnswer(string answer)

{

return Convert.ToInt32(answer) == this.CorrentAnswer;

}

}

class BQuestion : BaseQuestion

{

public new List<string> Answers = new List<string>();

public new List<int> CorrentAnswer;

public override bool CheckAnswer(string answer)

{

string pattern = @"(\d+)";

var answers = new List<int>();

foreach (Match match in Regex.Matches(answer, pattern))

{

answers.Add(Convert.ToInt32(match.Value));

}

answers.Sort();

if (answers.Count != CorrentAnswer.Count)

{

return false;

}

for (int i = 0; i < answers.Count; i++)

{

if (answers[i] != CorrentAnswer[i])

{

return false;

}

}

return true;

}

}

class CQuestion : BaseQuestion

{

public new object Answers = null;

public new string CorrentAnswer;

public override bool CheckAnswer(string answer)

{

return answer == this.CorrentAnswer;

}

}

List<BaseQuestion> allqlist = new List<BaseQuestion>();

List<BaseQuestion> currentqlist = new List<BaseQuestion>();

int currentquestionnumber = 0;

int correctanswers = 0;

int type = 0;

public testing()

{

InitializeComponent();

}

private void Form2\_FormClosing(object sender, FormClosingEventArgs e)

{

Environment.Exit(0); //завершение программы

}

private void Form2\_Load(object sender, EventArgs e)

{

ParseQuestionsFromList(File.ReadAllLines("File.txt", Encoding.GetEncoding("windows-1251")));

Restart();

}

void ParseQuestionsFromList(IList<string> stringlist)

{

int n = 1;

for (int i = 0; i < stringlist.Count; i++)

{

n = 1;

char letter = stringlist[i][0];

switch (letter)

{

case 'A':

var aq = new AQuestion()

{

Quest = stringlist[i]

};

while (!stringlist[i + n].Contains("Ответ:"))

{

aq.Answers.Add(stringlist[i + n]);

n++;

}

aq.CorrentAnswer = int.Parse(stringlist[i + n].Split(new char[1] { ':' }, StringSplitOptions.RemoveEmptyEntries)[1]);

n++;

allqlist.Add(aq);

break;

case 'B':

var bq = new BQuestion()

{

Quest = stringlist[i]

};

while (!stringlist[i + n].Contains("Ответ:"))

{

bq.Answers.Add(stringlist[i + n]);

n++;

}

string pattern = @"(\d+)";

var answers = new List<int>();

foreach (Match match in Regex.Matches(stringlist[i + n], pattern))

{

answers.Add(Convert.ToInt32(match.Value));

}

answers.Sort();

bq.CorrentAnswer = answers;

n++;

allqlist.Add(bq);

break;

case 'C':

CQuestion cq = new CQuestion()

{

Quest = stringlist[i]

};

cq.CorrentAnswer = stringlist[i + n].Split(new char[1] { ':' }, StringSplitOptions.RemoveEmptyEntries)[1];

cq.CorrentAnswer = cq.CorrentAnswer.Trim();

allqlist.Add(cq);

break;

default:

continue;

}

}

}

void NextQuestion()

{

switch (type)

{

case 0:

break;

case 1:

foreach (RadioButton rbb in this.Controls.OfType<RadioButton>())

{

if (rbb.Checked)

{

string pattern = @"(\d+)";

var match = Regex.Match(rbb.Text, pattern, RegexOptions.Singleline);

if (currentqlist[currentquestionnumber - 1].CheckAnswer(match.Value))

{

correctanswers++;

}

}

}

for (int i = 0; i < this.Controls.Count; i++)

{

if (this.Controls[i].GetType() == typeof(RadioButton))

{

this.Controls[i].Dispose(); i--;

}

}

break;

case 2:

string answer = "";

foreach (CheckBox rbb in this.Controls.OfType<CheckBox>())

{

if (rbb.Checked)

{

string pattern = @"(\d+)";

var match = Regex.Match(rbb.Text, pattern, RegexOptions.Singleline);

answer += match.Value + ", ";

}

}

if (currentquestionnumber != 0 && currentqlist[currentquestionnumber - 1].CheckAnswer(answer))

{

correctanswers++;

}

for (int i = 0; i < this.Controls.Count; i++)

{

if (this.Controls[i].GetType() == typeof(CheckBox))

{

this.Controls[i].Dispose(); i--;

}

}

break;

case 3:

foreach (TextBox rbb in this.Controls.OfType<TextBox>())

{

if (rbb.TextLength != 0)

{

if (currentqlist[currentquestionnumber - 1].CheckAnswer(rbb.Text))

{

correctanswers++;

}

}

}

for (int i = 0; i < this.Controls.Count; i++)

{

if (this.Controls[i].GetType() == typeof(TextBox))

{

this.Controls[i].Dispose(); i--;

}

}

break;

}

if (currentquestionnumber >= currentqlist.Count)

{

MessageBox.Show(String.Format("Тест окончен! Вы набрали {0} правильных ответов из {1} возможных", correctanswers, currentqlist.Count));

Restart();

Environment.Exit(0); //завершение программы

return;

}

label2.Text = String.Format("Вопрос {0} из {1}", currentquestionnumber + 1, currentqlist.Count);

var q = currentqlist[currentquestionnumber];

if (q.GetType() == typeof(AQuestion))

{

type = 1;

AQuestion aq = q as AQuestion;

label1.Text = aq.Quest;

Random r = new Random();

for (int i = 0; i < aq.Answers.Count; i++)

{

var rb = new RadioButton()

{

Text = aq.Answers[i],

Location = new Point(50, 100 + i \* 30),

Width = 1000

};

rb.CheckedChanged += (s, ee) =>

{

if (rb.Checked)

{

foreach (RadioButton rbb in this.Controls.OfType<RadioButton>())

{

if (s != rbb) rbb.Checked = false;

}

}

};

this.Controls.Add(rb);

}

}

else if (q.GetType() == typeof(BQuestion))

{

type = 2;

BQuestion bq = q as BQuestion;

label1.Text = bq.Quest;

Random r = new Random();

for (int i = 0; i < bq.Answers.Count; i++)

{

var rb = new CheckBox()

{

Text = bq.Answers[i],

Location = new Point(50, 100 + i \* 30),

Width = 1000

};

this.Controls.Add(rb);

}

}

else

{

type = 3;

CQuestion cq = q as CQuestion;

label1.Text = cq.Quest;

var inp = new TextBox()

{

Location = new Point(50, 130),

Width = 1000

};

this.Controls.Add(inp);

}

this.Invalidate();

currentquestionnumber++;

}

void Restart()

{

Random r = new Random();

currentqlist = allqlist.OrderBy(x => r.Next()).Take(20).ToList(); //случайный порядок вопросов

//currentqlist = allqlist; //порядок вопросов как в текстовом файле

currentquestionnumber = 0;

correctanswers = 0;

NextQuestion();

}

private void button1\_Click(object sender, EventArgs e)

{

NextQuestion();

}

}

}