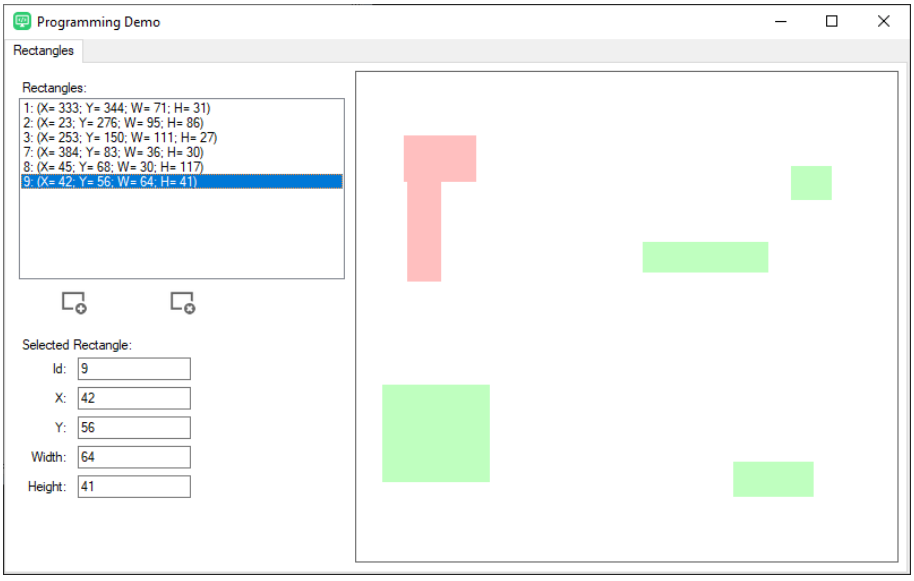
Задание №5.
Коллекции

Цель задания:

Освоить использование списков в C# на основе разработки программы, определяющей пересечение прямоугольников.

Макет приложения:



Приложение предоставляет следующую функциональность:

1) Добавление и удаление новых объектов прямоугольников (ранее созданный класс Rectangle).
2) Отображение всех прямоугольников на отдельной панели ("канва", элемент Panel).
3) Определение пересекающихся прямоугольников – если прямоугольник пересекается с любым другим прямоугольником, то он должен отображаться красным цветом, иначе – зеленым.

4) Возможность редактирования прямоугольников. При изменении положения или размеров прямоугольника, его отображение на канве должно тут же обновляться, включая проверку на пересечение с другими прямоугольниками.
5) Валидация вводимых пользователем значений.

Последовательность выполнения:

1. Создать в ранее созданном проекте Programming Demo новую вкладку Rectangles, согласно макету. Особенности макета:
   - Кнопки для добавления и удаления прямоугольников должны быть выполнены в плоском стиле.
   - Правая часть вкладки (канва) реализована как элемент Panel, с установкой свойства BorderStyle в значение FixedSingle.
   - Верстка должна быть адаптивной.
2. Дать всем элементам новой вкладки правильные названия, согласно требованиям к именованию.
3. Ранее созданные вкладки Enums и Classes должны остаться в приложении.
4. Создать в главном окне поля _rectangles и _currentRectangle типа List<Rectangle> и Rectangle соответственно.
5. Реализовать логику кнопки "Добавить прямоугольник". В обработчике кнопки должен создаваться новый экземпляр прямоугольника, который будет добавляться в список _rectangles.
6. Все прямоугольники списка _rectangles должны отображаться в списке ListBox на вкладке, при этом должен соблюдаться формат вывода строки – в строке должны отображаться Id, положение и размеры прямоугольника так, как это показано на макете.
7. Для удобства дальнейшей отладки и тестирования предлагается создавать новые экземпляры не с нулевыми, а со случайными значениями координат и размеров.
8. Реализовать логику кнопки "Удалить прямоугольник". В обработчике кнопки должен определяться текущий выбранный в списке прямоугольник и удаляться как из списка _rectangles, так и из списка ListBox. Если в ListBox ничего не выбрано или список пуст, то при нажатии на кнопку ничего не происходит.
9. При выборе прямоугольника в списке, все его данные должны отображаться в соответствующих текстовых полях под ListBox. Текущий выбранный прямоугольник должен сохраняться в поле _currentRectangle. Если в списке не

выбран ни один прямоугольник или список пуст, текстовые поля должны очищаться.

10. Реализовать возможность редактирования данных прямоугольника – при изменении значений положения или размеров прямоугольника, строка в ListBox должна сразу же обновляться.

11. Все текстовые поля должны реализовывать валидацию согласно ограничениям на поля в классе Rectangle и Point2D.

12. Поле Id остается полем доступным только для чтения.

13. Если нет выбранного прямоугольника или нет прямоугольников в списке, то все текстовые поля должны быть пустыми и нормального цвета.

14. Теперь необходимо реализовать отображение прямоугольников на канве.

15. Один из подходов для отображения прямоугольников на панели является программное добавление новых экземпляров Panel на панель канвы.

16. Создайте в главном окне еще одно поле _rectanglePanels типа List<Panel>.

17. Добавьте в обработчик кнопки "Добавить прямоугольник" код, который будет создавать переменную типа Panel. Свойства размеров и положения созданной панели (Location, Width, Height) должны инициализироваться значениями прямоугольника Rectangle, который создается в этом же обработчике. Также назначьте панели Panel цвет фона Color.FromArgb(127, 127, 255, 127).

18. Для того, чтобы новая панель отрисовалась на канве, необходимо добавить её на канву. Примерный код:

```
CanvasPanel.Controls.Add(panel);
```

Аналогичным способом можно добавлять любые элементы управления на панель, GroupBox или иной контейнер.

19. Запустите и убедитесь, что при добавлении новых прямоугольников в список, на канве отрисовываются панели зеленого цвета. Если панели не отрисовываются, найдите ошибку и исправьте.

20. Реализуйте логику удаления панелей с канвы при удалении прямоугольников из списка. Примерный код для удаления:

```
// Вариант удаления панели по объекту
CanvasPanel.Controls.Remove(panel);

// Вариант удаления панели по индексу
CanvasPanel.Controls.RemoveAt(selectedIndex);
```

21. Не забудьте удалить панель также и из списка _rectanglePanels.

22. Теперь реализуйте логику индикации прямоугольников, которые пересекаются. Если два прямоугольника пересекаются (накладываются друг на друга на канве), они должны отображаться красным цветом Color.FromArgb(127, 255, 127, 127). Для этого используйте ранее созданный класс CollisionManager.

23. Для индикации пересечений создайте отдельный закрытый метод FindCollisions() в главном окне. Логика метода должна быть следующей:

- перекрасить цвет всех панелей в зеленый;
- перебрать все прямоугольники в двойном цикле;
- на каждой итерации должно происходить сравнение i-го и j-го прямоугольников с помощью метода CollisionManager.IsCollision();
- если прямоугольники пересекаются, то цвет соответствующих им панелей необходимо поменять на красный;
- если прямоугольник сравнивается сам с собой, то перекрашивать прямоугольник не надо.

24. Запустите и убедитесь, что индикация пересечения прямоугольников работает. Если панели не меняют цвет или меняют цвет неправильно, найдите ошибку и исправьте.

25. Метод FindCollisions() должен вызываться каждый раз, когда происходит добавление или удаление прямоугольников из списке _rectangles.

26. Теперь необходимо реализовать перерисовку прямоугольника при его редактировании.

27. Если пользователь задаст выбранному прямоугольнику другие значения положения или размеры, то соответствующая прямоугольнику панель на канве также должна поменять своё положение и размер.

28. Положение и размер панели на канве меняется только, если пользователь задал корректные значения для полей прямоугольника.

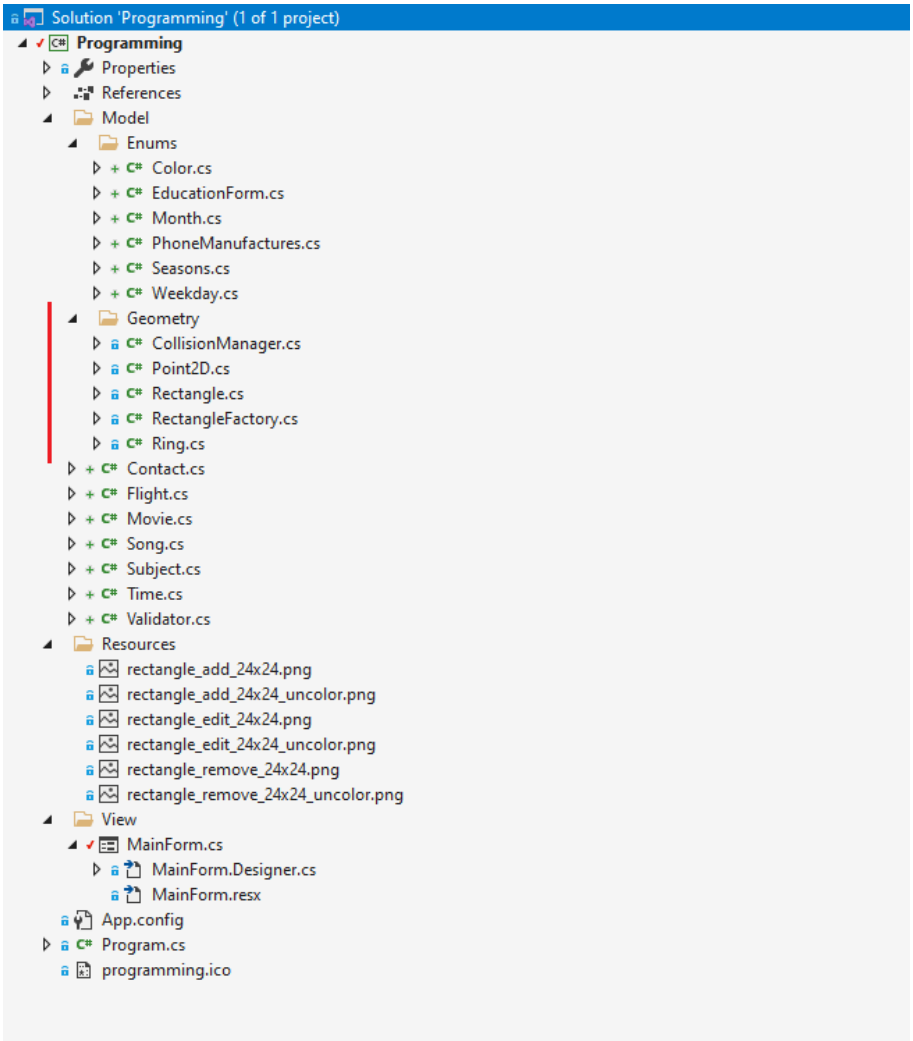
29. Если после изменения положения или размера панель стала пересекаться с другими панелями, то пересекающиеся панели должны подсветиться красным. Аналогично, если после изменения положения или размера панель перестала пересекаться, то соответствующие панели должны стать зелеными.

30. Проверьте оформление кода.

31. Помимо метода FindCollisions(), вынесите в отдельные закрытые методы следующую логику:

- UpdateRectangleInfo(Rectangle rectangle) – метод выполняет обновление данных в текстовых полях по указанному прямоугольнику;
- ClearRectangleInfo() – метод очищает все текстовые поля со значениями прямоугольника;
- статический класс RectangleFactory с методом Randomize(), возвращающим новый случайный объект прямоугольника.

32. Так как для работы с геометрическими фигурами в программе стало много классов, поместите все классы бизнес-логики в отдельную подпапку Geometry:



*** Дополнительные задания:**

33. В предоставленном примере программы обратите внимание на стили кнопок добавления и удаления прямоугольников и сравните их со стилями обычных кнопок. На новой вкладке кнопки не просто плоские, но они также: а) меняют цвет пиктограмм при наведении курсора мыши; б) не имеют внешнего контура при наведении мыши; в) имеют менее насыщенный цвет при нажатии кнопки, чем обычная кнопка. Попробуйте реализовать аналогичные кнопки. Один из вариантов реализации – использовать PictureBox вместо обычного Button, и различные события мыши.

34. В предоставленном примере программы обратите внимание, что любые новые прямоугольники генерируются строго внутри канвы и **никогда** не выходят за её пределы (отступ до края канвы не менее 15 пикселей). При этом если изменить размер окна, то новые прямоугольники генерируются с учетом новых размеров канвы. Реализуйте генерацию новых прямоугольников аналогичным способом.

35. Предлагаемый алгоритм определения пересечений прямоугольников FindCollisions() неоптимален, так как двойной цикл будет всегда перебирать все комбинации прямоугольников. Это будет заметно при добавлении большого количества прямоугольников. Попробуйте придумать более эффективный способ определения пересечений, чтобы не вызывать перерисовку всех панелей каждый раз на любые изменения в данных.