Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

“Российский экономический университет имени Г.В. Плеханова”

МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ

Специальность 09.02.07 Информационные системы и программирование

Квалификация: Программист

Отчет по практике

ПМ 01. «Разработка модулей программного обеспечения для компьютерных систем»

УП 01.01 «Прикладное программирование»

Листов: 36.

|  |  |
| --- | --- |
| Выполнил:  студент группы П50-1-18  Дрюпин Андрей Александрович | Проверил:  преподаватель  Горбунов Антон Дмитриевич |

2020 г.

Содержание.

[Практическая работа №1. 3](#_Toc45011079)

[Игра «Крестики-нолики» 3](#_Toc45011080)

[Практическая работа №2. 7](#_Toc45011081)

[Шлюз авторизации 7](#_Toc45011082)

[Практическая работа №3. 14](#_Toc45011083)

[Текстовый редактор 14](#_Toc45011084)

[Практическая работа №4. 20](#_Toc45011085)

[Веб-браузер 20](#_Toc45011086)

[Практическая работа №5. 27](#_Toc45011087)

[Калькулятор 27](#_Toc45011088)

[Практическая работа №6. 34](#_Toc45011089)

[Почтовый клиент 34](#_Toc45011090)

# Практическая работа №1.

## Игра «Крестики-нолики»

Постановка задачи: Ознакомиться со структурой WinForms. Создать программу «Крестики и нолики». С информирующими всплывающими окнами (MessageBox) о победе X или O и ничья. Должна присутствовать возможность начать игру заново без перезапуска приложения. Научиться использовать элементы Button и TableLayoutPanel, обрабатывать событие Click.

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
| 1 | Form1.cs | Класс, который содержит методы Form1, NextStep, WinCheck, NewGame. Отвечает за обработку действий пользователя, проверку на выигрыш и ведение счета. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

int XScore = 0;

int OScore = 0;

int all = 0;

bool OrXO = true;

Button[] buttons;

public Form1()

{

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.Fixed3D; //Хмм... А зачем нам менять размер окна?

InitializeComponent();

buttons = new Button[]

{

button1, button2, button3, button4, button5, button6, button7, button8, button9

};

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Click += NextStep;

}

}

private void NextStep(object sender, EventArgs e)

{

all++;

if (OrXO)

{

(sender as Button).Text = "X";

OrXO = false;

label1.Text = "Ход O";

}

else

{

(sender as Button).Text = "O";

OrXO = true;

label1.Text = "Ход X";

}

(sender as Button).Enabled = false;

WinCheck();

}

private void WinCheck()

{

if(((String)button1.Text == "X" && (String)button4.Text == "X" && (String)button7.Text == "X") || ((String)button2.Text == "X" && (String)button5.Text == "X" && (String)button8.Text == "X") || ((String)button3.Text == "X" && (String)button6.Text == "X" && (String)button9.Text == "X") || ((String)button1.Text == "X" && (String)button2.Text == "X" && (String)button3.Text == "X") || ((String)button4.Text == "X" && (String)button5.Text == "X" && (String)button6.Text == "X") || ((String)button7.Text == "X" && (String)button8.Text == "X" && (String)button9.Text == "X") || ((String)button1.Text == "X" && (String)button5.Text == "X" && (String)button9.Text == "X") || ((String)button3.Text == "X" && (String)button5.Text == "X" && (String)button7.Text == "X"))

{

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Enabled = false;

}

label1.Text = "Нажмите на кнопку 'Новая игра'";

MessageBox.Show("Крестики выиграли!");

all = 0;

XScore++;

label5.Text = XScore.ToString();

}

else if(((String)button1.Text == "O" && (String)button4.Text == "O" && (String)button7.Text == "O") || ((String)button2.Text == "O" && (String)button5.Text == "O" && (String)button8.Text == "O") || ((String)button3.Text == "O" && (String)button6.Text == "O" && (String)button9.Text == "O") || ((String)button1.Text == "O" && (String)button2.Text == "O" && (String)button3.Text == "O") || ((String)button4.Text == "O" && (String)button5.Text == "O" && (String)button6.Text == "O") || ((String)button7.Text == "O" && (String)button8.Text == "O" && (String)button9.Text == "O") || ((String)button1.Text == "O" && (String)button5.Text == "O" && (String)button9.Text == "O") || ((String)button3.Text == "O" && (String)button5.Text == "O" && (String)button7.Text == "O"))

{

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Enabled = false;

}

label1.Text = "Нажмите на кнопку 'Новая игра'";

MessageBox.Show("Нолики выиграли!");

all = 0;

OScore++;

label6.Text = OScore.ToString();

}

else if(all == 9)

{

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Enabled = false;

}

label1.Text = "Нажмите на кнопку 'Новая игра'";

MessageBox.Show("Ничья");

all = 0;

}

}

private void NewGame(object sender, EventArgs e)

{

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Text = "";

buttons[i].Enabled = true;

}

OrXO = true;

label1.Text = "Ход X";

all = 0;

}

}

Результат работы программы:

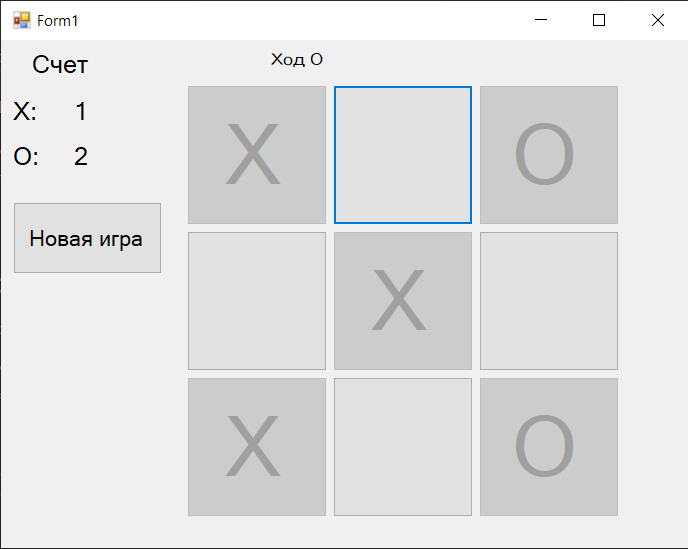


Рисунок 1 - Результат работы программы.

Описание: После запуска приложения нажмите на кнопку “Новая игра” чтобы начать новую игру. Так же вы можете использовать эту кнопку для перезапуска уже начатой партии. Игра рассчитана на двух игроков, которые ходят по очереди, нажимая на одну из девяти игровых плиток. Слева отображается счет, а сверху подсказки. Игра оптимизирована под портативные устройства с сенсорным управлением.

Вывод: выполнив данную работу, я научился работать с TableLayoutPanel и Button и создал простейшую, но весьма занимательную игру.

# Практическая работа №2.

## Шлюз авторизации

Постановка задачи: разработать окно авторизации и регистрации. При регистрации предусмотреть наличие следующих полей: логин, пароль, повторение пароля, серия и номер паспорта, номер телефона и электронная почта. Если пользователь не заполнил или заполнил неправильно необходимо выдать соответствующее сообщение ему. Данные должны хранится в файле.

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
| 1 | Form1.cs | Класс, который содержит методы Form1, RegBtn, LogBtn. Отвечает за форму авторизации: обрабатывает вводимые пользователем данные и производит проверку на соответствие с записями в файле. |
| 2 | Reg.cs | Класс, который содержит методы Form1, Register. Отвечает за форму регистрации: обрабатывает вводимые пользователем данные, проводит проверку их актуальности, а так же добавляет новые записи в файл. |
| 3 | INIManager | Класс, который содержит методы GetPrivateString, WritePrivateString. Отвечает за функцию чтения/записи в INI-файл. |
| 4 | Change.cs | Класс, который содержит методы ChangePass\_Click, Change. Отвечает за форму восстановления пароля: обрабатывает вводимые пользователем данные, проводит проверку их актуальности, а так же заменяет данные в уже имеющемся файле. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void RegBtn(object sender, EventArgs e)

{

Reg frm = new Reg();

frm.Show();

}

private void LogBtn(object sender, EventArgs e)

{

INIManager manager = new INIManager(Directory.GetCurrentDirectory() + "\\Users.ini");

string Login = textBox1.Text;

string Password = textBox2.Text;

if (Login.Length <= 0 || Password.Length <= 0) MessageBox.Show("Ошибка! Заполнены не все поля");

else if (Regex.IsMatch(Password, @"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[^a-zA-Z0-9])\S{1,16}$", RegexOptions.IgnoreCase)) {

if (manager.GetPrivateString(Login, "Password").Length != -1 && manager.GetPrivateString(Login, "Password") == Password) MessageBox.Show("Привет," + manager.GetPrivateString(Login, "Name") +" "+ manager.GetPrivateString(Login, "Surname"));

else MessageBox.Show("Ошибка! Неправильный логин или пароль");

}

else MessageBox.Show("Ошибка! Пароль не соответствует условиям");

}

}

### Reg.cs

public partial class Reg : Form

{

public Reg()

{

InitializeComponent();

}

private void Register(object sender, EventArgs e)

{

int rightOK = 0;

String Login = textBox1.Text;

String Password = textBox2.Text;

String RePassword = textBox3.Text;

String Name = textBox4.Text;

String Passport = textBox5.Text;

String Email = textBox7.Text;

String Phone = textBox6.Text;

if (Login.Length <= 0 || Passport.Length <= 0 || Name.Length <= 0 || Password.Length <= 0 || RePassword.Length <= 0 || Email.Length <= 0 || Phone.Length <= 0) MessageBox.Show("Ошибка! Заполнены не все поля");

else

{

if (Password != RePassword) MessageBox.Show("Ошибка! Пароли не совпадают");

else rightOK++;

if (!Regex.IsMatch(Password, @"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[^a-zA-Z0-9])\S{1,16}$", RegexOptions.IgnoreCase)) MessageBox.Show("Ошибка! Пароль не соответствует условиям");

else rightOK++;

if (!Regex.IsMatch(Phone, @"^((\\+7-?)|8)?[0-9]{10}$", RegexOptions.IgnoreCase)) MessageBox.Show("Ошибка! Телефон не соответствует условиям");

else rightOK++;

if (Passport.Length != 10) MessageBox.Show("Ошибка! Серия и номер паспорта не соответствует условиям");

else rightOK++;

if (!Regex.IsMatch(Email, @"^\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$", RegexOptions.IgnoreCase)) MessageBox.Show("Ошибка! Почта не соответствует условиям");

else rightOK++;

}

if (rightOK == 5)

{

INIManager manager = new INIManager(Directory.GetCurrentDirectory() + "\\Users.ini");

if (manager.GetPrivateString(Login, Password).Length != -1) {

manager.WritePrivateString(Login, "Password", Password);

manager.WritePrivateString(Login, "Name", Name);

manager.WritePrivateString(Login, "Passport", Passport);

manager.WritePrivateString(Login, "Email", Email);

manager.WritePrivateString(Login, "Phone", Phone);

Close();

}

else MessageBox.Show("Ошибка! Такой пользователь уже существует");

}

}

}

### INIManager

public class INIManager

{

//Конструктор, принимающий путь к INI-файлу

public INIManager(string aPath)

{

path = aPath;

}

//Конструктор без аргументов (путь к INI-файлу нужно будет задать отдельно)

public INIManager() : this("") { }

//Возвращает значение из INI-файла (по указанным секции и ключу)

public string GetPrivateString(string aSection, string aKey)

{

//Для получения значения

StringBuilder buffer = new StringBuilder(SIZE);

//Получить значение в buffer

GetPrivateString(aSection, aKey, null, buffer, SIZE, path);

//Вернуть полученное значение

return buffer.ToString();

}

//Пишет значение в INI-файл (по указанным секции и ключу)

public void WritePrivateString(string aSection, string aKey, string aValue)

{

//Записать значение в INI-файл

WritePrivateString(aSection, aKey, aValue, path);

}

//Возвращает или устанавливает путь к INI файлу

public string Path { get { return path; } set { path = value; } }

//Поля класса

private const int SIZE = 1024; //Максимальный размер (для чтения значения из файла)

private string path = null; //Для хранения пути к INI-файлу

//Импорт функции GetPrivateProfileString (для чтения значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "GetPrivateProfileString")]

private static extern int GetPrivateString(string section, string key, string def, StringBuilder buffer, int size, string path);

//Импорт функции WritePrivateProfileString (для записи значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "WritePrivateProfileString")]

private static extern int WritePrivateString(string section, string key, string str, string path);

}

### Change.cs

public partial class Change : Form

{

public Change()

{

InitializeComponent();

}

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

{

int rightOK = 0;

String Login = textBox1.Text;

String Password = textBox4.Text;

String RePassword = textBox5.Text;

String Name = textBox2.Text;

String Phone = textBox3.Text;

if (Login.Length <= 0 || Name.Length <= 0 || Password.Length <= 0 || RePassword.Length <= 0 || Phone.Length <= 0) MessageBox.Show("Ошибка! Заполнены не все поля");

else

{

if (Password != RePassword) MessageBox.Show("Ошибка! Пароли не совпадают");

else rightOK++;

if (!Regex.IsMatch(Password, @"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[^a-zA-Z0-9])\S{1,16}$", RegexOptions.IgnoreCase)) MessageBox.Show("Ошибка! Пароль не соответствует условиям");

else rightOK++;

if (!Regex.IsMatch(Phone, @"^((\\+7-?)|8)?[0-9]{10}$", RegexOptions.IgnoreCase)) MessageBox.Show("Ошибка! Телефон не соответствует условиям");

else rightOK++;

}

if (rightOK == 3)

{

INIManager manager = new INIManager(Directory.GetCurrentDirectory() + "\\Users.ini");

if (manager.GetPrivateString(Login, "Name") == Name && manager.GetPrivateString(Login, "Phone") == Phone)

{

manager.WritePrivateString(Login, "Password", Password);

Close();

}

else MessageBox.Show("Ошибка! Данные введены неправильно. Пароль не был изменен.");

}

}

}

Результат работы программы:

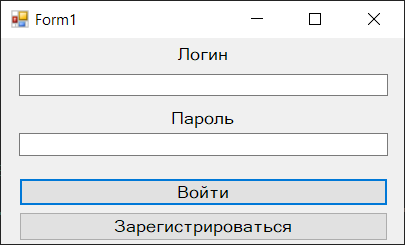


Рисунок 2 – Форма входа.

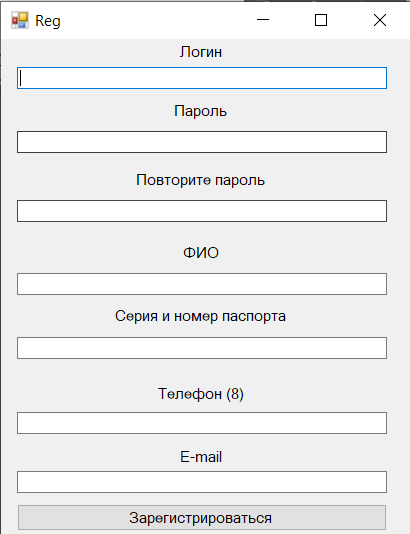


Рисунок 2 – Форма регистрации.

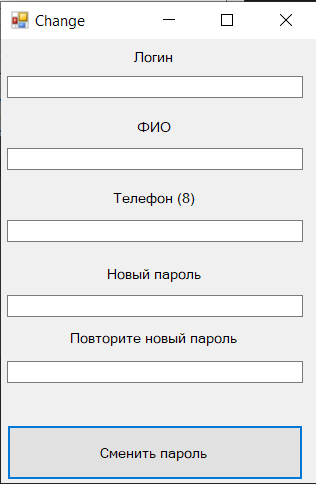


Рисунок 3 – Форма восстановления пароля.

Описание: Для создания нового пользователя нажмите на кнопку “Зарегистрироваться” в форме авторизации. В открывшемся окне заполните все поля согласно подсказкам программы и нажмите на кнопку “Зарегистрироваться”, после чего форма регистрации закроется.  
Для авторизации введите свои учетные данные в форме авторизации и нажмите на кнопку “Войти”, после чего, в случае успешной авторизации, произойдет переход в защищенную область программы.   
Если вы забыли пароль, то нажмите на кнопку “Сменить пароль” в форме авторизации. В открывшемся окне заполните все поля согласно подсказкам программы и нажмите на кнопку “Сменить пароль”, после чего форма восстановления закроется и ваш пароль будет сменен.

Вывод: выполнив данную работу, я научился работать с переходами между форм, INI-файлами и производить проверку вводимых пользователем данных.

# Практическая работа №3.

## Текстовый редактор

Постановка задачи: разработать программу текстовый редактор. Предусмотреть наличие следующего функционала: Открытие файла, сохранения файла, форматирование выделенного текста (шрифт, размер, цвет), выделение всего текста, копирование, вставка, удаление выделенного текста, выравнивание абзаца (по левому краю, по правому краю, по середине), изменение межстрочного интервала абзаца, создание списка(маркированного). Предусмотреть наличие контекстного меню (по правой кнопки мыши) с следующими функциями: копировать, вставить, выделить всё, удалить. Формат файла: RTF (Rich Text File).

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
|  | Form1.cs | Класс, который содержит методы Form1, OpenToolStripMenuItem\_Click, SaveToolStripMenuItem\_Click, ExitToolStripMenuItem\_Click, AboutToolStripMenuItem\_Click, ExitToolStripMenuItem\_Click, TextPlace\_TextChanged, Saving, ClosingApp, CopyToolStripMenuItem\_Click, PasteToolStripMenuItem\_Click, CutToolStripMenuItem\_Click, SelectAllToolStripMenuItem\_Click, RemoveToolStripMenuItem\_Click, Bold\_Click, CursiveBtn\_Click, LineBtn\_Click, FontStyleBtn\_Click, comboBox1\_SelectedIndexChanged, comboBox2\_SelectedValueChanged, ListBtn\_Click, TextPlace\_KeyDown. Отвечает за работу с файлами, текстами, шрифтами и стилями текста. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

bool saved = true;

bool Bold = true;

bool Cursive = true;

bool Line = true;

public Form1()

{

InitializeComponent();

fontDialog1.ShowColor = true;

TextPlace.Font = new Font("Calibri", 12);

TextPlace.ContextMenuStrip = contextMenuStrip1;

FontStyleBtn.Text = TextPlace.Font.Name + " " + TextPlace.Font.Size;

}

private void OpenToolStripMenuItem\_Click(object sender, EventArgs e) //Открываем файл

{

if (openFileDialog1.ShowDialog() == DialogResult.Cancel)

return;

// получаем выбранный файл

string filename = openFileDialog1.FileName;

// читаем файл в строку

string fileText = System.IO.File.ReadAllText(filename);

TextPlace.Rtf = fileText;

}

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

{

Saving(false);

} //Обработка сохранения

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

{

ClosingApp();

} //Обрабатываем выход из диалога

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

{

MessageBox.Show("Приложение создано Андреем Дрюпиным :)", "Что вы хотели здесь увидеть?");

} //Я. Просто Я.

private void ExitToolStripMenuItem\_Click(object sender, FormClosingEventArgs e)

{

ClosingApp();

} //Обрабатываем закрытие окна

private void TextPlace\_TextChanged(object sender, EventArgs e)

{

saved = false;

} //Обрабатываем изменение текста

private void Saving(bool exit)

{

if (saveFileDialog1.ShowDialog() == DialogResult.Cancel)

return;

// получаем выбранный файл

string filename = saveFileDialog1.FileName;

System.IO.File.WriteAllText(filename, TextPlace.Rtf);

saved = true;

if (exit) Application.Exit();

} //Сохранение

private void ClosingApp()

{

if (!saved)

{

var result = MessageBox.Show("Вы хотите сохранить изменения в файле?", "Предупреждение", MessageBoxButtons.YesNoCancel);

if (result == DialogResult.Yes) Saving(true);

if (result == DialogResult.No) Application.Exit();

}

else Application.Exit();

} //Закрываем если сохранили

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

{

SendKeys.Send("^c");

}

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

{

SendKeys.Send("^v");

}

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

{

SendKeys.Send("^x");

}

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

{

SendKeys.Send("^a");

}

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

{

SendKeys.Send("{DELETE}");

}

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

{

if(Bold)

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Bold);

(sender as Button).ForeColor = Color.White;

(sender as Button).BackColor = Color.Blue;

Bold = false;

}

else

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Regular);

(sender as Button).ForeColor = default;

(sender as Button).BackColor = default;

Bold = true;

}

}

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

{

if (Cursive)

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Italic);

(sender as Button).ForeColor = Color.White;

(sender as Button).BackColor = Color.Blue;

Cursive = false;

}

else

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Regular);

(sender as Button).ForeColor = default;

(sender as Button).BackColor = default;

Cursive = true;

}

}

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

{

if (Line)

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Underline);

(sender as Button).ForeColor = Color.White;

(sender as Button).BackColor = Color.Blue;

Line = false;

}

else

{

TextPlace.SelectionFont = new Font(TextPlace.Font, FontStyle.Regular);

(sender as Button).ForeColor = default;

(sender as Button).BackColor = default;

Line = true;

}

}

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

{

if (fontDialog1.ShowDialog() == DialogResult.Cancel)

return;

// установка шрифта

TextPlace.SelectionFont = fontDialog1.Font;

TextPlace.SelectionColor = fontDialog1.Color;

FontStyleBtn.Text = fontDialog1.Font.Name + " " + fontDialog1.Font.Size;

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

switch((sender as ComboBox).SelectedIndex)

{

case 0:

TextPlace.SelectionAlignment = HorizontalAlignment.Left;

break;

case 1:

TextPlace.SelectionAlignment = HorizontalAlignment.Center;

break;

case 2:

TextPlace.SelectionAlignment = HorizontalAlignment.Right;

break;

default:

TextPlace.SelectionAlignment = HorizontalAlignment.Left;

break;

}

} //Устанавливаем выравнивание

private void comboBox2\_SelectedValueChanged(object sender, EventArgs e)

{

TextPlace.SelectionCharOffset = Convert.ToInt32((sender as ComboBox).Text);

} //Устанавливаем межстрочный интервал

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

{

string temptext = TextPlace.SelectedText;

int SelectionStart = TextPlace.SelectionStart;

int SelectionLength = TextPlace.SelectionLength;

TextPlace.SelectionStart = TextPlace.GetFirstCharIndexOfCurrentLine();

TextPlace.SelectionLength = 0;

TextPlace.SelectedText = "1. ";

int j = 2;

for (int i = SelectionStart; i < SelectionStart + SelectionLength; i++)

if (TextPlace.Text[i] == '\n')

{

TextPlace.SelectionStart = i + 1;

TextPlace.SelectionLength = 0;

TextPlace.SelectedText = j.ToString() + ". ";

j++;

SelectionLength += 3;

}

} //Создаем список по клику

private void TextPlace\_KeyDown(object sender, KeyEventArgs e)

{

int tempNum;

if (e.KeyCode == Keys.Enter)

try

{

if (char.IsDigit(TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine()]))

{

if (char.IsDigit(TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine() + 1]) && TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine() + 2] == '.')

tempNum = int.Parse(TextPlace.Text.Substring(TextPlace.GetFirstCharIndexOfCurrentLine(), 2));

else tempNum = int.Parse(TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine()].ToString());

if (TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine() + 1] == '.' || (char.IsDigit(TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine() + 1]) && TextPlace.Text[TextPlace.GetFirstCharIndexOfCurrentLine() + 2] == '.'))

{

tempNum++;

TextPlace.SelectedText = "\r\n" + tempNum.ToString() + ". ";

e.SuppressKeyPress = true;

}

}

}

catch { }

} //Продолжаем список по нажатию на Enter

}

}

Результат работы программы:

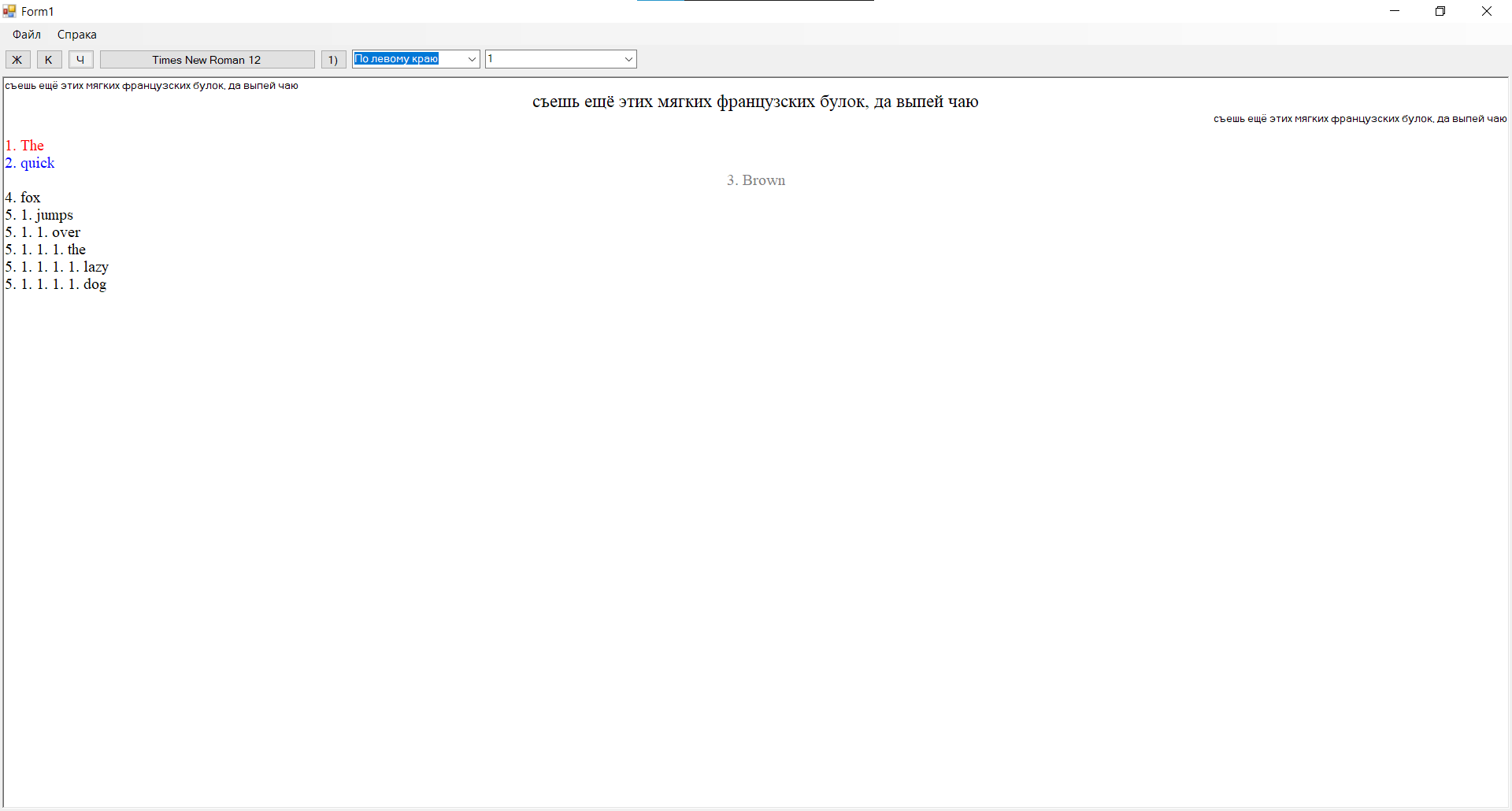


Рисунок 1 - Результат работы программы.

Описание: Основное поле предназначено для текста.  
Для работы с файлами нажмите на соответствующую кнопку в меню «Файл».  
Для получения помощи и справки нажмите на соответствующую кнопку в меню «Справка». Для изменения стиля, размера и цвета шрифта нажмите на кнопку с выбранным в настоящий момент шрифтом. Для быстрого изменения начертания текста воспользуйтесь тремя кнопками изменения шрифта, расположенными на главной панели. Для создания нумерованного списка нажмите на кнопку «1)  
». Список будет автоматически продолжен после нажатия на клавишу «Enter». Для выключения автоматического продолжения списка удалите последний номер в списке. Для изменения горизонтального положения текста выберите соответствующий пункт из выпадающего списка «Положение текста». Для изменения межстрочного интервала выберите соответствующий пункт из выпадающего списка «Межстрочный интервал» или введите его вручную.

Вывод: Я научился работать с текстовыми файлами и элементом «RichTextBox» и создал текстовый редактор.

# Практическая работа №4.

## Веб-браузер

Постановка задачи: разработать программу "Браузер". Необходимо предусмотреть: вкладочное меню, открытие введённых пользователем ссылок, переход на предыдущую страницу, переход на следующую страницу, обновление страницы, переход на домашнюю страницу, хранение истории посещенных сайтов.

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
| 1 | Form1.cs | Класс, который содержит методы Form1, tabControl1\_Selected, WebDocumentNavigate, client\_DownloadFileCompleted, WebDocumentTitleChanged, tabControl1\_DoubleClick, ReturnBtnClick, FowardBtnClick, RefreshBtnClick, GoToBtn\_Click, DownloadsBtn\_Click, HomeBtn\_Click, HistoryBtn\_Click. Отвечает за основную логику приложения. Создает и закрывает вкладки, переходит по адресам, и т. д. |
| 2 | Downloads.cs | Класс, который содержит метод Downloads. Отвечает за считывание из файла загрузок и их демонстрацию пользователю. |
| 3 | History.cs | Класс, который содержит метод History. Отвечает за считывание из файла истории и их демонстрацию пользователю. |
| 4 | INIManager | Класс, который содержит методы GetPrivateString, WritePrivateString. Отвечает за функцию чтения/записи в INI-файл. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

Dictionary<int, WebBrowser> pages = new Dictionary<int, WebBrowser>(); //Хранилище вкладок

INIManager DownloadManager = new INIManager(Directory.GetCurrentDirectory() + "\\Downloads.ini"); //Загрузки

INIManager HistoryManager = new INIManager(Directory.GetCurrentDirectory() + "\\History.ini"); //История

String URL = "http://google.com"; //Адрес по умолчанию

int DownloadsCount = 0; //id загрузки

int HistoryCount = 0; //id истории

public Form1()

{

InitializeComponent();

while (DownloadManager.GetPrivateString(DownloadsCount.ToString(), "URL").Length != 0)

{

DownloadsCount++;

}

while (HistoryManager.GetPrivateString(HistoryCount.ToString(), "URL").Length != 0)

{

HistoryCount++;

}

}

private void tabControl1\_Selected(object sender, TabControlCancelEventArgs e)

{

if (e.TabPageIndex == tabControl1.TabPages.Count - 1)

{

TabPage tabpage = new TabPage("New tab");

WebBrowser webcontrol = new WebBrowser();

webcontrol.Dock = System.Windows.Forms.DockStyle.Fill;

webcontrol.Navigate(URL); //ПоМЕТОЧка

webcontrol.Navigating += WebDocumentNavigate;

webcontrol.DocumentTitleChanged += WebDocumentTitleChanged;

pages.Add(e.TabPageIndex, webcontrol);

tabpage.Controls.Add(webcontrol);

tabControl1.TabPages.Add(tabpage);

try

{

tabControl1.TabPages.Remove(tabControl1.SelectedTab);

}

catch { }

tabControl1.TabPages.Add(new TabPage("+"));

tabControl1.SelectTab(e.TabPageIndex);

//ShowIterator<int, WebBrowser>(pages); //отладка

}

try

{

URLBox.Text = pages[tabControl1.SelectedIndex].Url.AbsoluteUri;

}

catch { }

}

private void WebDocumentNavigate(object sender, WebBrowserNavigatingEventArgs e)

{

HistoryManager.WritePrivateString(HistoryCount.ToString(), "URL", e.Url.AbsoluteUri);

HistoryManager.WritePrivateString(HistoryCount.ToString(), "DATE", DateTime.Now.ToString());

HistoryCount++;

if (e.Url.Segments[e.Url.Segments.Length - 1].EndsWith(".pdf") || e.Url.Segments[e.Url.Segments.Length - 1].EndsWith(".png") || e.Url.Segments[e.Url.Segments.Length - 1].EndsWith(".jpeg") || e.Url.Segments[e.Url.Segments.Length - 1].EndsWith(".jpg"))

{

e.Cancel = true;

string filepath = null;

saveFileDialog1.FileName = e.Url.Segments[e.Url.Segments.Length - 1];

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

filepath = saveFileDialog1.FileName;

WebClient client = new WebClient();

client.DownloadFileCompleted += new AsyncCompletedEventHandler(client\_DownloadFileCompleted);

client.DownloadFileAsync(e.Url, filepath);

DownloadManager.WritePrivateString(DownloadsCount.ToString(), "URL", e.Url.AbsoluteUri);

DownloadManager.WritePrivateString(DownloadsCount.ToString(), "PATH", filepath);

DownloadManager.WritePrivateString(DownloadsCount.ToString(), "DATE", DateTime.Now.ToString());

DownloadsCount++;

}

}

} //Обрабатываем сохранение

private void client\_DownloadFileCompleted(object sender, AsyncCompletedEventArgs e)

{

MessageBox.Show("File downloaded");

}

private void WebDocumentTitleChanged(object sender, EventArgs e)

{

tabControl1.SelectedTab.Text = (sender as WebBrowser).DocumentTitle; //меняет заголовок активной

}

private void tabControl1\_DoubleClick(object sender, EventArgs e)

{

pages.Remove(tabControl1.SelectedIndex);

//ShowIterator<int, WebBrowser>(pages); //отладка

//tabControl1.SelectedTab.Parent = null;

Dictionary<int, WebBrowser> temppages = new Dictionary<int, WebBrowser>();

int t = 0;

for (int i = 0; i < tabControl1.TabPages.Count; i++)

{

try

{

temppages.Add(t, pages[i]);

t++;

}

catch {}

}

pages.Clear();

pages = temppages;

//ShowIterator<int, WebBrowser>(pages); //отладка

//ShowIterator<int, WebBrowser>(temppages); //отладка

tabControl1.TabPages.Remove(tabControl1.SelectedTab);

} //удаление вкладки двойным кликом. Не Работает удаление

private void ReturnBtnClick(object sender, EventArgs e)

{

try

{

pages[tabControl1.SelectedIndex].GoBack();

}

catch { }

} //назад

private void FowardBtnClick(object sender, EventArgs e)

{

try

{

pages[tabControl1.SelectedIndex].GoForward();

}

catch { }

} //вперед

private void RefreshBtnClick(object sender, EventArgs e)

{

try

{ pages[tabControl1.SelectedIndex].Refresh(); }

catch { }

} //обновить

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

{

String TempURL = URLBox.Text;

Uri uriResult;

if (TempURL.Length == 0) pages[tabControl1.SelectedIndex].Navigate(TempURL);

else if (Uri.TryCreate(TempURL, UriKind.Absolute, out uriResult) && (uriResult.Scheme == Uri.UriSchemeHttp || uriResult.Scheme == Uri.UriSchemeHttps))

try

{

pages[tabControl1.SelectedIndex].Navigate(URLBox.Text);

}

catch { }

else

try

{

pages[tabControl1.SelectedIndex].Navigate("https://www.google.com/search?newwindow=1&q=" + TempURL);

}

catch { }

} //переход по URL

void ShowIterator<K, V>(Dictionary<K, V> myList)

{

if (myList == null)

return;

string s = "";

foreach (KeyValuePair<K, V> kvp in myList)

s += string.Format("Key = {0}, Value = {1}",

kvp.Key, kvp.Value) + Environment.NewLine;

MessageBox.Show(s);

} //отладка Dictionary

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

{

Downloads dwn = new Downloads();

dwn.Show();

} //Открываем Загрузки

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

{

try

{

pages[tabControl1.SelectedIndex].Navigate(URL);

}

catch { }

}

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

{

History hst = new History();

hst.Show();

}

}

### Downloads.cs

public partial class Downloads : Form

{

INIManager DownloadManager = new INIManager(Directory.GetCurrentDirectory() + "\\Downloads.ini");

int DownloadsCount = 0;

public Downloads()

{

InitializeComponent();

while (DownloadManager.GetPrivateString(DownloadsCount.ToString(), "URL").Length != 0)

{

Console.WriteLine(DownloadsCount);

ListViewItem item = new ListViewItem(DownloadManager.GetPrivateString(DownloadsCount.ToString(), "PATH"));

item.SubItems.Add(DownloadManager.GetPrivateString(DownloadsCount.ToString(), "URL"));

item.SubItems.Add(DownloadManager.GetPrivateString(DownloadsCount.ToString(), "DATE"));

listView2.Items.Add(item);

DownloadsCount++;

}

}

}

### History.cs

public partial class History : Form

{

INIManager HistoryManager = new INIManager(Directory.GetCurrentDirectory() + "\\History.ini");

int HistoryCount = 0;

public History()

{

InitializeComponent();

while (HistoryManager.GetPrivateString(HistoryCount.ToString(), "URL").Length != 0)

{

Console.WriteLine(HistoryManager);

ListViewItem item = new ListViewItem(HistoryManager.GetPrivateString(HistoryCount.ToString(), "URL"));

item.SubItems.Add(HistoryManager.GetPrivateString(HistoryCount.ToString(), "DATE"));

listView1.Items.Add(item);

HistoryCount++;

}

}

}

### INIManager

public class INIManager

{

//Конструктор, принимающий путь к INI-файлу

public INIManager(string aPath)

{

path = aPath;

}

//Конструктор без аргументов (путь к INI-файлу нужно будет задать отдельно)

public INIManager() : this("") { }

//Возвращает значение из INI-файла (по указанным секции и ключу)

public string GetPrivateString(string aSection, string aKey)

{

//Для получения значения

StringBuilder buffer = new StringBuilder(SIZE);

//Получить значение в buffer

GetPrivateString(aSection, aKey, null, buffer, SIZE, path);

//Вернуть полученное значение

return buffer.ToString();

}

//Пишет значение в INI-файл (по указанным секции и ключу)

public void WritePrivateString(string aSection, string aKey, string aValue)

{

//Записать значение в INI-файл

WritePrivateString(aSection, aKey, aValue, path);

}

//Возвращает или устанавливает путь к INI файлу

public string Path { get { return path; } set { path = value; } }

//Поля класса

private const int SIZE = 1024; //Максимальный размер (для чтения значения из файла)

private string path = null; //Для хранения пути к INI-файлу

//Импорт функции GetPrivateProfileString (для чтения значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "GetPrivateProfileString")]

private static extern int GetPrivateString(string section, string key, string def, StringBuilder buffer, int size, string path);

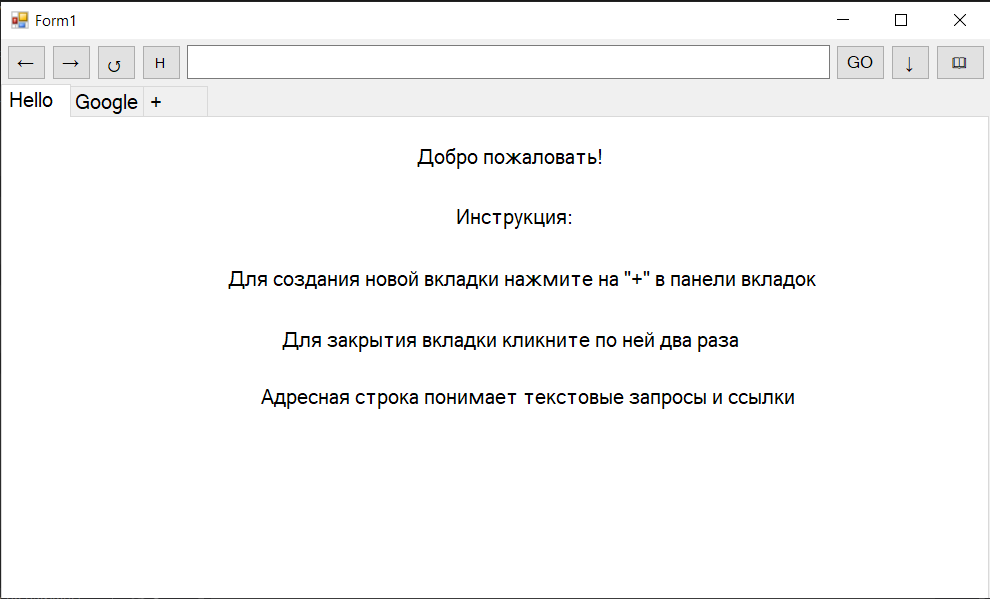
//Импорт функции WritePrivateProfileString (для записи значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "WritePrivateProfileString")]

private static extern int WritePrivateString(string section, string key, string str, string path);

}

Результат работы программы:

Рисунок 1 - Результат работы программы.

Описание: Для открытия адреса или запроса создайте новую вкладку кнопкой «+» в панели вкладок, введите запрос в адресную строку и нажмите на кнопку «GO». Для закрытия вкладки дважды кликните на нее. Для навигации используйте кнопки рядом с адресной строкой. Для открытия менеджера загрузок нажмите на кнопку «↓». Для открытия истории нажмите на кнопку «📖».

Вывод: Я научился работать с WebBrowser, Dictionaries и ListBox, грамотно их согласовывать, и создал приложение «Браузер».

# Практическая работа №5.

## Калькулятор

Постановка задачи: разработать программу калькулятор. Обычный: сложение, вычитание, умножение, деление. Инженерный: сложение, вычитание, умножение, деление, возведение в квадрат, возведение в куб, вычисление логарифма, вычисление факториала. Программист: возможность сложения, вычитания, умножения, деления в нескольких системах счисления (двоичная, восьмеричная, десятичная, шестнадцатиричная). Калькулятор дат: возможность вычислять разницу между датами.

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
| 1 | Form1.cs | Класс, который содержит методы Form1, AddNum, textBox1\_KeyPress, buttonAdd\_Click, buttonEqual\_Click, buttonLog\_Click, buttonDiv\_Click, buttonMulti\_Click, buttonSub\_Click, buttonAddMin\_Click, buttonAdd2\_Click, buttonPer\_Click, buttonSqr\_Click, buttonFactor\_Click, comboBox1\_SelectedIndexChanged, buttonRemove\_Click. Отвечает за проверку введенных пользователем данных и обработку/вычисление математических операций. |
| 2 | Prog.cs | Класс, который содержит методы prog, BtnClick. Отвечает за проверку введенных пользователем данных и обработку/вычисление математических операций. |
| 3 | Dates.cs | Класс, который содержит методы Dates, dateTimePicker1\_ValueChanged, ToDateDiff. Отвечает за проверку введенных пользователем данных и вычисление даты. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

float a, b;

int count;

bool znak = true;

Button[] buttons;

public Form1()

{

InitializeComponent();

buttons = new Button[]

{

button0, button1, button2, button3, button4, button5, button6, button7, button8, button9, buttonComma

};

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Click += AddNum;

}

}

private void AddNum(object sender, EventArgs e)

{

textBox1.Text = textBox1.Text + (sender as Button).Text;

} //Добавление цифры

private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

e.Handled = !char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar);

} //Проверка на буквы

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 1;

label1.Text = a.ToString() + "+";

znak = true;

} //Сумма

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

{

switch (count)

{

case 1:

b = a + float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 2:

b = a - float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 3:

b = a \* float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 4:

b = a / float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 5:

b = a \* float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 6:

b = a/100\* float.Parse(textBox1.Text);

textBox1.Text = b.ToString();

break;

case 7:

double c;

c = Math.Sqrt(a);

textBox1.Text = c.ToString();

break;

default:

break;

}

label1.Text = "";

} //Равно

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

{

if (textBox1.Text != "")

{

try

{

Console.WriteLine(textBox1.Text);

if (double.Parse(textBox1.Text) != 0 && double.Parse(textBox1.Text) > 0)

{

textBox1.Clear();

textBox1.Text = Math.Log(double.Parse(textBox1.Text)).ToString();

}

}

catch { MessageBox.Show("неправильно введено число"); }

}

} //Логарифм

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 4;

label1.Text = a.ToString() + "/";

znak = true;

} //деление

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 3;

label1.Text = a.ToString() + "\*";

znak = true;

} //Умножение

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 2;

label1.Text = a.ToString() + "-";

znak = true;

} //вычитание

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

{

if (znak == true)

{

textBox1.Text = "-" + textBox1.Text;

znak = false;

}

else if (znak == false)

{

textBox1.Text = textBox1.Text.Replace("-", "");

znak = true;

}

} //Добавление минуса

private void buttonAdd2\_Click(object sender, EventArgs e)

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 5;

label1.Text = a.ToString() + "^";

znak = true;

} //квадрат

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 6;

label1.Text = a.ToString() + "%";

znak = true;

} //Процент

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

{

a = float.Parse(textBox1.Text);

textBox1.Clear();

count = 7;

label1.Text = "sqrt(" + a.ToString() + ")";

znak = true;

} //Квадратный корень

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

{

if (textBox1.Text != "")

{

if (int.Parse(textBox1.Text) != 0 && int.Parse(textBox1.Text) > 0)

{

int res = 1;

for (int i = int.Parse(textBox1.Text); i > 1; i--)

res \*= i;

textBox1.Clear();

textBox1.Text = res.ToString();

}

}

} //Факториал

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

switch((sender as ComboBox).SelectedIndex)

{

case 1:

prog prog = new prog();

prog.Show();

break;

case 2:

Dates dates = new Dates();

dates.Show();

break;

}

} //Меняем тип калькулятора

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

{

try

{

label1.Text = "";

textBox1.Text = textBox1.Text.Remove(textBox1.Text.Length - 1);

}

catch { }

} //Стирание последнего символа

}

### Prog.cs

public partial class prog : Form

{

Button[] buttons;

public prog()

{

InitializeComponent();

buttons = new Button[]

{

button1, button2, button3, button5, button6, button7, button9, button10, button11, button13, button15

};

for (int i = 0; i < buttons.Length; i++)

{

buttons[i].Click += BtnClick;

}

}

private void BtnClick(object sender, EventArgs e)

{

textBox3.Text += (sender as Button).Text;

if((sender as Button).Text == "Clear")

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox4.Clear();

}

}

private void button4\_Click(object sender, EventArgs e)

{

try

{

textBox1.Text = Convert.ToString(int.Parse(textBox3.Text), 2);

textBox2.Text = Convert.ToString(int.Parse(textBox3.Text), 8);

textBox4.Text = Convert.ToString(int.Parse(textBox3.Text), 16);

}

catch { }

}

}

### Dates.cs

public partial class Dates : Form

{

public Dates()

{

InitializeComponent();

}

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

DateTime First = dateTimePicker1.Value;

DateTime Second = dateTimePicker2.Value;

TimeSpan div = (Second - First).Duration();

label1.Text = ToDateDiff(div.Days);

}

private string ToDateDiff(int Day)

{

int year = Day / 365;

int month = (Day % 365) / 30;

int day = Day % 30;

string YearRes;

if (year > 19) { year = year % 10; }

switch (year)

{

case 1: YearRes = "год"; break;

case 4: YearRes = "года"; break;

case 0: YearRes = "лет"; break;

default: YearRes = "ошибка"; break;

}

string nYear = YearRes;

string MonthRes;

if (month > 19) { month = month % 10; }

switch (month)

{

case 1: MonthRes = "месяц"; break;

case 4: MonthRes = "месяца"; break;

case 0: MonthRes = "месяцев"; break;

default: MonthRes = "ошибка"; break;

}

string nMonth = MonthRes;

string DayRes;

if (day > 19) { day = day % 10; }

switch (day)

{

case 1: DayRes = "день"; break;

case 4: DayRes = "дня"; break;

case 0: DayRes = "дней"; break;

default: DayRes = "ошибка"; break;

}

string nDay = DayRes;

if(year == 0 && month == 0 && day == 0) return string.Format("Одинаковые даты");

else return string.Format($"{year} {nYear} {month} {nMonth} {day} {nDay}");

}

Результат работы программы:

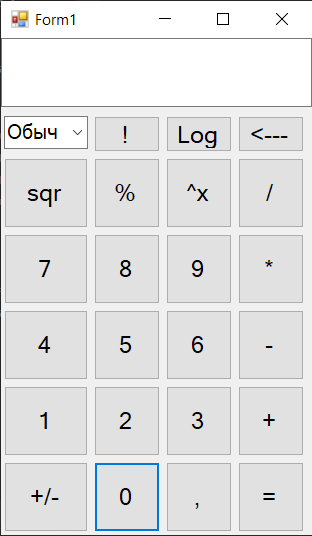


Рисунок 1 - Результат работы программы.

Описание: Для изменения типа калькулятора выберите соответствующий пункт в выпадающем списке. Для получения результата введите нужные действия и нажмите кнопку «=». Для вычисления разности дат введите обе даты в поля и результат отобразиться ниже.

Вывод: Я научился работать с математическими операциями, строками и датами, и создал приложение «Калькулятор» под разные типы задач.

# Практическая работа №6.

## Почтовый клиент

Постановка задачи: разработать программу для отправки электронных писем. Необходимо предусмотреть возможность ввода и сохранения настроек пользователя (сервер, порт, логин, пароль), отправку сообщений (ввод темы письма и его содержимого)

Таблица 1- Модули.

|  |  |  |
| --- | --- | --- |
| № | Название класса | Описание |
| 1 | Form1.cs | Класс, который содержит методы Form1, SendBtn\_Click. Отвечает за проверку введенных пользователем данных и отправку письма. |
| 2 | INIManager | Класс, который содержит методы GetPrivateString, WritePrivateString. Отвечает за функцию чтения/записи в INI-файл. |

Исходный код программы:

### Form1.cs

public partial class Form1 : Form

{

INIManager manager = new INIManager(Directory.GetCurrentDirectory() + "\\User.ini");

public Form1()

{

InitializeComponent();

EmailBox.Text = manager.GetPrivateString("User", "Login");

PasswordBox.Text = manager.GetPrivateString("User", "Password");

}

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

{

String Email = EmailBox.Text;

String Password = PasswordBox.Text;

String To = ToBox.Text;

String Subject = SubjectBox.Text;

String Message = MessageBox.Text;

if (Regex.IsMatch(Email, @"^\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$", RegexOptions.IgnoreCase) && Password != "" && Regex.IsMatch(To, @"^\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$", RegexOptions.IgnoreCase))

{

manager.WritePrivateString("User", "Login", Email);

manager.WritePrivateString("User", "Password", Password);

MailAddress from = new MailAddress(Email);

MailAddress to = new MailAddress(To);

MailMessage m = new MailMessage(from, to);

if(Subject == "") m.Subject = "Без темы";

else m.Subject = Subject;

if (Message == "") m.Body = "Без сообщения";

else m.Body = Message;

SmtpClient smtp = new SmtpClient();

try

{

smtp = new SmtpClient("smtp.gmail.com", 587);

}

catch

{

try

{

smtp = new SmtpClient("smtp.yandex.ru", 587);

}

catch

{

try

{

smtp = new SmtpClient("smtp.mail.ru", 587);

}

catch

{

label6.Text = "Не удалось подключиться к почтовому серверу";

}

}

}

smtp.Credentials = new NetworkCredential(Email, Password);

smtp.EnableSsl = true;

try {

smtp.Send(m);

label6.Text = "Отправлено";

}

catch {

label6.Text = "Ошибка отправки";

}

}

else

{

label6.Text = "Заполнены не все поля или неправильно введен адрес";

}

}

}

### INIManager

public class INIManager

{

//Конструктор, принимающий путь к INI-файлу

public INIManager(string aPath)

{

path = aPath;

}

//Конструктор без аргументов (путь к INI-файлу нужно будет задать отдельно)

public INIManager() : this("") { }

//Возвращает значение из INI-файла (по указанным секции и ключу)

public string GetPrivateString(string aSection, string aKey)

{

//Для получения значения

StringBuilder buffer = new StringBuilder(SIZE);

//Получить значение в buffer

GetPrivateString(aSection, aKey, null, buffer, SIZE, path);

//Вернуть полученное значение

return buffer.ToString();

}

//Пишет значение в INI-файл (по указанным секции и ключу)

public void WritePrivateString(string aSection, string aKey, string aValue)

{

//Записать значение в INI-файл

WritePrivateString(aSection, aKey, aValue, path);

}

//Возвращает или устанавливает путь к INI файлу

public string Path { get { return path; } set { path = value; } }

//Поля класса

private const int SIZE = 1024; //Максимальный размер (для чтения значения из файла)

private string path = null; //Для хранения пути к INI-файлу

//Импорт функции GetPrivateProfileString (для чтения значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "GetPrivateProfileString")]

private static extern int GetPrivateString(string section, string key, string def, StringBuilder buffer, int size, string path);

//Импорт функции WritePrivateProfileString (для записи значений) из библиотеки kernel32.dll

[DllImport("kernel32.dll", EntryPoint = "WritePrivateProfileString")]

private static extern int WritePrivateString(string section, string key, string str, string path);

}

Результат работы программы:

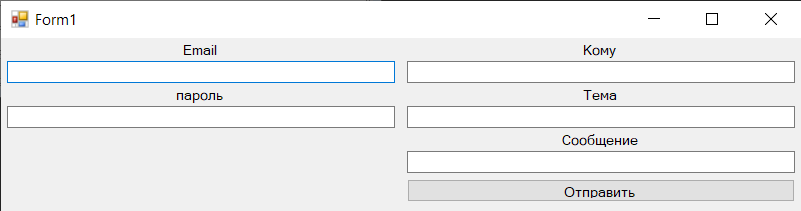


Рисунок 1 - Результат работы программы.

Описание: Для отправки сообщения заполните все поля и нажмите на кнопку «Отправить». Под полем «Пароль» отобразится статус письма.

Вывод: Я научился работать с SMTP-клиентом и создал простейший почтовый клиент.