Условие:

Написать программу Windows Presentation Foundation (WPF) на C# для вычисления наибольшего общего делителя по классическому и модифицированному (бинарному) алгоритмам Евклида.

Исходный код программы:

GCDAlgorithms.cs

namespace GreatestCommonDivisor

{

static class GCDAlgorithms

{

public static int FindGCDEuclid(int a, int b)

{

while (b != 0)

b = a % (a = b);

return a;

}

public static int FindGCDEuclid(int a, int b, int c)

{

while (b != 0)

b = a % (a = b);

return FindGCDEuclid(a,c);

}

public static int FindGCDEuclid(int a, int b, int c, int d)

{

while (b != 0)

b = a % (a = b);

return FindGCDEuclid(a, c, d);

}

public static int FindGCDEuclid(int a, int b, int c, int d, int e)

{

while (b != 0)

b = a % (a = b);

return FindGCDEuclid(a, c, d, e);

}

//Бинарный (модифицированный) алгоритм товарища Евклида

public static long Stain(long a, long b)

{

if (a == 0)

return b;

if (b == 0)

return a;

if (a == b)

return a;

if (a == 1 || b == 1)

return 1;

if ((a & 1) == 0)

return ((b & 1) == 0)

? Stain(a >> 1, b >> 1) << 1

: Stain(a >> 1, b);

else

return ((b & 1) == 0)

? Stain(a, b >> 1)

: Stain(b, a > b ? a - b : b - a);

}

}

}

MainWindow.xaml.cs

using System.Windows;

using System.Windows.Controls;

namespace GreatestCommonDivisor

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

/// <summary>

/// Do the GCD calculations

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

int firstNumber;

int secondNumber;

int thirdNumber;

int fourthNumber;

int fifthNumber;

if (!GetPostiveIntegerFromTextBox(integer1, out firstNumber)) return;

if (!GetPostiveIntegerFromTextBox(integer2, out secondNumber)) return;

if (!GetPostiveIntegerFromTextBox(integer3, out thirdNumber)) return;

if (!GetPostiveIntegerFromTextBox(integer4, out fourthNumber)) return;

if (!GetPostiveIntegerFromTextBox(integer5, out fifthNumber)) return;

if (sender == findGCD)

{

resultEuclid.Content = "Euclid: "+GCDAlgorithms.FindGCDEuclid(firstNumber, secondNumber);

resultStein.Content = "Stein: " + GCDAlgorithms.Stain(firstNumber, secondNumber);

}

if (sender == findGCD3)

resultEuclid.Content = "Euclid: " + GCDAlgorithms.FindGCDEuclid(firstNumber, secondNumber, thirdNumber);

if (sender == findGCD4)

resultEuclid.Content = "Euclid: " + GCDAlgorithms.FindGCDEuclid(firstNumber, secondNumber, thirdNumber, fourthNumber);

if (sender == findGCD5)

resultEuclid.Content = "Euclid: " + GCDAlgorithms.FindGCDEuclid(firstNumber, secondNumber, thirdNumber, fourthNumber, fifthNumber);

}

private bool GetPostiveIntegerFromTextBox(TextBox textBox, out int i)

{

i = -1;

if (int.TryParse(textBox.Text, out i))

{

if (i >= 0) return true;

}

MessageBox.Show("Not a positive integer value: " + textBox.Text);

return false;

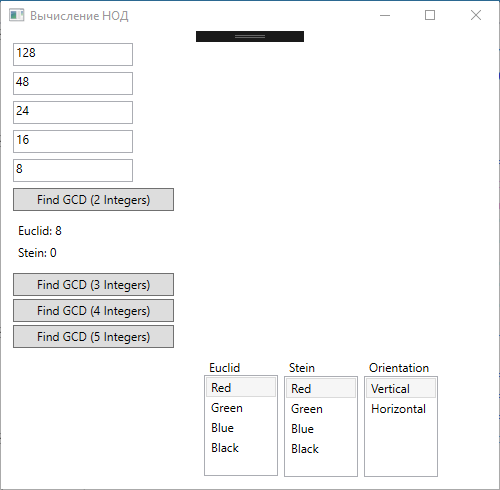
}

}

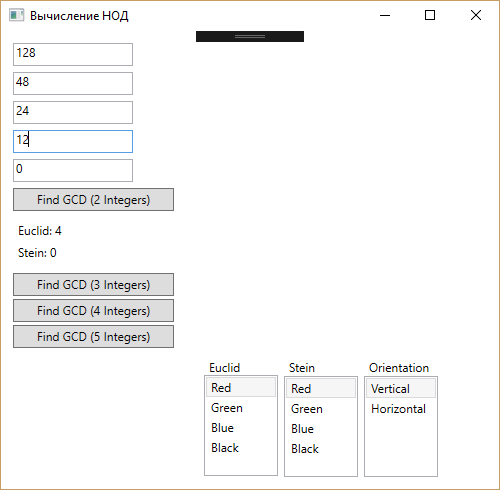
}

Скриншоты программы:

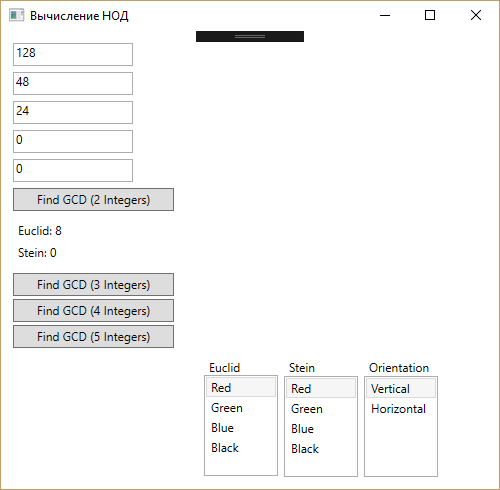
Для 5 значений:



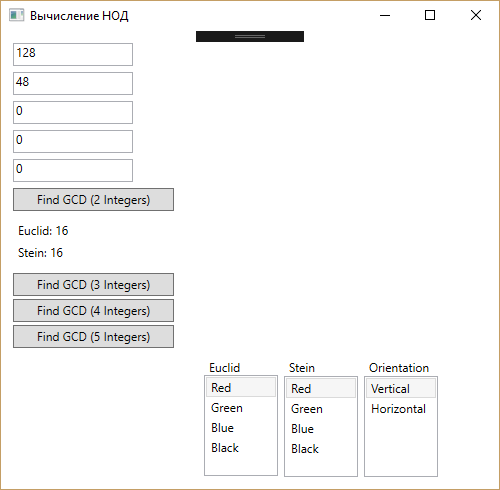
Для 4 значений:



Для 3 значений:

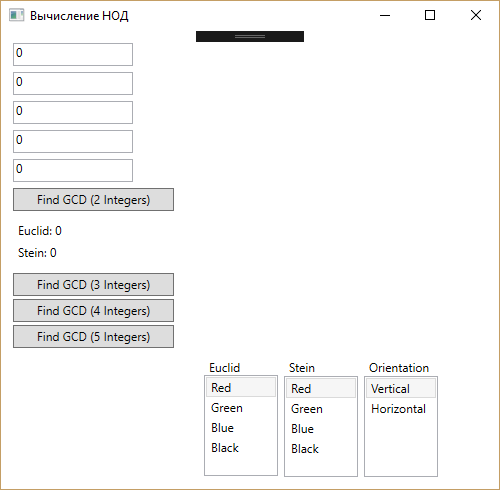


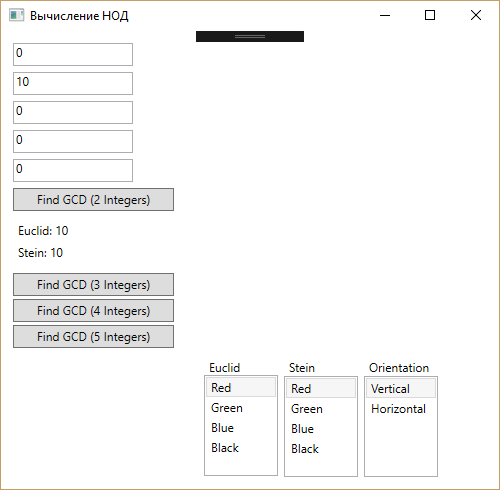
Для 2 значений:

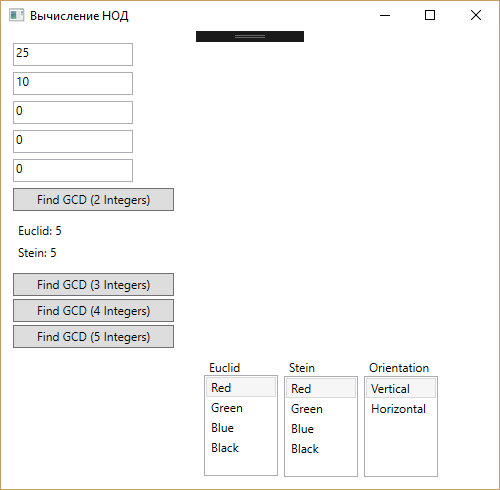


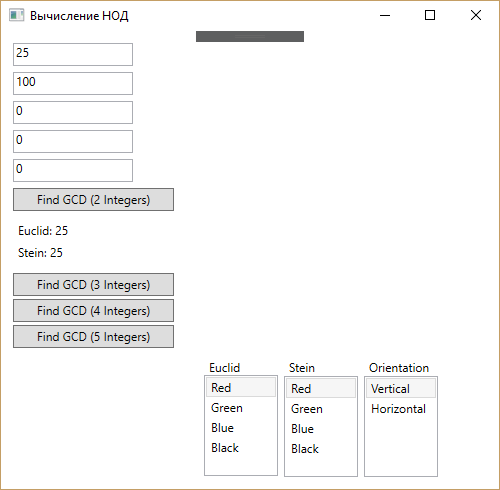
Тестирование программы:

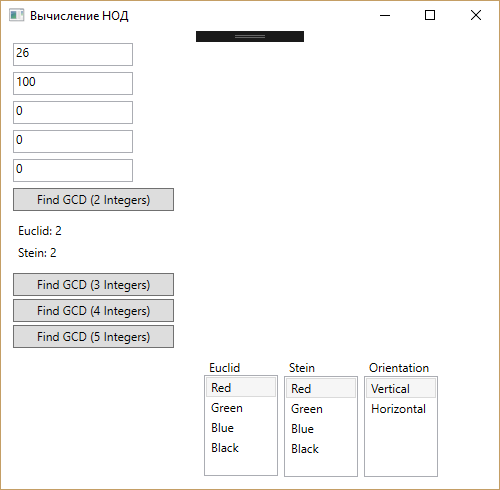
| **№** | **Первое число** | **Второе число** | **Результат** |
| --- | --- | --- | --- |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 10 | 10 |
| 3 | 25 | 10 | 5 |
| 4 | 25 | 100 | 25 |
| 5 | 26 | 100 | 2 |
| 6 | 27 | 100 | 1 |

1) 

2) 

3) 

4) 

5) 

6) 