# Код программы «Алгоритмы на графах»

<Window x:Class="WpfApp1.MainWindow"

 xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

 xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:WpfApp1"

mc:Ignorable="d"

 Title="Алгоритмыработысграфами" Height="400" Width="525">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="auto"/>

<RowDefinition Height="200"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<MenuGrid.Row="0"Grid.ColumnSpan="2">

<MenuItem Header="Открытьматрицуграфа" Click="MenuItem\_Click"></MenuItem>

<MenuItem Header="АлгоритмДейкстра" Click="MenuItem\_Click\_1"></MenuItem>

<MenuItem Header="A\*" Click="MenuItem\_Click\_2" ></MenuItem>

<MenuItem Header="АлгоритмПрима" Click="MenuItem\_Click\_3"></MenuItem>

<MenuItem Header="Очистить" Click="MenuItem\_Clear"></MenuItem>

</Menu>

<GridGrid.Row="1"Grid.Column="0">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<ComboBoxGrid.Row="0" Name="ComboBox" Visibility="Hidden">

<ComboBoxItem Selected="ComboBoxItem\_Selected">Расстояние</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_1" >Время</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_2" >Рейтинг</ComboBoxItem>

</ComboBox>

<TextBoxGrid.Row="1" Name="Answer2" Visibility="Hidden"/>

<TextBoxGrid.Row="2" Name="Answer3" Visibility="Hidden"/>

<TextBoxGrid.Row="3" Name="Answer4" Visibility="Hidden"/>

</Grid>

<TextBoxGrid.Column="0"Grid.Row="2"Grid.ColumnSpan="2" Name="matrix"RenderTransformOrigin="0.5,0.5">

<TextBox.RenderTransform>

<TransformGroup>

<ScaleTransform/>

<SkewTransform/>

<RotateTransform Angle="-0.127"/>

<TranslateTransform/>

</TransformGroup>

</TextBox.RenderTransform>

</TextBox>

<TextBoxGrid.Column="0"Grid.Row="3"Grid.ColumnSpan="2" Name="vrct" />

<GridGrid.Row="1"Grid.Column="1">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<ButtonGrid.Row="0" Name="Butt" Click="Butt\_Click" Visibility="Hidden" Content="Обходвширину" />

<TextBox Name="Answer"Grid.Row="3" Visibility="Hidden"/>

<Label Name ="lable" Content="Обходвширину"Grid.Row="2" Visibility="Hidden"/>

<ComboBoxGrid.Row="0" Name="ComboBox2" Visibility="Hidden" >

<ComboBoxItem Selected="ComboBoxItem\_Selected\_3" >Расстояние</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_4">Время</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_5">Рейтинг</ComboBoxItem>

</ComboBox>

<ComboBoxGrid.Row="0" Name="ComboBox3" Visibility="Hidden">

<ComboBoxItem Selected="ComboBoxItem\_Selected\_6" >Расстояние</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_7">Время</ComboBoxItem>

<ComboBoxItem Selected="ComboBoxItem\_Selected\_8" >Рейтинг</ComboBoxItem>

</ComboBox>

<TextBoxGrid.Row="1" Name="Answer5" Visibility="Hidden"/>

<TextBoxGrid.Row="2" Name="Answer6" Visibility="Hidden"/>

<TextBoxGrid.Row="3" Name="Answer7" Visibility="Hidden"/>

<TextBoxGrid.Row="1" Name="Answer8" Visibility="Hidden"/>

<TextBoxGrid.Row="2" Name="Answer9" Visibility="Hidden"/>

<TextBoxGrid.Row="3" Name="Answer10" Visibility="Hidden"/>

</Grid>

</Grid>

</Window>

using System;

usingSystem.Collections;

usingSystem.Collections.Generic;

usingSystem.Collections.ObjectModel;

using System.IO;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows;

usingSystem.Windows.Controls;

usingSystem.Windows.Data;

usingSystem.Windows.Documents;

usingSystem.Windows.Input;

usingSystem.Windows.Media;

usingSystem.Windows.Media.Imaging;

usingSystem.Windows.Navigation;

usingSystem.Windows.Shapes;

namespace WpfApp1

{

///<summary>

///ЛогикавзаимодействиядляMainWindow.xaml

///</summary>

publicpartialclassMainWindow : Window

 {

publicMainWindow()

 {

InitializeComponent();

 }

constint n = 8;

string[] graf = newstring[n] { "0-Новосибирск", "1-Екатеринбург", "2-Москва", "3-Санкт-Петербург", "4-Париж", "5-Милан", "6-Барселона", "7-Лион" };

double[,] vrshina = newdouble[n, n];

constint m = 10;

List<int> ostov = newList<int>();

double[] vrc1 = newdouble[10] { 0, 7710.7, 9153.7, 9437.7, 8823.7, 8615.7, 9353.7, 0, 7776.7, 6979.7 };

double[] vrc2 = newdouble[10] { 0, 2065, 730, 670, 765, 750, 1350, 0, 700, 705 };

double[] vrc3 = newdouble[10] { 0, 5.884, 5.41, 6.33, 5.41, 5.91, 6.41, 0, 5.41, 5.41 };

ObservableCollection<Class1> graf2 = newObservableCollection<Class1>

 {

newClass1{Name="Новосибирск", index=0},

newClass1{Name="Екатеринбург", index=1},

newClass1{Name="Москва", index=2},

newClass1{Name="Санкт-Петербург", index=3},

newClass1{Name="Париж", index=4},

newClass1{Name="Милан", index=5 },

newClass1{Name="Барселона", index=6},

newClass1{Name="Лион", index=7},

newClass1{Name="Москва(1)", index=8},

newClass1{Name="Москва (2)", index=9},

 };

double[,] vrshina2 = newdouble[m, m];

voidPrintMatrix()

 {

matrix.Clear();

for (inti = 0; i< n; i++)

matrix.Text += graf[i] + " ";

matrix.Text += "\n";

matrix.Text += "\t\t";

for (inti = 0; i< n; i++)

matrix.Text += "" + (i) + "\t";

matrix.Text += "\n";

for (inti = 0; i< n; i++)

 {

matrix.Text += "\t" + (i) + "\t";

for (int j = 0; j < n; j++)

 {

matrix.Text += vrshina[i, j] + "\t";

 }

matrix.Text += "\n";

 }

matrix.Text += "\n";

 }

voidGetMatrix(string matrix)

 {

string[] elements = newstring[n];

StreamReader r = newStreamReader(@matrix);

inti = 0;

while (!r.EndOfStream)

 {

 elements = r.ReadLine().Split(',');

for (int j = 0; j < n; j++)

 {

if (elements[j].IndexOf(".") != 0) elements[j] = elements[j].Replace(".", ",");

 vrshina[i, j] = Convert.ToDouble(elements[j]);

 }

 ++i;

 }

r.Close();

 }

voidObhod()

 {

Queue<int>och = newQueue<int>();

List<int> list = newList<int>();

list.Add(0);

for (int j = 0; j < n; j++)

if (vrshina[0, j] != 0) och.Enqueue(j);

while (och.Count != 0)

 {

int k = och.Dequeue();

if (list.Contains(k))

 k = 0;

else

 {

list.Add(k);

for (int j = 0; j < n; j++)

if (vrshina[k, j] != 0)

if (och.Contains(j) | list.Contains(j)) ;

elseoch.Enqueue(j);

 }

 }

int[] mas = newint[n];

 mas = list.ToArray();

for (inti = 0; i<mas.Count(); i++)

Answer.Text += mas[i] + " ";

 }

void PrintMatrix2()

 {

matrix.Clear();

for (inti = 0; i< m; i++)

matrix.Text += " " + Convert.ToString(graf2[i].index) + "-" + " " + graf2[i].Name;

matrix.Text += "\n";

matrix.Text += "\t\t";

for (inti = 0; i< m; i++)

matrix.Text += "" + (i) + "\t";

matrix.Text += "\n";

for (inti = 0; i< m; i++)

 {

matrix.Text += "\t" + (i) + "\t";

for (int j = 0; j < m; j++)

 {

matrix.Text += vrshina2[i, j] + "\t";

 }

matrix.Text += "\n";

 }

matrix.Text += "\n";

 }

void GetMatrix2(string matrix)

 {

string[] elements = newstring[m];

StreamReader r = newStreamReader(@matrix);

inti = 0;

while (!r.EndOfStream)

 {

 elements = r.ReadLine().Split(',');

for (int j = 0; j < m; j++)

 {

if (elements[j].IndexOf(".") != 0) elements[j] = elements[j].Replace(".", ",");

 vrshina2[i, j] = Convert.ToDouble(elements[j]);

 }

 ++i;

 }

r.Close();

 }

bool[] done = newbool[10];

int[] parent = newint[10];

double[] distances = newdouble[10];

double[] funk = newdouble[10];

List<int>vrc = newList<int>();

publicvoid All()

 {

for (inti = 0; i< 10; i++)

for (int j = 0; j < 10; j++)

 {

if (vrshina2[i, j] == 0)

 vrshina2[i, j] = int.MaxValue;

 }

for (inti = 0; i< 10; i++)

 { parent[i] = -1;

 distances[i] = int.MaxValue;

 funk[i] = int.MaxValue;

 done[i] = false;

 }

 }

publicvoidVrc(double[] vrc)

 {

vrct.Text = "";

for (inti = 0; i< graf2.Count(); i++)

 {

vrct.Text += graf2[i].Name + " - " + vrc[i] + " \n";

 }

 }

privatevoid Dijkstra(int start, int end)

 {

 distances[start] = 0;

int current = start;

while (!done[current])

 {

 done[current] = true;

for (inti = 0; i< 10; i++)

 {

if (vrshina2[current, i] != int.MaxValue)

 {

doubledist = vrshina2[current, i] + distances[current];

if (dist< distances[i])

 {

 distances[i] = dist;

 parent[i] = current;

 }

 }

 }

double min = int.MaxValue;

for (inti = 0; i< 10; i++)

 {

if (distances[i] < min && !done[i])

 {

 current = i;

 min = distances[i];

 }

 }

 }

 }

privatevoid Prim (int start)

 {

ostov.Clear();

 distances[start] = 0;

int current = start;

int k=0;

while (ostov.Count!=10)

 {

double min = int.MaxValue;

ostov.Add(current);

for (inti=0;i<ostov.Count;i++)

 {

int l = ostov.ElementAt(i);

for (int j=0;j<10;j++)

 {

if (ostov.Contains((graf2[j].index)) == false)

 {

if (vrshina2[l,j]!= int.MaxValue)

 {

 distances[j] = vrshina2[l, j] + distances[l];

 parent[j] = l;

if (vrshina2[l, j] < min)

 {

 min = vrshina2[l, j];

 k = j;

 }

 }

 }

 }

 }

 current = k;

 }

}

publicvoidAstar(int start, int end, double[] vrc)

 {

 distances[start] = 0;

int current = start;

while (!done[current])

 {

 done[current] = true;

for (inti = 0; i< 10; i++)

 {

if ((vrshina2[current, i] != int.MaxValue) & (!done[i]))

 {

doubledist = vrshina2[current, i] + distances[current] + vrc[i];

if (dist< funk[i])

 {

 funk[i] = dist;

 distances[i] = vrshina2[current, i] + distances[current];

 parent[i] = current;

 }

 }

 }

double min = int.MaxValue;

for (inti = 0; i< 10; i++)

 {

if (funk[i] < min && !done[i])

 {

 current = i;

 min = funk[i];

 }

 }

if (done[end] == true) break;

 }

 }

publicint[] Answers()

 {

Stack<int>otvet = newStack<int>();

int k;

int j = 7;

while (parent[j] != -1)

 {

 k = parent[j];

otvet.Push(j);

 j = k;

 }

otvet.Push(0);

int[] mas = newint[otvet.Count];

 mas = otvet.ToArray();

otvet.Clear();

return mas;

 }

privatevoidComboBoxItem\_Selected(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D1.txt");

 PrintMatrix2();

 All();

 Dijkstra(0, 7);

int[] mas = Answers();

 Answer2.Visibility = Visibility.Visible;

 Answer2.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer2.Text += graf2[l].Name + " ";

 }

 Answer2.Text += "\n" + "Минимальноерасстояние:" + " " + distances[7];

 }

privatevoidMenuItem\_Click(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

GetMatrix("f.txt");

PrintMatrix();

Butt.Visibility = Visibility.Visible;

 }

privatevoidButt\_Click(object sender, RoutedEventArgs e)

 {

Answer.Visibility = Visibility.Visible;

lable.Visibility = Visibility.Visible;

Obhod();

 }

privatevoid ComboBoxItem\_Selected\_1(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D2.txt");

 PrintMatrix2();

 PrintMatrix2();

 All();

 Dijkstra(0, 7);

int[] mas = Answers();

 Answer3.Visibility = Visibility.Visible;

 Answer3.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer3.Text += graf2[l].Name + " ";

 }

 Answer3.Text += "\n" + "Минимальноевремя:" + " " + distances[7];

 }

privatevoid ComboBoxItem\_Selected\_2(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D3.txt");

 PrintMatrix2();

 PrintMatrix2();

 All();

 Dijkstra(0, 7);

int[] mas = Answers();

 Answer4.Visibility = Visibility.Visible;

 Answer4.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer4.Text += graf2[l].Name + " ";

 }

 Answer4.Text += "\n" + "Минимальныйпуть:" + " " + distances[7];

 }

privatevoid MenuItem\_Click\_1(object sender, RoutedEventArgs e)

 {

ComboBox.Visibility = Visibility.Visible;

 }

privatevoid MenuItem\_Click\_2(object sender, RoutedEventArgs e)

 {

Butt.Visibility = Visibility.Hidden;

Answer.Visibility = Visibility.Hidden;

lable.Visibility = Visibility.Hidden;

 ComboBox3.Visibility = Visibility.Hidden;

 Answer8.Visibility = Visibility.Hidden;

 Answer9.Visibility = Visibility.Hidden;

 Answer10.Visibility = Visibility.Hidden;

 ComboBox2.Visibility = Visibility.Visible;

 }

privatevoid ComboBoxItem\_Selected\_3(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D1.txt");

 PrintMatrix2();

 All();

Vrc(vrc1);

Astar(0, 7, vrc1);

int[] mas = Answers();

 Answer5.Visibility = Visibility.Visible;

 Answer5.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer5.Text += graf2[l].Name + " ";

 }

 Answer5.Text += "\n" + "Минимальныйпуть:" + " " + funk[7];

 }

privatevoid ComboBoxItem\_Selected\_4(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D2.txt");

 PrintMatrix2();

 PrintMatrix2();

 All();

Vrc(vrc2);

Astar(0, 7, vrc2);

int[] mas = Answers();

 Answer6.Visibility = Visibility.Visible;

 Answer6.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer6.Text += graf2[l].Name + " ";

 }

 Answer6.Text += "\n" + "Минимальныйпуть:" + " " + funk[7];

 }

privatevoid ComboBoxItem\_Selected\_5(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D3.txt");

 PrintMatrix2();

 PrintMatrix2();

 All();

Vrc(vrc3);

Astar(0, 7, vrc3);

int[] mas = Answers();

 Answer7.Visibility = Visibility.Visible;

 Answer7.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer7.Text += graf2[l].Name + " ";

 }

 Answer7.Text += "\n" + "Минимальныйпуть:" + " " + funk[7];

 }

privatevoid MenuItem\_Click\_3(object sender, RoutedEventArgs e)

 {

 ComboBox2.Visibility = Visibility.Hidden;

 Answer5.Visibility = Visibility.Hidden;

 Answer6.Visibility = Visibility.Hidden;

 Answer7.Visibility = Visibility.Hidden;

 ComboBox3.Visibility = Visibility.Visible;

 }

privatevoid ComboBoxItem\_Selected\_6(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D1.txt");

 PrintMatrix2();

 All();

 Prim(0);

int[] mas = Answers();

 Answer8.Visibility = Visibility.Visible;

 Answer8.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer8.Text += graf2[l].Name + " ";

}

 Answer8.Text += "\n" + "Минимальныйпуть:" + " " + distances[7]+"\n"+"Остов:";

for (inti = 0; i<ostov.Count(); i++)

 {

 Answer8.Text +=Convert.ToString(ostov.ElementAt(i))+ " ";

 }

 }

privatevoid ComboBoxItem\_Selected\_7(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D2.txt");

 PrintMatrix2();

 All();

 Prim(0);

int[] mas = Answers();

 Answer9.Visibility = Visibility.Visible;

 Answer9.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer9.Text += graf2[l].Name + " ";

 }

 Answer9.Text += "\n" + "Минимальныйпуть:" + " " + distances[7] + "\n" + "Остов:";

for (inti = 0; i<ostov.Count(); i++)

 {

 Answer9.Text += Convert.ToString(ostov.ElementAt(i)) + " ";

 }

 }

privatevoid ComboBoxItem\_Selected\_8(object sender, RoutedEventArgs e)

 {

matrix.Visibility = Visibility.Visible;

 GetMatrix2("D3.txt");

 PrintMatrix2();

 All();

 Prim(0);

int[] mas = Answers();

 Answer10.Visibility = Visibility.Visible;

 Answer10.Text = "";

for (inti = 0; i<mas.Count(); i++)

 {

int l = mas[i];

 Answer10.Text += graf2[l].Name + " ";

 }

 Answer10.Text += "\n" + "Минимальныйпуть:" + " " + distances[7] + "\n" + "Остов:";

for (inti = 0; i<ostov.Count(); i++)

 {

 Answer10.Text += Convert.ToString(ostov.ElementAt(i)) + " ";

 }

 }

privatevoidMenuItem\_Clear(object sender, RoutedEventArgs e)

 {

Answer.Visibility = Visibility.Hidden;

 Answer2.Visibility = Visibility.Hidden;

 Answer3.Visibility = Visibility.Hidden;

 Answer4.Visibility = Visibility.Hidden;

 Answer5.Visibility = Visibility.Hidden;

 Answer6.Visibility = Visibility.Hidden;

 Answer7.Visibility = Visibility.Hidden;

 Answer8.Visibility = Visibility.Hidden;

 Answer9.Visibility = Visibility.Hidden;

 Answer10.Visibility = Visibility.Hidden;

Butt.Visibility = Visibility.Hidden;

Answer.Visibility = Visibility.Hidden;

lable.Visibility = Visibility.Hidden;

 ComboBox3.Visibility = Visibility.Hidden;

 ComboBox2.Visibility = Visibility.Hidden;

ComboBox.Visibility = Visibility.Hidden;

matrix.Visibility = Visibility.Hidden;

vrct.Visibility = Visibility.Hidden;

}

 }

 }