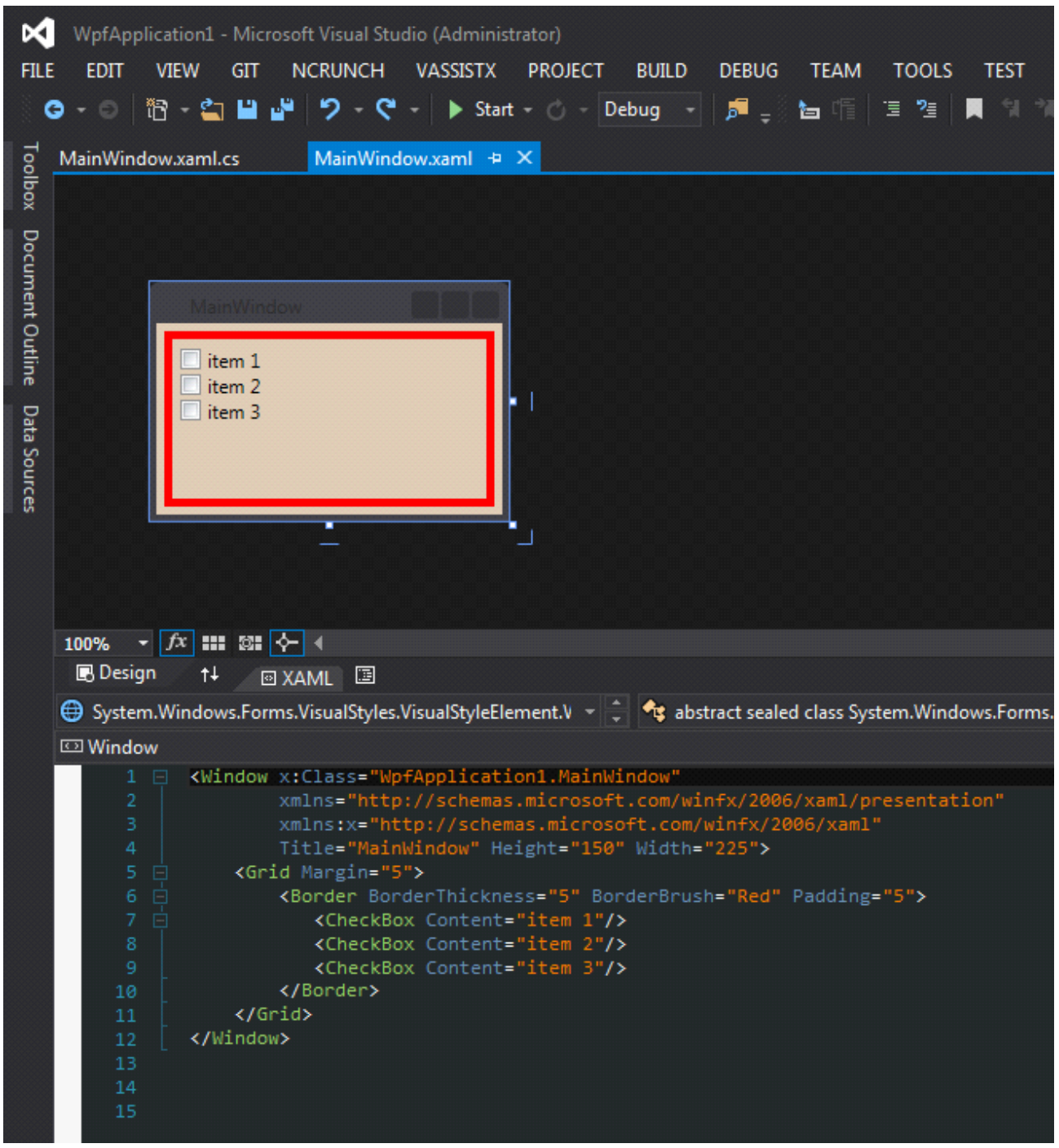
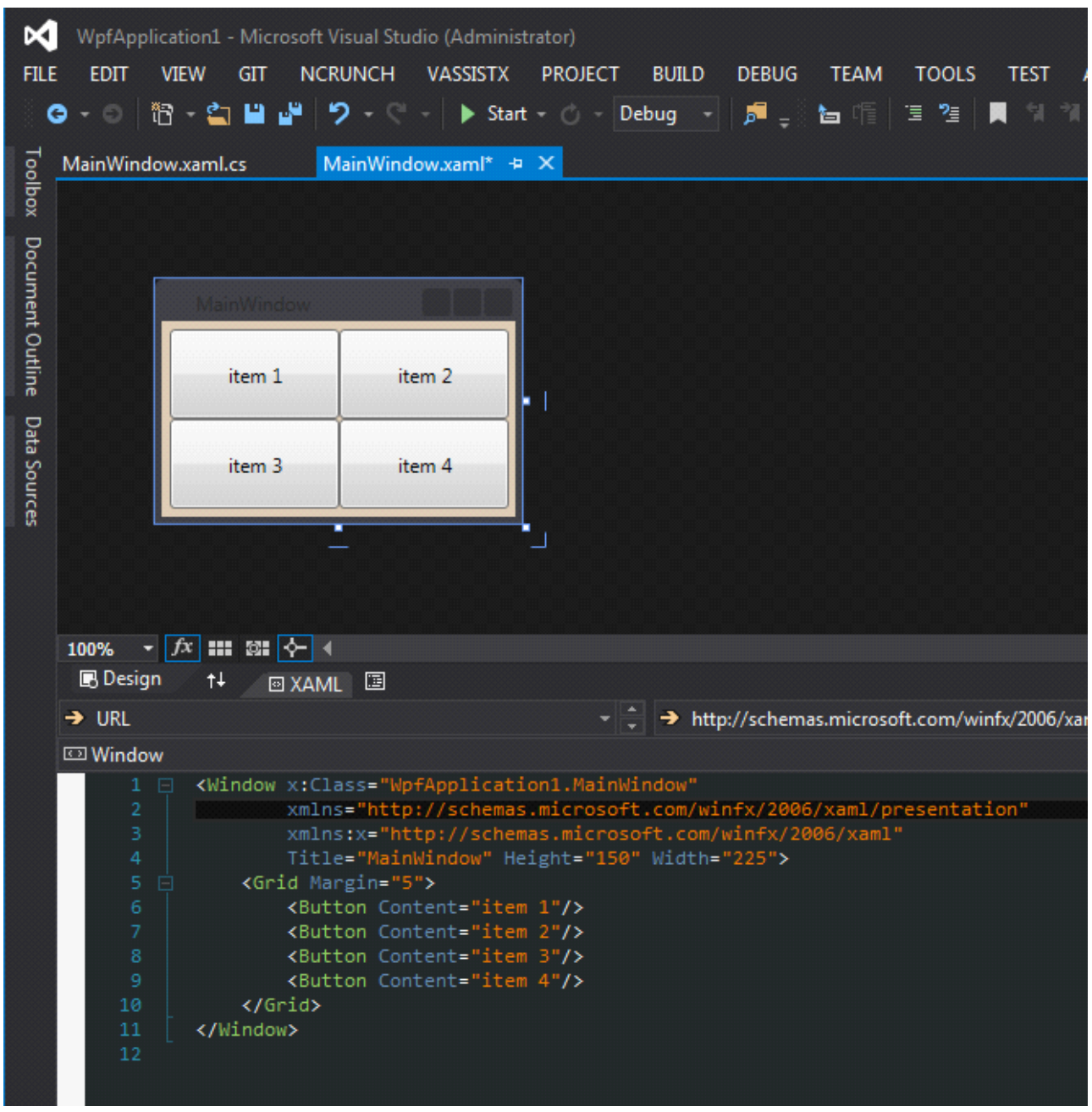
Jump Trading - WPF Test

The estimated time to complete this test is 2 hours, although you may complete it at your own pace.

Question 1: Below is a series of images, with each image containing a mismatch between the UI shown in the preview pane, and the XAML code. For each image, please explain why the XAML would not actually produce the UI as shown, and then provide a version of the XAML which does work.

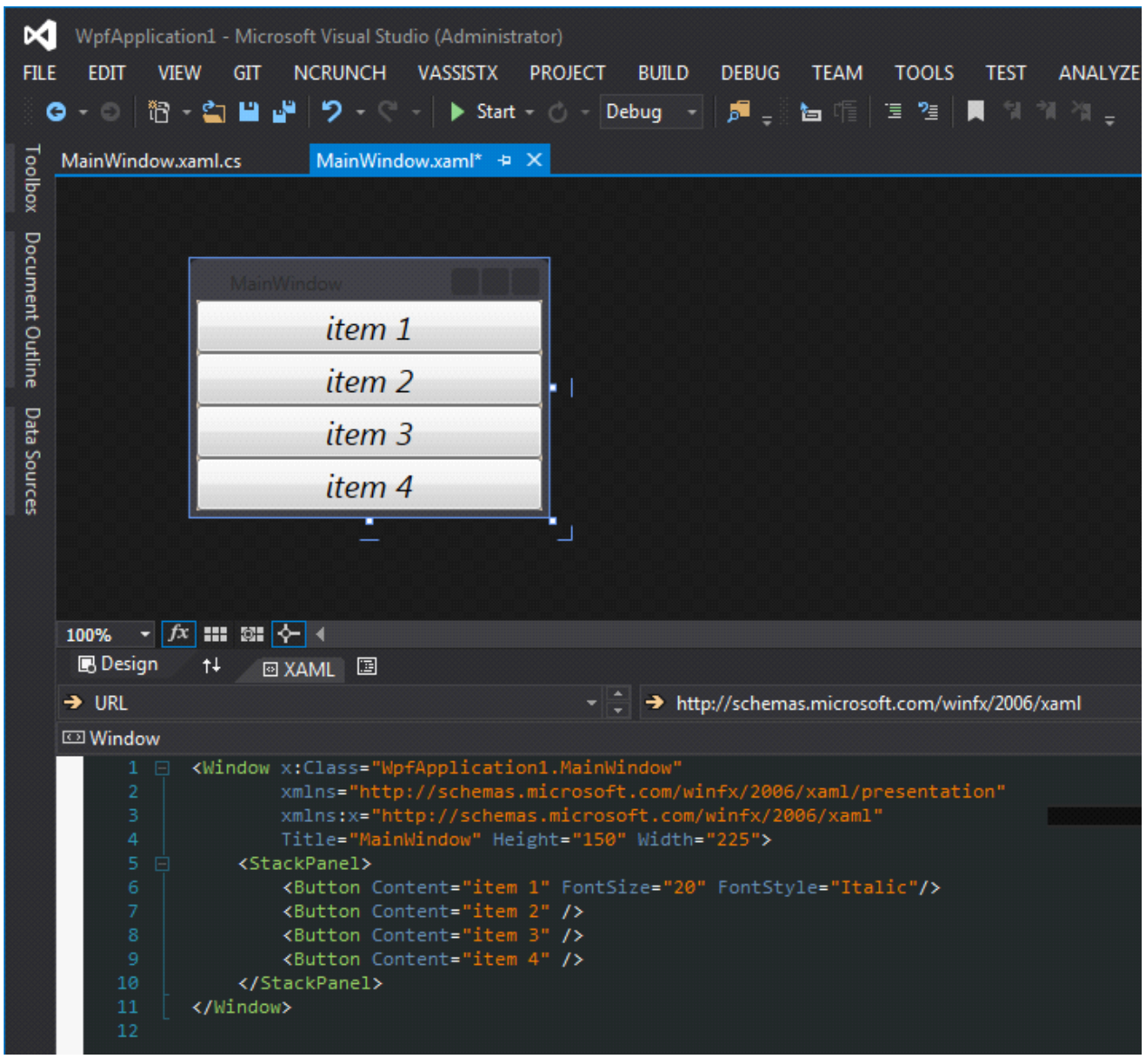


(Question 1A)

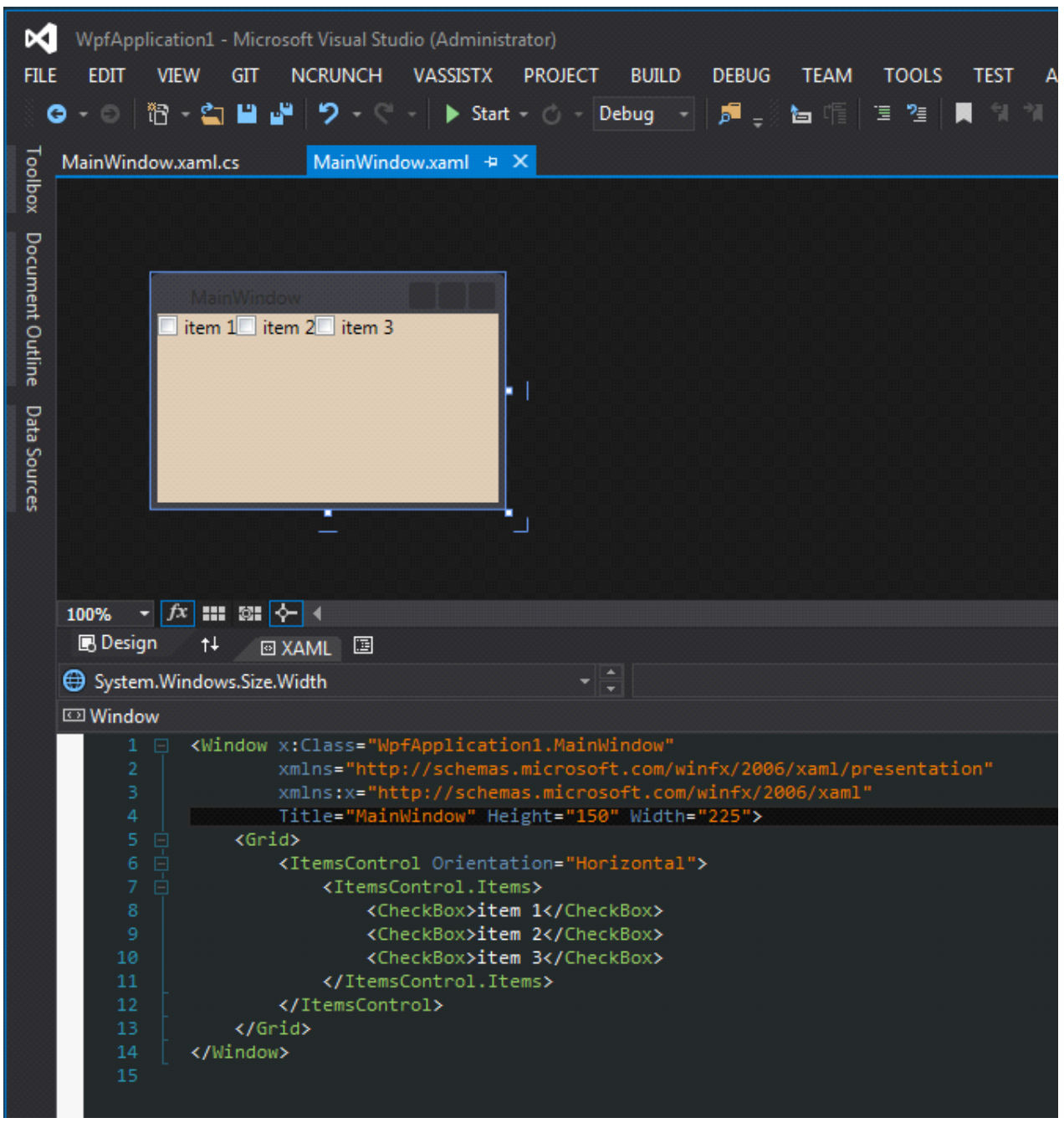


(Question 1B)

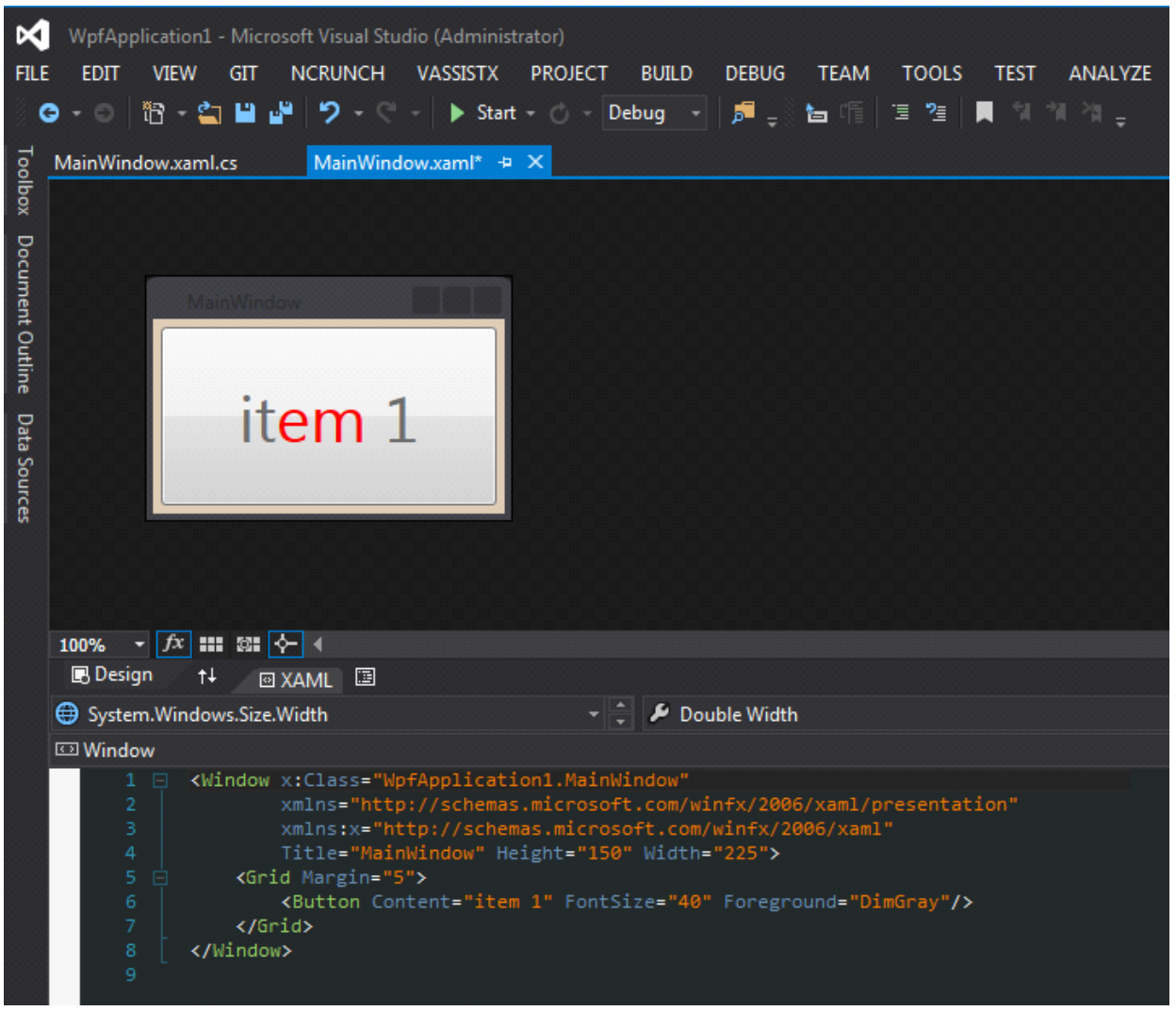
(Question 1C)



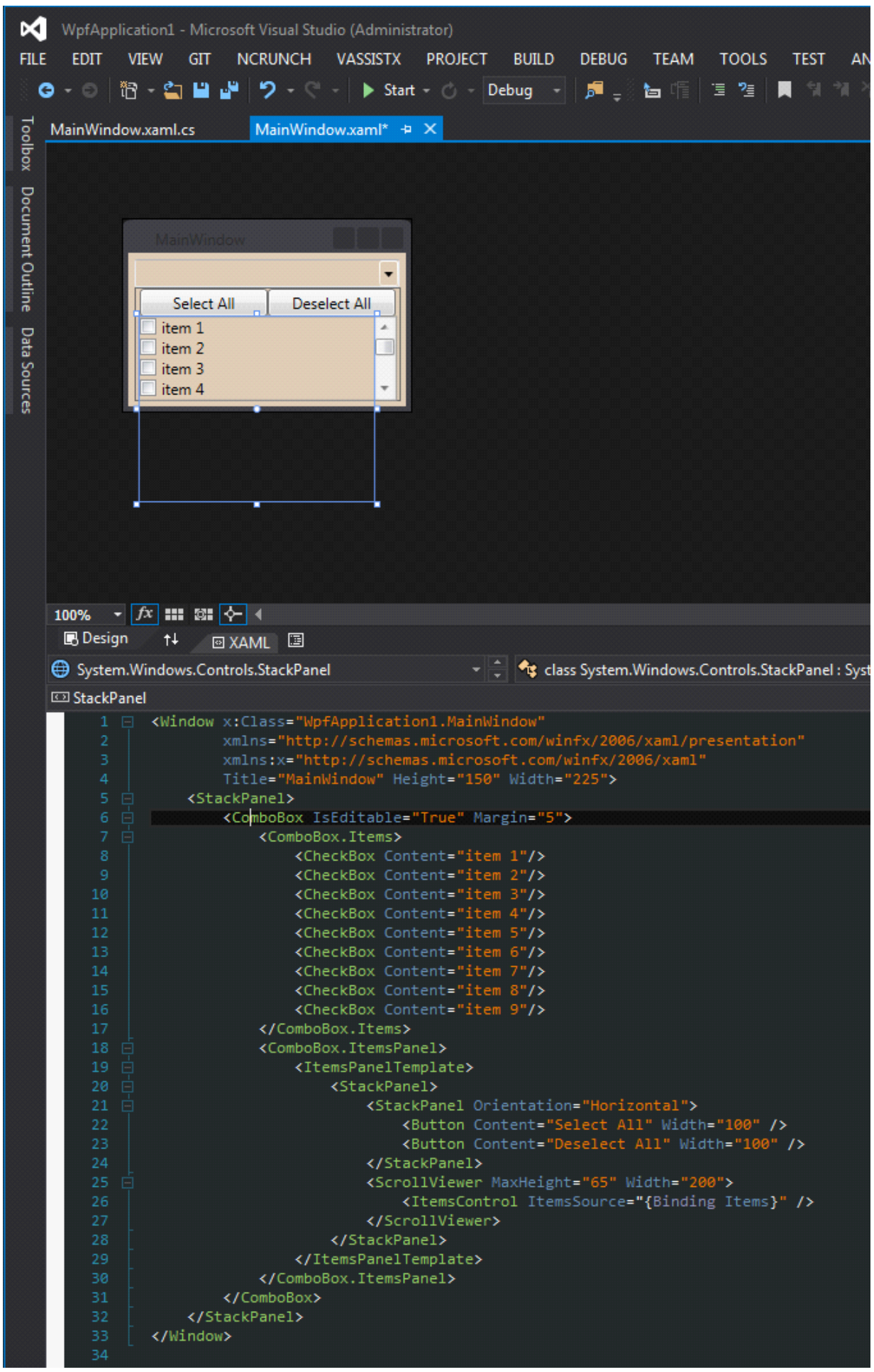
(Question 1D)



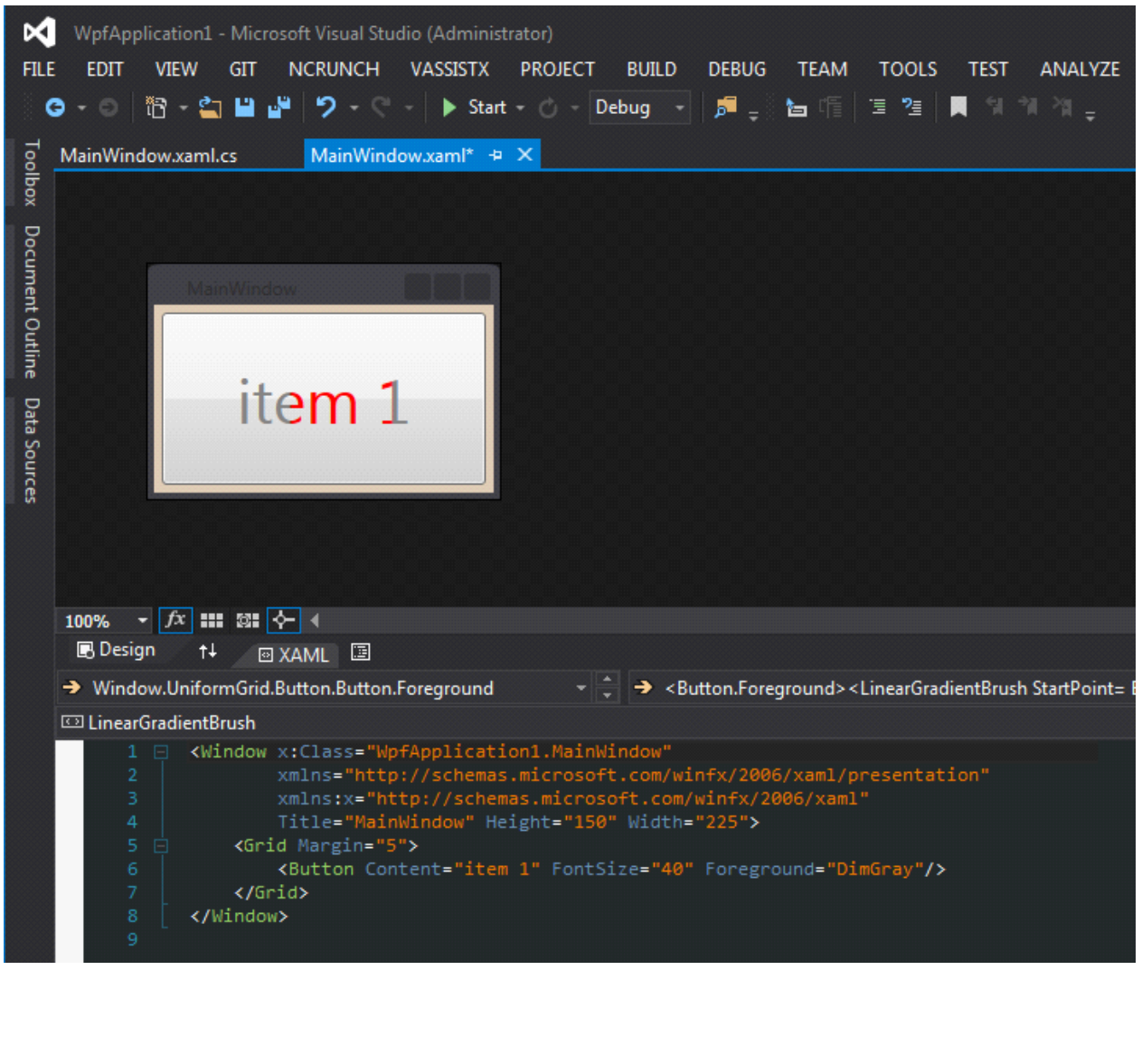
(Question 1E)



(Question 1F)



(Question 1G)


WpfApplication1 - Microsoft Visual Studio (Administrator)
FILE EDIT VIEW GIT NCRUNCH VASSISTX PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE
Start
Debug
Toolbox
Document Outline
Data Sources
MainWindow.xaml.cs
MainWindow.xaml*
MainWindow
item 1
100%
Design
XAML
Window.UniformGrid.Button.Button.Foreground
<Button.Foreground><LinearGradientBrush StartPoint=
LinearGradientBrush
<Window x:Class="WpfApplication1.MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
Title="MainWindow" Height="150" Width="225">
<Grid Margin="5">
<Button Content="item 1" FontSize="40" Foreground="DimGray"/>
</Grid>
</Window>

Question 2: We have defined a style for a button, which we call “Button.Dark”. This name has been defined using x:Key=”Button.Dark” in one of our resource dictionaries. In this, x has been defined as the namespace for XAML. In one of our user controls we have merged the dictionary in a ResourceDictionary in our UserControl’s resources.

Our goal is to use the Button.Dark style in our UserControl for all the buttons apart from the ones (below) that specify something different in the comment. What do we need to change to achieve this? Feel free to provide more than one solution if you believe that different solutions can benefit us in different ways.

app.xaml

```
<Application x:Class="OurApplication.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             StartupUri="MainWindow.xaml">
    <Application.Resources>
        <Style TargetType="Button">
            ...
        </Style>
    </Application.Resources>
</Application>
```

User Control

```
<UserControl
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
    <UserControl.Resources>
        <ResourceDictionary>
            <ResourceDictionary.MergedDictionaries>
                <ResourceDictionary Source="pack://application:,,,/OurApplication;component/ButtonStyles.xaml"/>
            </ResourceDictionary.MergedDictionaries>

            <!--<Style TargetType="Button" BasedOn="{StaticResource Button.Dark}"></Style>-->
        </ResourceDictionary>
    </UserControl.Resources>
    <StackPanel>
        <!-- This button needs to be styled according to the style in the app.xaml -->
        <Button />

        <Button />
        <Button />

        <!-- This button needs to be styled according to -->
        <!-- a new style that you will have to create -->
        <!-- The style should set the button's background to gray -->
        <Button />
    </StackPanel>
</UserControl>
```

The app.xaml file should not be changed.

Question 3: The following is a DataGrid with two columns and is part of a DataTemplate. Our intention is to be able to hide the second column when the IsDetailsModelEnabled property is false. Assume that the DataContext of the (top level) Grid has been set to an object that contains the IsDetailsModelEnabled property and collection of Items (called Items). Each item in the collection contains a Name and a ShortDescription (both string properties). Can you spot any mistakes in the xaml below? If so, please provide a description of what the mistake is and also provide a solution. Is there anything else that we can do to simplify the this code?

```
    <Grid>
        <DataGrid AutoGenerateColumns="False" ItemsSource="{Binding Items}">
            <DataGrid.Columns>
                <DataGridTemplateColumn>
                    <DataGridTemplateColumn.CellTemplate>
                        <DataTemplate>
                            <TextBlock Text="{Binding Name}"/>
                        </DataTemplate>
                    </DataGridTemplateColumn.CellTemplate>
                </DataGridTemplateColumn>
                <DataGridTemplateColumn Visibility="{Binding IsDetailsModeEnabled, RelativeSource={RelativeSource FindAncestor,
AncestorType=DataGrid}}">
                    <DataGridTemplateColumn.CellTemplate>
                        <DataTemplate>
                            <StackPanel>
                                <TextBlock Text="{Binding ShortDescription}" />
                            </StackPanel>
                        </DataTemplate>
                    </DataGridTemplateColumn.CellTemplate>
                </DataGridTemplateColumn>
            </DataGrid.Columns>
        </DataGrid>
    </Grid>
```