Условие:

Написать программу Windows Presentation Foundation (WPF) на C# для перемножения матриц разного вида

Исходный код программы:

using System.Windows;

using System.Windows.Controls;

namespace MatrixMultiplication

{

public partial class MainWindow : Window

{

double[,] matrix1, matrix2, result;

public MainWindow()

{

InitializeComponent();

}

private void matrixdimensions\_changed(object sender, SelectionChangedEventArgs e)

{

int m1rows = 1;

int m1columns\_m2rows = 1;

int m2columns = 1;

if (matrix1width != null) m1columns\_m2rows = matrix1width.SelectedIndex + 1;

if (matrix1height != null) m1rows = matrix1height.SelectedIndex + 1;

if (matrix2width != null) m2columns = matrix2width.SelectedIndex + 1;

matrix1 = new double[m1columns\_m2rows, m1rows];

matrix2 = new double[m2columns, m1columns\_m2rows];

result = new double[m2columns, m1rows];

initializeGrid(grid1, matrix1);

initializeGrid(grid2, matrix2);

initializeGrid(grid3, result);

}

private void initializeGrid(Grid grid, double[,] matrix)

{

if (grid != null)

{

grid.Children.Clear();

grid.ColumnDefinitions.Clear();

grid.RowDefinitions.Clear();

int columns = matrix.GetLength(0);

int rows = matrix.GetLength(1);

for (int x = 0; x < columns; x++)

grid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(1, GridUnitType.Star), });

for (int y = 0; y < rows; y++)

{

grid.RowDefinitions.Add(new RowDefinition() { Height = new GridLength(1, GridUnitType.Star),});

}

for (int x = 0; x < columns; x++)

for (int y = 0; y < rows; y++)

{

double cell = (double)matrix[x, y];

TextBox t = new TextBox();

t.Text = cell.ToString();

t.VerticalAlignment = System.Windows.VerticalAlignment.Center;

t.HorizontalAlignment = System.Windows.HorizontalAlignment.Center;

t.SetValue(Grid.RowProperty, y);

t.SetValue(Grid.ColumnProperty, x);

grid.Children.Add(t);

}

}

}

private void buttonCalculate\_Click(object sender, RoutedEventArgs e)

{

getValuesFromGrid(grid1, matrix1);

getValuesFromGrid(grid2, matrix2);

int m1stolb\_m2strok, m1strok, m2stolb;

m1stolb\_m2strok = matrix1.GetLength(0);

m1strok = matrix1.GetLength(1);

m2stolb = matrix2.GetLength(0);

for (int row = 0; row < m1strok; row++)

{

for (int column = 0; column < m2stolb; column++)

{

double accumulator = 0;

for (int cell = 0; cell < m1stolb\_m2strok; cell++)

{

accumulator = accumulator + (matrix1[cell, row] \* matrix2[column, cell]);

}

result[column, row] = accumulator;

}

}

initializeGrid(grid3, result);

}

private void getValuesFromGrid(Grid grid, double[,] matrix)

{

int columns = grid.ColumnDefinitions.Count;

int rows = grid.RowDefinitions.Count;

for (int c = 0; c < grid.Children.Count; c++)

{

TextBox t = (TextBox)grid.Children[c];

int row = Grid.GetRow(t);

int column = Grid.GetColumn(t);

matrix[column, row] = double.Parse(t.Text);

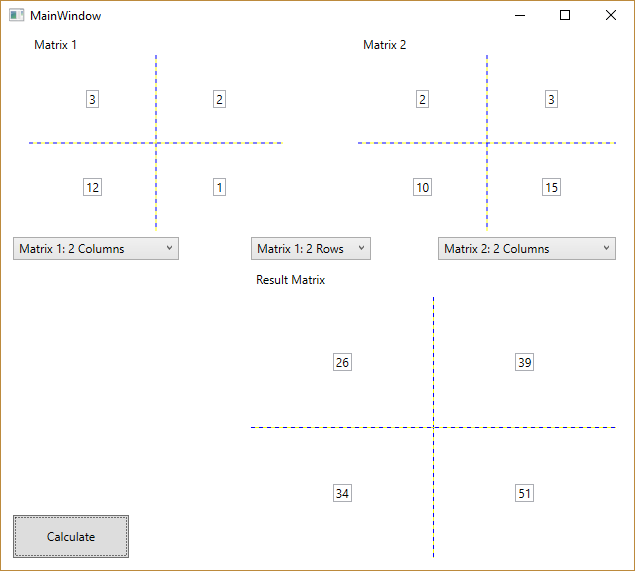
}

}

}

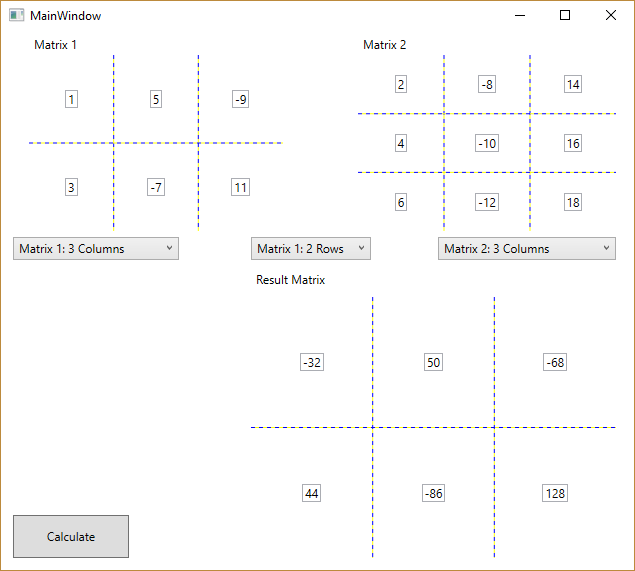
}

Скриншоты программы:

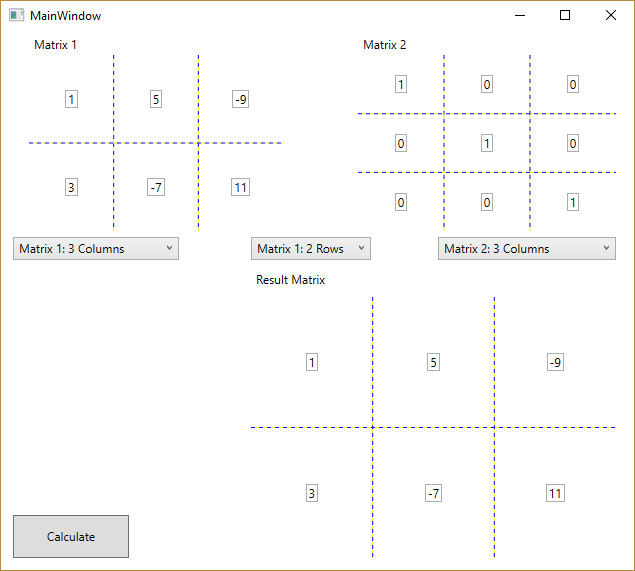


Тестирование программы:

| Матрица 1 | | |
| --- | --- | --- |
| 1 | 5 | –9 |
| 3 | –7 | 11 |
| Матрица 2 | | |
| 2 | –8 | 14 |
| 4 | –10 | 16 |
| 6 | –12 | 18 |
| Результат | | |
| –32 | 50 | –68 |
| 44 | –86 | 128 |



| Матрица 1 | | |
| --- | --- | --- |
| 1 | 514 | –9 |
| 3 | –7 | 11 |
| Матрица 2 | | |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |
| Результат | | |
| 1 | 5 | –9 |
| 3 | –7 | 11 |



| Матрица 1 | | |
| --- | --- | --- |
| 1 | 514 | –9 |
| 3 | –7 | 11 |
| Матрица 2 | | |
| –1 | 0 | 0 |
| 0 | –1 | 0 |
| 0 | 0 | –1 |
| Результат | | |
| –1 | –5 | 9 |
| –3 | 7 | –11 |

