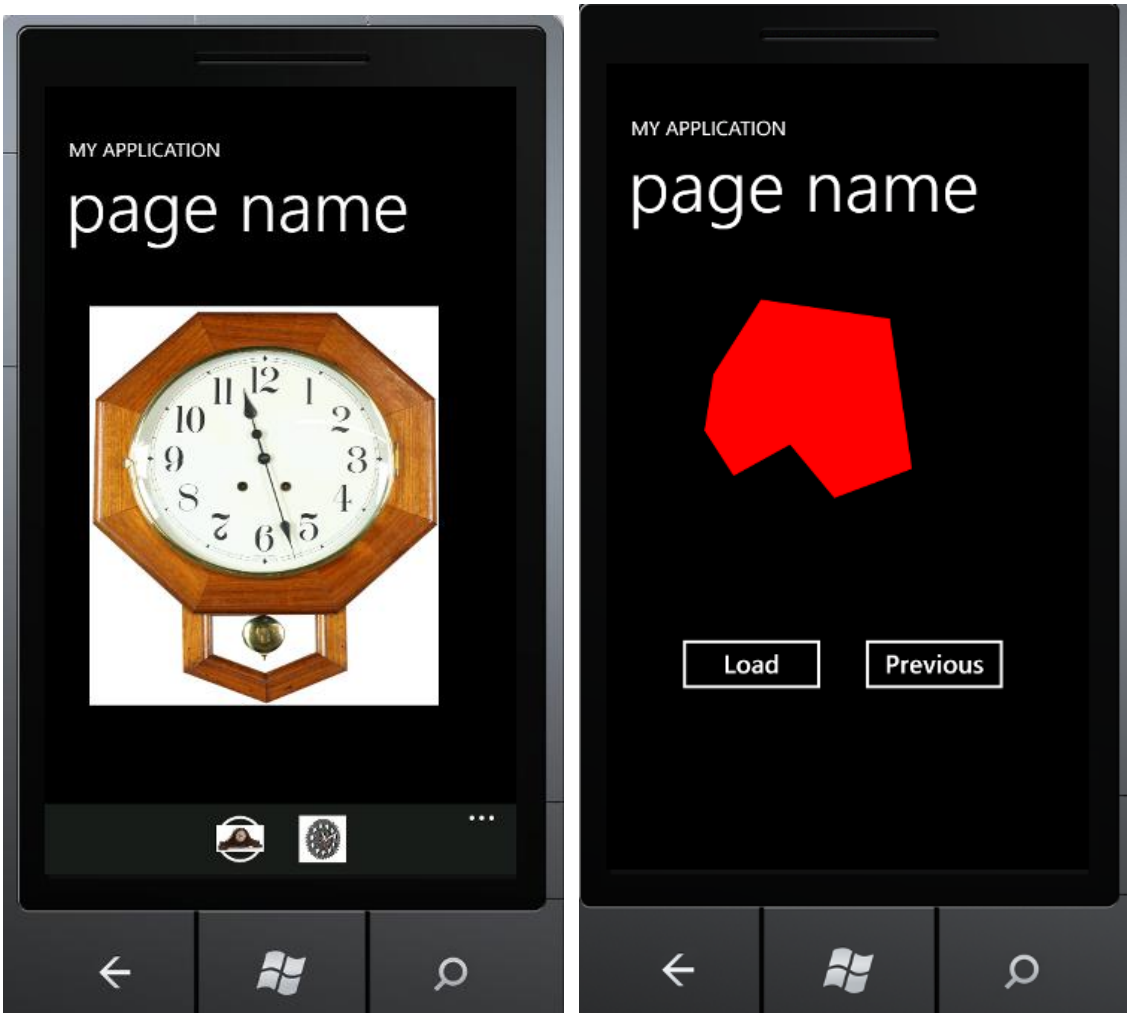
Задача.
Да се създаде **Windows Phone** приложение, което да включва 2 страници. В първата страница на приложението да се изведе изображение. При всяко докосване на екрана в рамките на изображението да се прихващат координатите (x,y) на точката на докосване и да се съхраняват като обект от клас **Point** в колекция (клас **ObservableCollection<Type>**). Първата страница на приложението да включва два бутона-икони в **ApplicationBar**; единият бутон-икона да служи за запазване на съдържанието на *колекцията* от обекти **Point** в паметта за *settings* на телефона, а вторият бутон-икона – за преход към втора страница на приложението. Във втората страница на приложението да се включат два бутона. При кликване върху първия бутон, да се прочита съдържанието на паметта на приложението за *settings* и да се изчертава запълнен с посочен цвят **Polygon**, чиито върхове са обектите **Point**, прочетени от *settings*. При кликване върху втория бутон във втора страница на приложението да се осъществява преход към първата страница на приложението.

Решението включва:

- Скициране на дизайна на екрана на мобилния (разположение на контролите) и работа с приложението;
- Структура на приложението - frame, pages, Grid, StackPanel и т.н. в каква йерархична връзка са;
- Планиране на ресурсите - .jpg файлове, .png файлове, текстови файлове;
- Планиране на събитията - брой, вид;
- Разпределение на кода между XAML и C#, както и между страниците на приложението;
- Формализирано описание (примерно на псевдо-код) на операциите вътре в методите за обработка на отделните събития.

В решението по-долу е приложен код на XAML и на C# единствено с илюстративна цел, т.е. не е необходимо да пишете код на изпита, но ако можете и/или Ви е по-удобно – пишете код дори да е непълен.

Решение.
1. Две страници на приложението. В първата страница се съдържа контрол **Image** и **ApplicationBar** с два бутона-икони: бутон-икона Save и бутон-икона NextPage. При всяко докосване на изображението се създава обект от клас **Point**, като се използват x,y координатите на точката на докосване. Обектът от клас **Point** се добавя към колекция от обекти от този вид. Когато се натисне бутон-икона Save съдържанието на колекцията от обекти от клас **Point** се запазва в паметта за settings на телефона. Когато се натисне бутон-икона NextPage се преминава към втока страница на приложението. Втората страница на приложението съдържа елемент **Polygon** и два контрола **Button**: бутон Load и бутон Previous. Когато се натисне бутон Load от паметта за settings на телефона се прочитат обектите от клас **Point** и всеки от тях се добавя към колекцията от върхове (свойство **Points** на обект **Polygon**) на полигона, като той се изчертава на екрана. Когато се натисне бутон Previous се осъществява връщане към първа страница на приложението. **Общ вид (скицират се на ръка):**



2. Структура на приложението. Един frame, две Page. Първата Page – **MainPage.xaml** включва контрол **Grid**, в който е разположен един контрол **Image** и **ApplicationBar** , който включва два бутона-икони – за преход към втора страница и за запазване в паметта за settings на телефона на колекция от точките, в които е докоснато изображението. Към приложението добавям нова страница – команда **Add/New Item/Windows Phone Portrait Page** като променям името й на **"second.xaml**". Втората страница на приложението има контрол **Grid**, в който са включени контрол **Polygon** и два контрола **Button**.

3. Ресурси на приложението. Към приложението се добавят един **.jpg** файл (изображението, което се извежда в контрол **Image**) и два .**png** файла (за двата бутона-икони от **ApplicationBar**) с команда **Add/Existing Item.** За всеки графичен файл се установяват свойства **Build Action = Content** и **Copy To Output Directory = Copy if newer**.

4. Използва се събитие **ManipulationStarted** за контрол **Image** и събитие **Click** за двата бутона-икони от **ApplicationBar** в първа страница и за двата контрола **Button** във втора страница.

5. В кода на XAML: Към първата страница на приложението добавям контрол **Image,** за който установявам свойство **Source = „xxx.jpg"** и добавям метод за обслужване на събитието **ManipulationStarted**. Премахвам коментара в XAML кода за **ApplicationBar**

и оставям само кода за два бутона-икони. В XAML установявам за всеки от двата бутони икони свойства **IconUri** да сочат към съответните изображения, свойства **Text=”Save”** и **Text=”NextPage”,** и добавям метод за обработка на събитие **Click** за всеки от двата бутона-икона.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <Image Height="405" Source="clock6.jpg"
ManipulationStarted="image1_ManipulationStarted" HorizontalAlignment="Left"
Margin="33,30,0,0" Name="image1" Stretch="Fill" VerticalAlignment="Top" Width="355" />
        </Grid>
    </Grid>

    <!--Sample code showing usage of ApplicationBar-->
    <phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="True">
            <shell:ApplicationBarIconButton IconUri="clock8.jpg" Text="SaveContour"
Click="ApplicationBarIconButton_Click"/>
            <shell:ApplicationBarIconButton IconUri="clock5.jpg" Text="GoToNextPage"
Click="ApplicationBarIconButton_Click_1"/>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>

</phone:PhoneApplicationPage>
```

В кода на C# за първа страница добавям пространства на имената `System.IO.IsolatedStorage; Syatem.IO, System.Collections.ObjectModel.` Към класа **MainPage** добавям **private** поле данни list от тип **ObservableCollection<Point>.** В конструктора на класа **MainPage** инициализирам полето данни **list** с обект от клас **ObservableCollection<Point>** като извиквам конструктора без параметри.
В метода за обслужване на събитието **ManipulationStarted** за контрол **Image** създавам обект от клас **Point** и го добавям към колекцията **list** (от втория параметър на метода за обслужване на събитието **ManipulationStarted** извличам **x** и **y** координатите на докосване на екрана и ги използвам за създаване на обекта от клас **Point).**
В метода за обслужване на събитието **Click** за бутон-икона ”NextPage”, извиквам статичен метод **Navigate** за класа **NavigationService** и подавам като параметър на метода обект от клас **Uri** с името на втората страница на приложението “second.xaml”.
В метода за обслужване на събитието **Click** за бутон-икона “Save” създавам референция към паметта на телефона за settings (**IsolatedStorageSettings**) и с цикъл **foreach** за всеки обект от клас **Point** в колекцията list записвам двойка „име на точка, обект Point” в паметта на телефона; извън цикъла извиквам метод **Save**() за референцията.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using System.Collections.ObjectModel;
using System.IO.IsolatedStorage;

namespace exam
{
```

```
    public partial class MainPage : PhoneApplicationPage
    {
        // Constructor
        private ObservableCollection<Point> contour;

        public MainPage()
        {
            InitializeComponent();
            contour = new ObservableCollection<Point>();
        }

        private void image1_ManipulationStarted(object sender,
ManipulationStartedEventArgs e)
        {
            contour.Add(new Point(e.ManipulationOrigin.X, e.ManipulationOrigin.Y));
        }

        private void ApplicationBarIconButton_Click(object sender, EventArgs e)
        {
            string s ="P";
            int i = 0;
            IsolatedStorageSettings sett =
IsolatedStorageSettings.ApplicationSettings;
            foreach (Point pt in contour)
            {
                sett.Add(s+i.ToString(), pt);
                i++;
            }
            sett.Save();
        }

        private void ApplicationBarIconButton_Click_1(object sender, EventArgs e)
        {
            NavigationService.Navigate(new Uri("/second.xaml", UriKind.Relative));
        }
    }
}
```

В XAML кода за втората страница, наречена **second.xaml,** добавям два бутона за които установявам свойства **Name = “previous**”; **Text=”Previous”; Name=”load**”; **Text = “Load”** и добавям методи за обработка на събитието **Click**. В кода на XAML създавам елемент **Polygon**, за който установявам свойство **Fill=”Red**” и свойство **Name=”pl”,** за да мога да се обръщам към този елемент от кода на C#.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <Button Content="Load" Height="72" Click="button1_Click"
HorizontalAlignment="Left" Margin="52,367,0,0" Name="button1" VerticalAlignment="Top"
Width="160" />
            <Button Content="Previous" Click="button2_Click" Height="72"
HorizontalAlignment="Left" Margin="233,367,0,0" Name="button2" VerticalAlignment="Top"
Width="160" />
            <Polygon Name="pl" Fill="Red"/>
        </Grid>
```

В кода на C# в second.xaml.cs в метода за обслужване на събитието **Click** за бутон ” Previous” извиквам метод **GoBack()** на класа **NavigationService**.
В метода за обслужване на събитието **Click** за бутон ”Load” създавам референция към паметта на телефона за settings. Проверявам дали в паметта на телефона за settings има данни и ако да – чета данните и за обекта **Polygon** с име pl добавям върхове като за целта използвам метод **Add** на свойство **Points** на обекта **Polygon**.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using System.IO.IsolatedStorage;

namespace exam
{
    public partial class second : PhoneApplicationPage
    {
        public second()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, RoutedEventArgs e)
        {
            IsolatedStorageSettings sett =
IsolatedStorageSettings.ApplicationSettings;
            if (sett.Values.Count > 0)
            {
                foreach(Point p in sett.Values)
                {
                    pl.Points.Add(p);
                }
            }
        }

        private void button1_Click(object sender, RoutedEventArgs e)
        {
            NavigationService.GoBack();
        }
    }
}
```