**Пермский национальный исследовательский политехнический университет**

Кафедра “Информационные технологии и автоматизированные системы”

**Творческая работа**

По дисциплине «Основы алгоритмизации и программирования»

**Тема:**

Разработка калькулятора, задача Коммивояжера

Выполнил:

студент группы ИВТ-19-1б

Дворянских Д.А.

Проверила:

доцент кафедры “ИТАС”

Полякова О.А.

Пермь, 2020

**Разработка калькулятора**

**Постановка задачи**

|  |  |
| --- | --- |
| Обыкновенные и десятичные дроби. | Сложение, вычитание, умножение, деление, преобразование десятичной дроби в обыкновенную и обратно, отмена последней операции, сброс результата. |

**Код**

MyForm.cpp

#include "MyForm.h"

#include <vector>

#include <string>

#include <msclr\marshal\_cppstd.h>

using namespace cl;

using namespace std;

Double num;

vector <float> vec{ 0,1,1 };

void main() {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew MyForm);

}

using namespace System;

System::Void cl::MyForm::textBox1\_KeyPress(System::Object^ sender, System::Windows::Forms::KeyPressEventArgs^ e)

{

char number = e->KeyChar;

if ((e->KeyChar <= 47 || e->KeyChar > 58)&& (e->KeyChar <= 39 || e->KeyChar > 46) && number != 47 && number!=8) //калькулятор

{

e->Handled = true;

}

}

System::Void cl::MyForm::button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

int c = 0;

vec[1] = 1; vec[2] = 1;

String^ dl = textBox1->Text;

string stroka = msclr::interop::marshal\_as<string>(dl);

string res;

//Вычисляем сложение и вычитание

for (int k = 0; k < stroka.size(); k++) {

////

if (k == stroka.size() - 1) { res.push\_back(stroka[k]);

/////

if (c == 1) { vec[1] \*= stof(res); c=0; res = ""; }else

if(c==2){ vec[2] \*= stof(res); c = 0; res = ""; }

else

vec[0] += stof(res);

}

//когда вижу знак плюс

if (stroka[k] == '+') {

if (c == 1) { vec[1] \*= stof(res); c=0; res = ""; }

else if(c==2){ vec[2] \*= stof(res); c = 0; res = ""; }

else {vec[0] += stof(res); res = ""; }

}

else

//Когда вижу знак минус

if ((stroka[k] == '-') && (k != 0)) {

if (c == 1) { vec[1] \*= stof(res); c = 0; res = ""; res.push\_back(stroka[k]); }

else

if (c == 2) { vec[2] \*= stof(res); c = 0; res = ""; res.push\_back(stroka[k]); }

else

{

vec[0] += stof(res); res = ""; res.push\_back(stroka[k]);

}

}

else

//Умножение и деление

if (stroka[k] == '\*') {

if (c == 0) { vec[1] /= vec[2]; vec[0] += vec[1]; vec[2] = 1; vec[1] = 1; }

if (c == 2) {

vec[2] \*= stof(res); res = ""; c = 1;

}

else

{

vec[1] \*= stof(res); res = ""; c = 1;

}

}

else

if (stroka[k] == '/') {

if (c == 0) { vec[1] /= vec[2]; vec[0] += vec[1]; vec[2] = 1; vec[1] = 1; }

if (c == 1) { vec[1] \*= stof(res); res = ""; c = 2; }

else { vec[1] \*= stof(res); res = ""; c = 2; }

}

else

res.push\_back(stroka[k]);

}

////

vec[1] /= vec[2];

vec[0] += vec[1]-1;

vec[1] = 0;

vec[2] = 0;

textBox2->Text = Convert::ToString(vec[0]);

vec[0] = 0;

}

System::Void cl::MyForm::button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox1->Text = ""; textBox2->Text = "";

}

MyForm.h

#pragma once

namespace cl {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->button2 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// button2

//

this->button2->Location = System::Drawing::Point(12, 165);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(154, 47);

this->button2->TabIndex = 1;

this->button2->Text = L"Сброс";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// textBox1

//

this->textBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox1->Location = System::Drawing::Point(12, 39);

this->textBox1->Multiline = true;

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(342, 38);

this->textBox1->TabIndex = 9;

this->textBox1->KeyPress += gcnew System::Windows::Forms::KeyPressEventHandler(this, &MyForm::textBox1\_KeyPress);

//

// button1

//

this->button1->Location = System::Drawing::Point(199, 165);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(155, 47);

this->button1->TabIndex = 0;

this->button1->Text = L"=";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// textBox2

//

this->textBox2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox2->Location = System::Drawing::Point(12, 106);

this->textBox2->Multiline = true;

this->textBox2->Name = L"textBox2";

this->textBox2->ReadOnly = true;

this->textBox2->Size = System::Drawing::Size(342, 38);

this->textBox2->TabIndex = 10;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 14.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(192)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));

this->label1->Location = System::Drawing::Point(12, 80);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(72, 24);

this->label1->TabIndex = 13;

this->label1->Text = L"Ответ:";

//

// label2

//

this->label2->AutoSize = true;

this->label2->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(192)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));

this->label2->Location = System::Drawing::Point(12, 23);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(346, 13);

this->label2->TabIndex = 14;

this->label2->Text = L"Возможные операции: деление, умножение, сложение, вычитание";

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::Color::Indigo;

this->ClientSize = System::Drawing::Size(366, 232);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button2);

this->Controls->Add(this->button1);

this->MaximizeBox = false;

this->MaximumSize = System::Drawing::Size(382, 271);

this->MinimumSize = System::Drawing::Size(382, 271);

this->Name = L"MyForm";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Calc";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e);

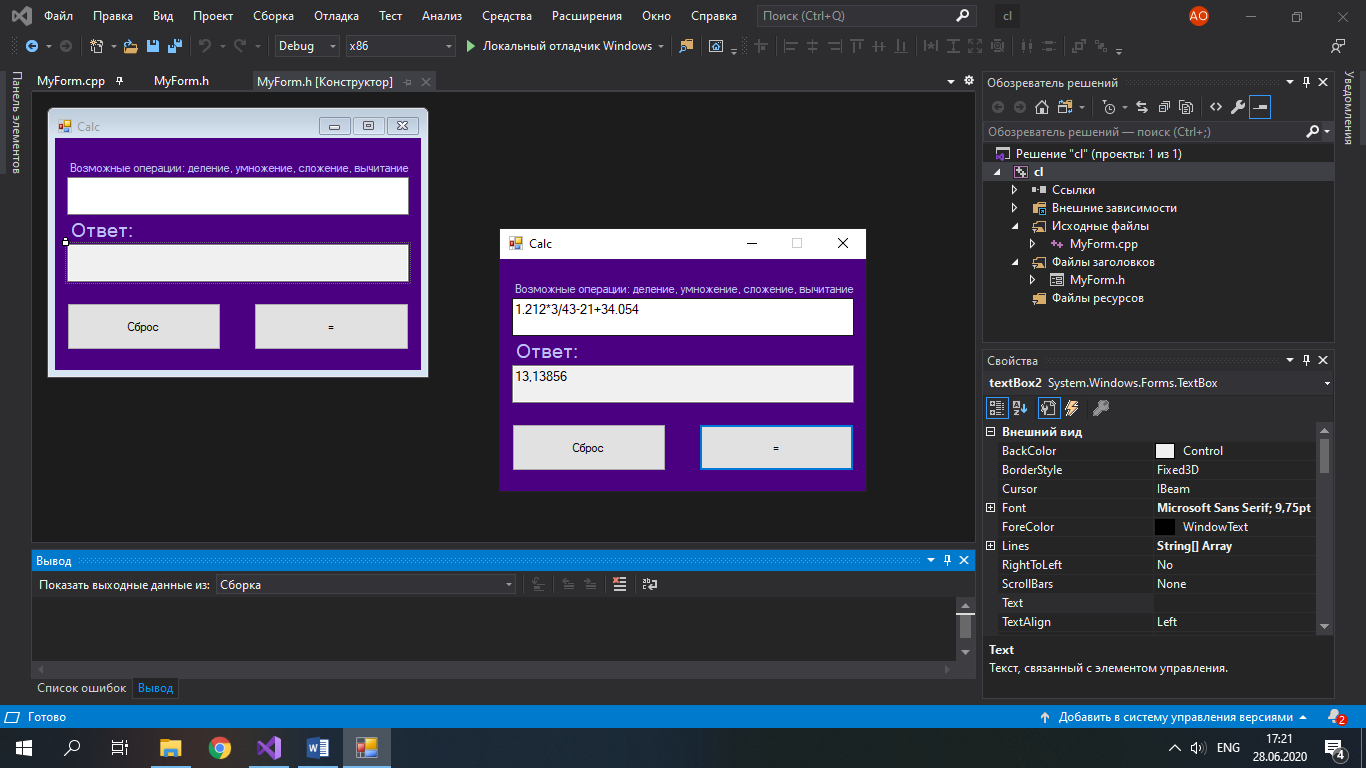
private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void textBox1\_KeyPress(System::Object^ sender, System::Windows::Forms::KeyPressEventArgs^ e);

};

}

**Тестирование**

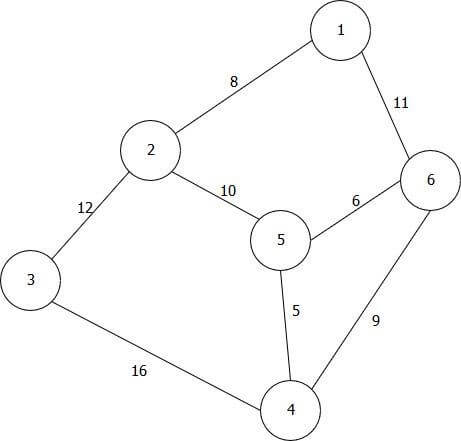


**Задача Коммивояжера**

**Постановка задачи**

Реализовать алгоритм Дейкстры для графа, соответствующего выбранному варианту.

1. Без использования Windows Forms.
2. Интерфейс на усмотрение разработчика.
3. Визуализация графа в OpenGL.
4. Реализуется только метод Дейкстры



**Код**

#include <glut.h>

#include <iostream>

#include <vector>

#include <string>

int value=0;

using namespace std;

int windh, windw; // window height and width

int c = 0;

vector <int> vec{ 1,2,3,4,6,5 };

vector <int> X{ 0,-1000,-1500,300,800,-80 };

vector <int> Y{ 0,-500,-1000,-1300,-500,-600 };

string res= "0";

class sv {

public:

int mass[6]{ 0,0,0,0,0,0 };

sv(int k1, int c1, int k2, int c2, int k3, int c3) {

mass[k1] = c1;

mass[k2] = c2;

mass[k3] = c3;

}

};

vector <sv> way;

vector <int> v{ rand() % 20+11,rand() % 20 + 10,rand() % 10 + 1,rand() % 20 + 12,rand() % 20 + 12,rand() % 20 + 11,rand() % 20 + 11,rand() % 20 + 11, };

//Просто рисует число и окружность

void draw(int k, int x, int y) {

glPushMatrix();

glTranslated(x, y, 0);

glutWireSphere(150, 20, 10);

glTranslated(-70, -40, 0.0);

string t = to\_string(k);

for (auto c1 = t.begin(); c1 != t.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

///Нажатие клавиши

void TRKey(unsigned char k, int a, int b) {

if (res[0] == '0') { res.pop\_back(); }

res.push\_back(k);

}

///

void drawnum(int c, int x, int y) {

glPushMatrix();

glTranslated(x, y, 0);

string t = to\_string(c);

for (auto c1 = t.begin(); c1 != t.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

void drawin(string c,int x,int y) {

glPushMatrix();

glTranslated(x, y, 0);

glScalef(0.8f, 0.8f, 0.8f);

for (auto c1 = c.begin(); c1 != c.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

//Результат

void drawres(string k, int x, int y) {

glPushMatrix();

glTranslated(x, y, 0);

glTranslated(200, 80, 0.0);

string t = k;

if (res[0] == '0') { t = "\_ \_"; }

for (auto c1 = t.begin(); c1 != t.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

//риусую линию

void drawline(int x, int y, int x1, int y1) {

glBegin(GL\_LINES);

if ((x > x1) && (y > y1)) { glVertex3f(x - 150, y, 0); glVertex3f(x1 + 150, y1, 0); }

if ((x > x1) && (y < y1)) { glVertex3f(x, y + 150, 0); ; glVertex3f(x1 + 150, y1, 0); }

if ((x < x1) && (y > y1)) { glVertex3f(x + 150, y, 0); ; glVertex3f(x1, y1 + 150, 0); }

if ((x < x1) && (y < y1)) { glVertex3f(x + 150, y, 0); ; glVertex3f(x1, y1 - 150, 0); }

glEnd();

}

//область рисования

void reshape(int wid, int ht)

{

gluOrtho2D(-wid / 2, wid / 2, -ht / 2, ht / 2);

windw = wid;

windh = ht;

}

void redraw(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

//начало откуда рисует

glPushMatrix();

glTranslated(0, windh / 3, 0);

glScalef(0.3f, 0.3f, 0.3f);

glColor3d(0, 0.1, 1);

glLineWidth(5);

for (int k = 0; k < vec.size(); k++) {

draw(vec[k], X[k], Y[k]);

}

for (int k = 0; k < vec.size() - 1; k++) {

drawline(X[k], Y[k], X[k + 1], Y[k + 1]);

}

///Нарисуем линии

drawline(X[4], Y[4], X[0], Y[0]);

drawline(X[5], Y[5], X[1], Y[1]);

drawline(X[3], Y[3], X[5], Y[5]);

//нарисуем расстояния

glLineWidth(2); glColor3d(0, 1, 1);

drawnum(v[0], -600, -200); drawnum(v[2], -1400, -700); drawnum(v[1], 500, -170);

drawnum(v[5], -400, -400); drawnum(v[6], 320, -450); drawnum(v[3], 700, -1000);

drawnum(v[4], -750, -1000); drawnum(v[7], 80, -1000);

//нарисуем результат

drawres(res, 1300, -1780);

if (stoi(res)== way[0].mass[3]) {

glPushMatrix();

glTranslated(1500, -40, 0.0);

glColor3d(0, 1, 0);

string h = "BepHo!";

for (auto c1 = h.begin(); c1 != h.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

else {

if(res.size()>1){

glPushMatrix();

glTranslated(1500, -40, 0.0);

glColor3d(1, 0, 0);

string h = "HE BepHo!";

for (auto c1 = h.begin(); c1 != h.end(); ++c1)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*c1);

}

glPopMatrix();

}

}

drawin("Kakoe MuHuMalHoe paccTo9Hue Do To4ku 4? BBeDuTe 3Ha4eHue", -2200, -1700);

glPopMatrix();

glutSwapBuffers();

}

int main(int argc, char\* argv[])

{

setlocale(LC\_ALL, "rus");

windw = 1600;

windh = 900;

way.push\_back(\*new sv(1, v[0], 5, v[1], 0, 0));

way.push\_back(\*new sv(0, v[0], 4, v[5], 2, v[2]));

way.push\_back(\*new sv(1, v[2], 3, v[4], 0, 0));

way.push\_back(\*new sv(2, v[4], 4, v[7], 5, v[3]));

way.push\_back(\*new sv(1, v[5], 3, v[7], 5, v[6]));

way.push\_back(\*new sv(3, v[3], 0, v[1], 4, v[6]));

for (int k = 1; k < 6; k++) {

for (int k1 = 0; k1 < 6; k1++) {

if ((way[k].mass[k1] != 0) && (way[0].mass[k1] != 0) && ((way[0].mass[k] + way[k].mass[k1]) < way[0].mass[k1])) {

way[0].mass[k1] = way[k].mass[k1] + way[0].mass[k];

}

if ((way[k].mass[k1] != 0) && (way[0].mass[k1] == 0)) { way[0].mass[k1] = way[k].mass[k1] + way[0].mass[k]; }

}

}

glutInit(&argc, argv);

glutInitWindowSize(windw, windh);

glutInitWindowPosition(0, 0);

glutInitDisplayMode(GLUT\_DEPTH | GLUT\_DOUBLE | GLUT\_RGBA);

(void)glutCreateWindow("31");

glutReshapeFunc(reshape);

glutDisplayFunc(redraw);

glClearColor(0, 0, 0, 1.0f);

cout << "От точки 1 наименьшие пути:\n";

for (int k = 1; k < 6; k++) {

cout << "До точки " << k + 1 << " = " << way[0].mass[k] << endl;

}

glutKeyboardFunc(TRKey);

glutMainLoop();

return 0;

}

**Тестирование**

