**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

****

**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет информационных технологий**

**Кафедра Информатики и информационных технологий**

**направление подготовки 09.03.02 «Информационные системы и технологии»,**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**Дисциплина:** Технология прикладного программирования

**Выполнил:** студент группы 231-338

Шаура И. М.

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил:** ст. преп.Калмыков Е. А.\_\_\_\_\_\_\_\_\_\_\_

**(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2023**

**Задание:**

В рамках данной лабораторной работы необходимо дополнить проект калькулятор следующими элементами:

1. ListBox/TextBox (на выбор). Данный элемент должен использоваться для сохранения всех операций калькулятора. В случае TextBox указываются свойства TextWrapping="Wrap" AcceptsReturn="True"

2. CheckBox. Данный элемент отвечает за включение/выключения функции записи операций калькулятора в ListBox/TextBox

3. Button. Кнопка для полной очистки ListBox/TextBox

4. Button. Кнопка для удаления последней операции из ListBox/TextBox

5. Radiobutton (2 шт.). Для выбора режима видимости/невидимости элементов из пп. 1-4.

|  |
| --- |
| <Window x:Class="\_3.CalculatorPlus.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:\_3.CalculatorPlus"  mc:Ignorable="d"  Title="Calculator" Height="530" Width="640" ResizeMode="NoResize">  <Grid>  <Grid.ColumnDefinitions>  <ColumnDefinition x:Name="Column1" Width="300"/>  <ColumnDefinition x:Name="Column2" Width="300"/>  </Grid.ColumnDefinitions>  <StackPanel>  <StackPanel Orientation="Horizontal" Margin="10">  <Border BorderBrush="Black" BorderThickness="2">  <TextBlock x:Name="ViewFieldTB" Text="" FontSize="30" Margin="10" Width="240"/>  </Border>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="5, 0">  <Button Content="7" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click7" Background="HotPink"/>  <Button Content="8" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click8" Background="HotPink"/>  <Button Content="9" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click9" Background="HotPink"/>  <CheckBox x:Name="SaveHistoryCBToHide" IsChecked="True" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="5"/>  <TextBlock x:Name="CBTextToHide" Text="Вкл/Выкл запись истории" TextWrapping="Wrap" Width="60" VerticalAlignment="Center"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="5, 0">  <Button Content="4" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click4" Background="HotPink"/>  <Button Content="5" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click5" Background="HotPink"/>  <Button Content="6" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click6" Background="HotPink"/>  <Button Content="C" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_ClickClear"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="5, 0">  <Button Content="1" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click1" Background="HotPink"/>  <Button Content="2" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click2" Background="HotPink"/>  <Button Content="3" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_Click3" Background="HotPink"/>  <RadioButton GroupName="HistoryView" VerticalAlignment="Center" Margin="10" Checked="RadioButton\_CheckedShow"/>  <TextBlock Text="Отображать историю" TextWrapping="Wrap" Width="70" VerticalAlignment="Center"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="5, 0">  <Button Content="0" Width="170" Height="50" FontSize="20" Margin="5" Click="Button\_Click0"/>  <RadioButton GroupName="HistoryView" VerticalAlignment="Center" Margin="10" Checked="RadioButton\_CheckedHide"/>  <TextBlock Text="Скрыть историю" TextWrapping="Wrap" Width="70" VerticalAlignment="Center"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="10">  <Button Content="+" x:Name="PlusButton" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_ClickPlus"/>  <Button Content="-" x:Name="MinusButton" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_ClickMinus"/>  <Button Content="\*" x:Name="MultiplyButton" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_ClickMultiply"/>  <Button Content="/" x:Name="DivideButton" Width="50" Height="50" FontSize="20" Margin="5" Click="Button\_ClickDivide"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="10">  <Button Content="=" x:Name="EqualButton" Width="150" Height="50" FontSize="20" Margin="5" Click="Button\_ClickEqual"/>  </StackPanel>  </StackPanel>  <StackPanel Grid.Column="1" x:Name="HistoryToHide">  <Grid>  <Button Content="Очистить историю" Width="130" Height="40" Margin="10" HorizontalAlignment="Left" Click="Button\_ClickClearHistory"/>  <Button Content="Удалить посл. эл-т" Width="130" Height="40" Margin="10" HorizontalAlignment="Right" Click="Button\_ClickRemoveLastItem"/>  </Grid>  <ListBox Margin="10" ItemsSource="{Binding HistoryList}" Height="400">  <ListBox.ItemTemplate>  <DataTemplate>  <TextBlock Text="{Binding Equasion}" />  </DataTemplate>  </ListBox.ItemTemplate>  </ListBox>  </StackPanel>  </Grid>  </Window> <StackPanel>  <Button Content="=" x:Name="EqualButton" Width="150" Height="50" FontSize="20" Margin="5" Click="Button\_ClickEqual"/>  </StackPanel>  </StackPanel>  </Window> |

|  |
| --- |
| using System.Collections.ObjectModel;  using System.Windows;  namespace \_3.CalculatorPlus  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  bool isRightSide = false;  bool isPlus = false;  bool isMinus = false;  bool isMultiply = false;  bool isDivide = false;  public ObservableCollection<CalculatorHistory> HistoryList { get; set; }  float Result  {  get  {  if (float.TryParse(leftPart, out float left) && float.TryParse(rightPart, out float right))  {  if (isPlus) return left + right;  if (isMinus) return left - right;  if (isMultiply) return left \* right;  if (isDivide) return left / right;  }  return 0;  }  }  string leftPart = "";  string rightPart = "";  public MainWindow()  {  InitializeComponent();  DataContext = this;  HistoryList = [];  }  private void Button\_Click0(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "0";  if (isRightSide) rightPart += "0";  else leftPart += "0";  }  private void Button\_Click1(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "1";  if (isRightSide) rightPart += "1";  else leftPart += "1";  }  private void Button\_Click2(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "2";  if (isRightSide) rightPart += "2";  else leftPart += "2";  }  private void Button\_Click3(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "3";  if (isRightSide) rightPart += "3";  else leftPart += "3";  }  private void Button\_Click4(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "4";  if (isRightSide) rightPart += "4";  else leftPart += "4";  }  private void Button\_Click5(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "5";  if (isRightSide) rightPart += "5";  else leftPart += "5";  }  private void Button\_Click6(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "6";  if (isRightSide) rightPart += "6";  else leftPart += "6";  }  private void Button\_Click7(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "7";  if (isRightSide) rightPart += "7";  else leftPart += "7";  }  private void Button\_Click8(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "8";  if (isRightSide) rightPart += "8";  else leftPart += "8";  }  private void Button\_Click9(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text += "9";  if (isRightSide) rightPart += "9";  else leftPart += "9";  }  private void Button\_ClickPlus(object sender, RoutedEventArgs e)  {  if (!isRightSide)  {  isRightSide = true;  leftPart = ViewFieldTB.Text;  }  else  {  leftPart = Result.ToString();  rightPart = "";  }  ViewFieldTB.Text += "+";  isPlus = true;  isMinus = false;  isMultiply = false;  isDivide = false;  SetButtonsInactive();  }  private void Button\_ClickMinus(object sender, RoutedEventArgs e)  {  if (!isRightSide)  {  isRightSide = true;  leftPart = ViewFieldTB.Text;  }  else  {  leftPart = Result.ToString();  rightPart = "";  }  ViewFieldTB.Text += "-";  isPlus = false;  isMinus = true;  isMultiply = false;  isDivide = false;  SetButtonsInactive();  }  private void Button\_ClickMultiply(object sender, RoutedEventArgs e)  {  if (!isRightSide)  {  isRightSide = true;  leftPart = ViewFieldTB.Text;  }  else  {  leftPart = Result.ToString();  rightPart = "";  }  ViewFieldTB.Text += "\*";  isPlus = false;  isMinus = false;  isMultiply = true;  isDivide = false;  SetButtonsInactive();  }  private void Button\_ClickDivide(object sender, RoutedEventArgs e)  {  if (!isRightSide)  {  isRightSide = true;  leftPart = ViewFieldTB.Text;  }  else  {  leftPart = Result.ToString();  rightPart = "";  }  ViewFieldTB.Text += "/";  isPlus = false;  isMinus = false;  isMultiply = false;  isDivide = true;  SetButtonsInactive();  }  private void Button\_ClickEqual(object sender, RoutedEventArgs e)  {  if (SaveHistoryCBToHide.IsChecked.Value) AddToHistory();  ViewFieldTB.Text = Result.ToString();  leftPart = Result.ToString();  rightPart = "";  isRightSide = false;  SetButtonsActive();  isPlus = false;  isMinus = false;  isMultiply = false;  isDivide = false;  }  private void Button\_ClickClear(object sender, RoutedEventArgs e)  {  ViewFieldTB.Text = "";  leftPart = "";  rightPart = "";  isRightSide = false;  isPlus = false;  isMinus = false;  isMultiply = false;  isDivide = false;  SetButtonsActive();  }  private void SetButtonsInactive()  {  PlusButton.IsEnabled = false;  MinusButton.IsEnabled = false;  MultiplyButton.IsEnabled = false;  DivideButton.IsEnabled = false;  }  private void SetButtonsActive()  {  PlusButton.IsEnabled = true;  MinusButton.IsEnabled = true;  MultiplyButton.IsEnabled = true;  DivideButton.IsEnabled = true;  }  private void AddToHistory()  {  if (isPlus) HistoryList.Add(new CalculatorHistory { Equasion = $"{leftPart} + {rightPart} = {Result}" });  if (isMinus) HistoryList.Add(new CalculatorHistory { Equasion = $"{leftPart} - {rightPart} = {Result}" });  if (isMultiply) HistoryList.Add(new CalculatorHistory { Equasion = $"{leftPart} \* {rightPart} = {Result}" });  if (isDivide) HistoryList.Add(new CalculatorHistory { Equasion = $"{leftPart} / {rightPart} = {Result}" });  }  private void Button\_ClickClearHistory(object sender, RoutedEventArgs e)  {  HistoryList.Clear();  }  private void Button\_ClickRemoveLastItem(object sender, RoutedEventArgs e)  {  if (HistoryList.Count == 0) return;  HistoryList.RemoveAt(HistoryList.Count - 1);  }  private void RadioButton\_CheckedHide(object sender, RoutedEventArgs e)  {  HistoryToHide.Visibility = Visibility.Hidden;  SaveHistoryCBToHide.Visibility = Visibility.Hidden;  CBTextToHide.Visibility = Visibility.Hidden;    Column1.Width = new GridLength(300);  Column2.Width = new GridLength(0);  Width = 350;  }  private void RadioButton\_CheckedShow(object sender, RoutedEventArgs e)  {  HistoryToHide.Visibility = Visibility.Visible;  SaveHistoryCBToHide.Visibility = Visibility.Visible;  CBTextToHide.Visibility = Visibility.Visible;  Column1.Width = new GridLength(300);  Column2.Width = new GridLength(300);  Width = 640;  }  }  }  namespace \_3.CalculatorPlus  {  public class CalculatorHistory  {  public string Equasion { get; set; }  }  } |



