МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное образовательное учреждение высшего образования

САНКТ – ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | Д.О. Шевяков |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №8  (ВАРИАНТ №27)  ИНДЕКСАТОРЫ. |
|  |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4321 |  |  |  | П.А. Евстафьева |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_gjdgxs)

[2 Выполнение задачи](#_30j0zll) 3

[2.1 Описание реализованной функции](#_3znysh7) 3

[2.2 Листинг программы](#_2et92p0) 4

[2.3 Тесты работы программы](#_tyjcwt) 6

2.4 Интерфейс программы 7

[3 Выводы](#_3dy6vkm) 7

**1 Постановка задачи**

Необходимо разработать Windows Forms приложение на языке C# для управления списком самолётов. Программа должна поддерживать следующие функции:

Поработать с идентификаторами.

Таблица 1 – Индивидуальное задание по варианту №27

|  |  |  |
| --- | --- | --- |
| Задание | Вход | Выход |
| **Данные:**   * Модель самолета; * Авиакомпания; * Дальность полета самолета; * Запас горючего.   **Задача:**  Провести сортировку самолетов компании по дальности полета. |  |  |

**2 Выполнение задачи**

**2.1 Описание реализованной функции**

В данной программе реализован поиск самолета по ID. Каждый самолет в программе имеет уникальный идентификатор типа Guid, который генерируется при создании объекта класса Airplane. Для поиска по этому идентификатору в программе используется коллекция, содержащая все объекты самолетов. Чтобы осуществить поиск, применен индексатор, который позволяет искать самолет по его уникальному идентификатору, а не по индексу в списке.

Индексатор был реализован в отдельном классе коллекции самолетов. Этот класс предоставляет доступ к объектам через индексатор, который принимает уникальный идентификатор типа Guid. Когда индексатор используется, он ищет в коллекции самолет с указанным идентификатором и возвращает его, если такой объект найден. Если самолет с таким ID отсутствует в коллекции, возвращается значение null.

Пример использования индексатора выглядит следующим образом: при обращении к коллекции с указанием уникального идентификатора, программа выполняет поиск по ID. Если самолет найден, выводится его информация с помощью метода GetInfo(), а если самолет не найден, появляется соответствующее сообщение.

Такой подход с использованием индексаторов позволяет значительно упростить код, улучшить его читаемость и обеспечить более удобную работу с данными. Поиск по уникальному идентификатору является важной частью программы, так как гарантирует точное различие между объектами и позволяет быстро находить нужный самолет среди множества других.

**2.2 Листинг программы**

Класс Airplane на рисунках 1,2:

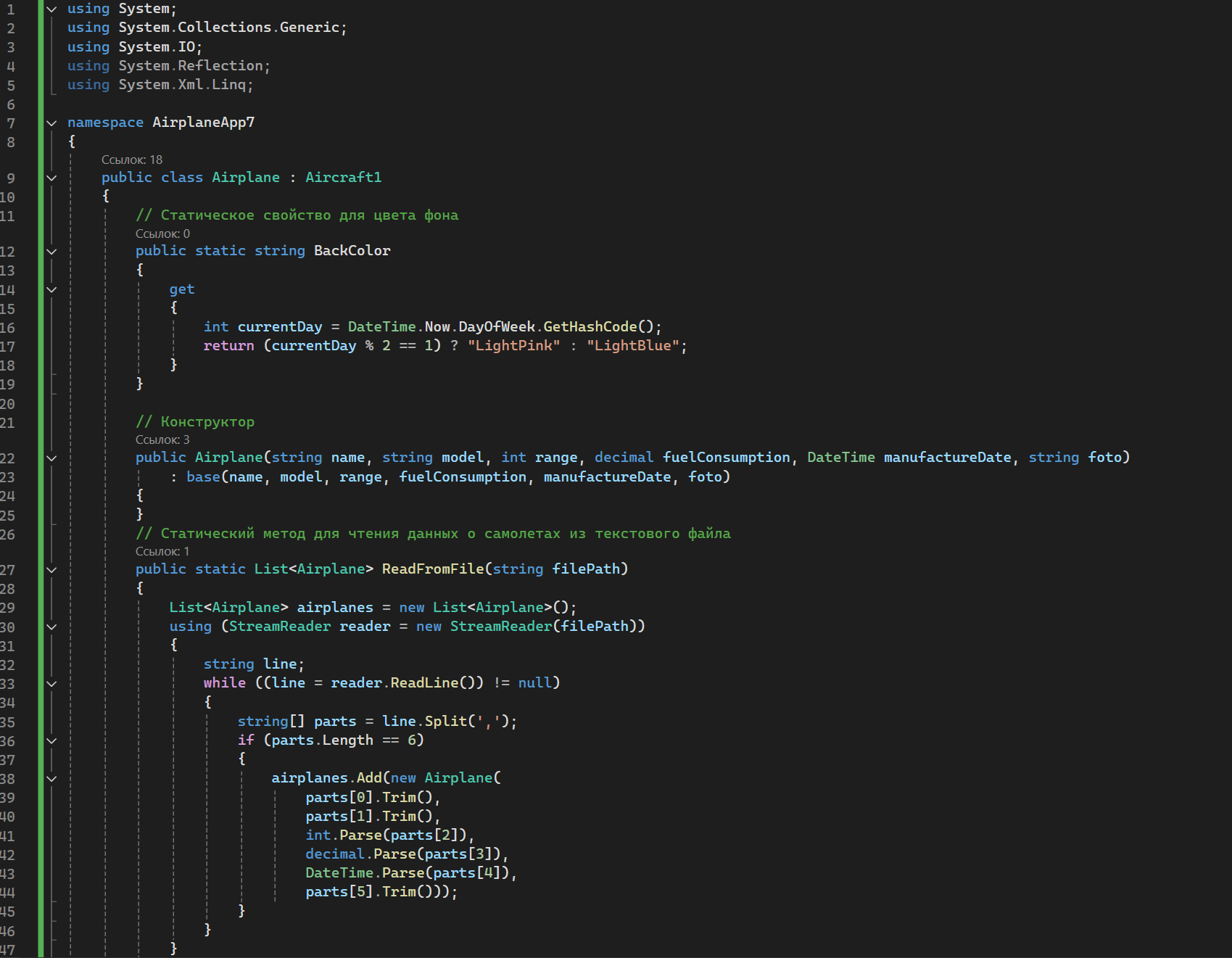


Рисунок 1 – Код класса Airplane

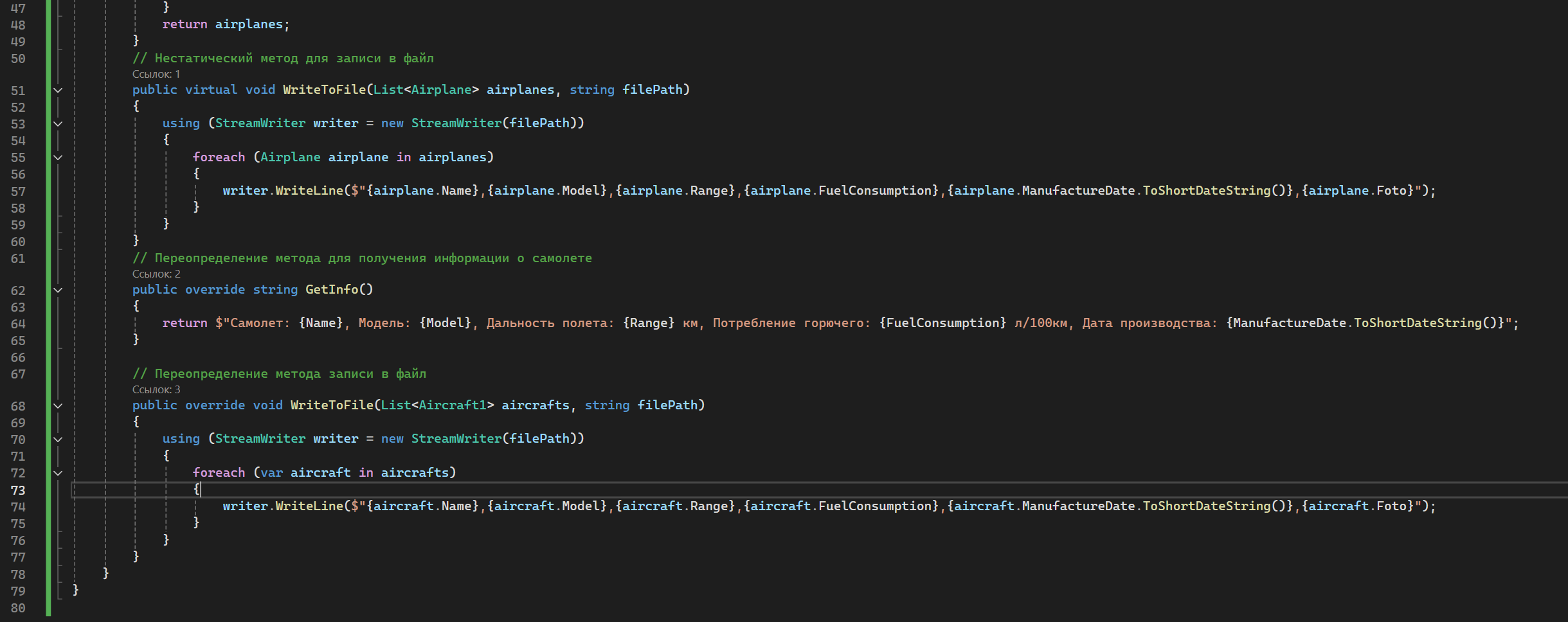


Рисунок 2 – Код класса Airplane

Код формы (Form1) на рисунках 3, 4, 5, 6:

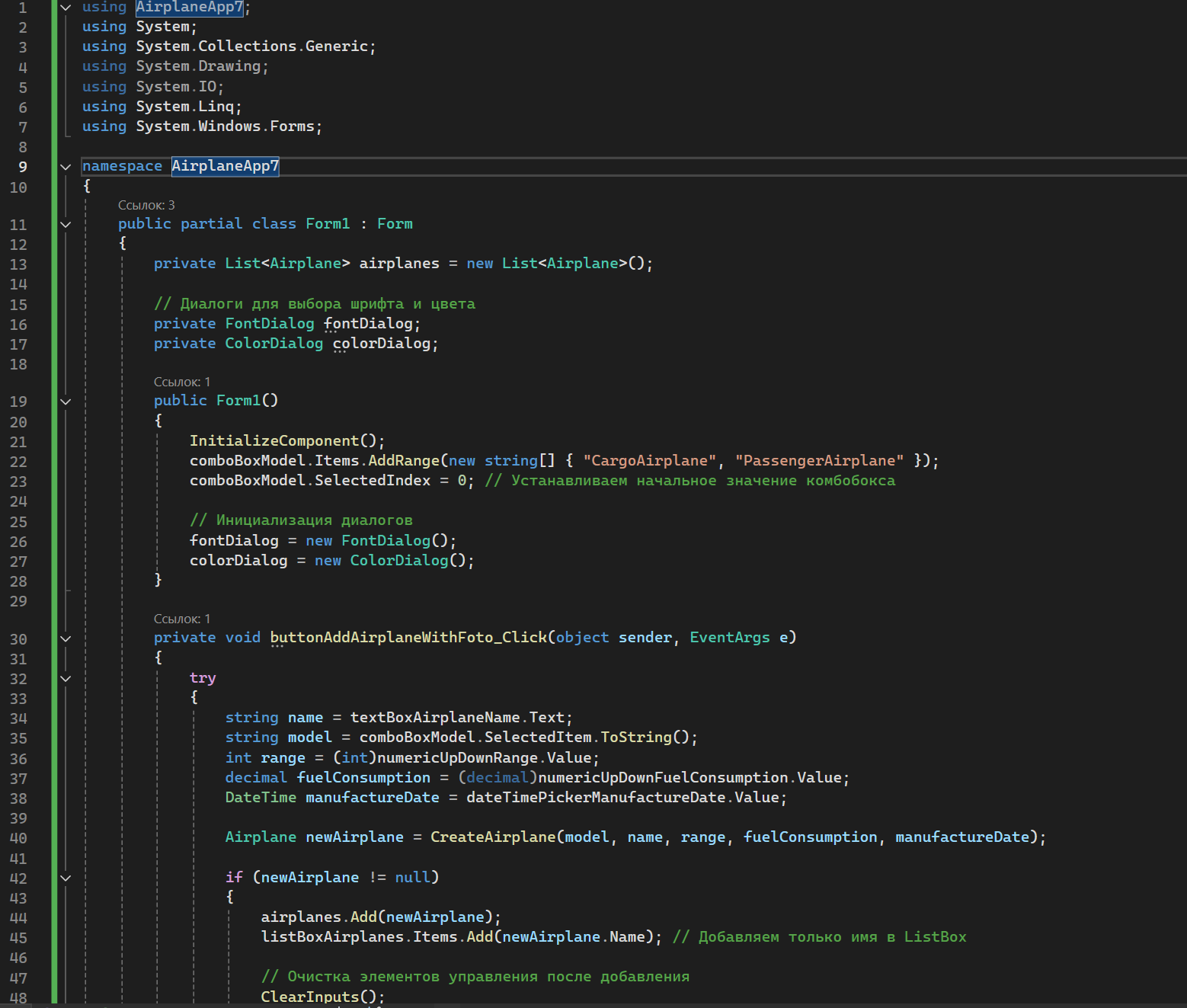


Рисунок 3 – Код формы (Form1)

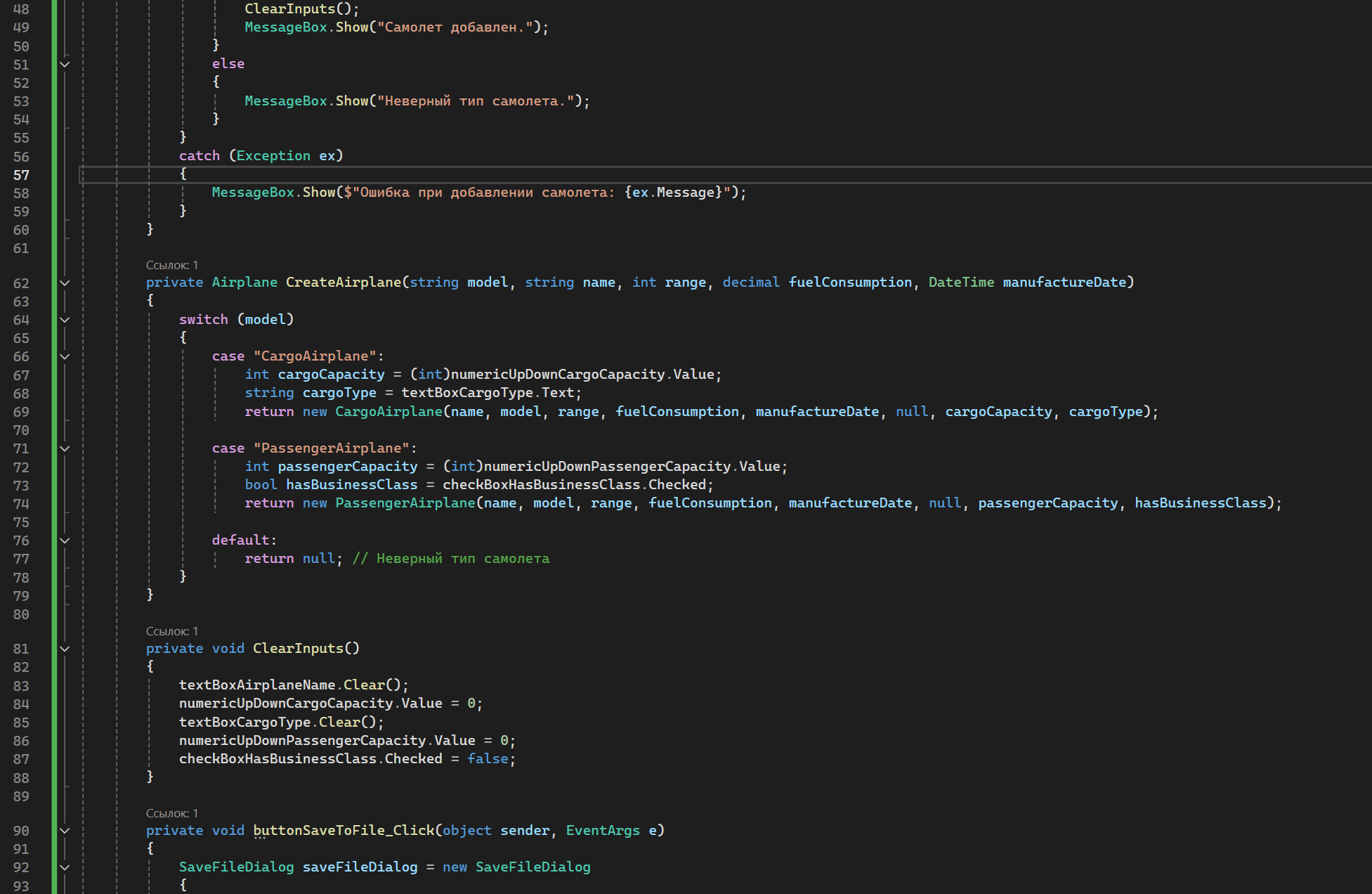


Рисунок 4 – Код формы (Form1)



Рисунок 5 – Код формы (Form1)

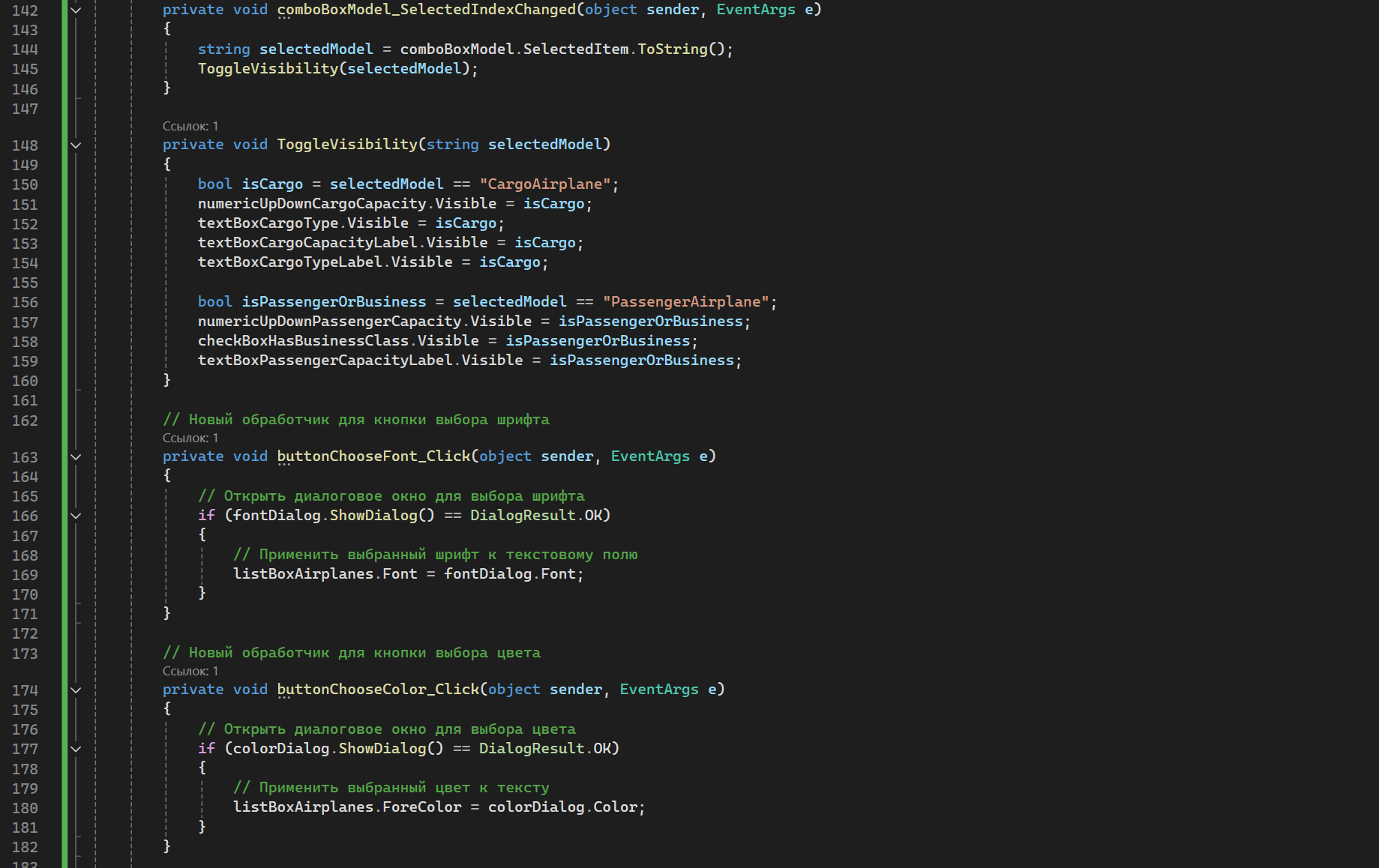


Рисунок 6 – Код формы (Form1)

На рисунке 7 изображен код дочернего класса CargoAirplane:

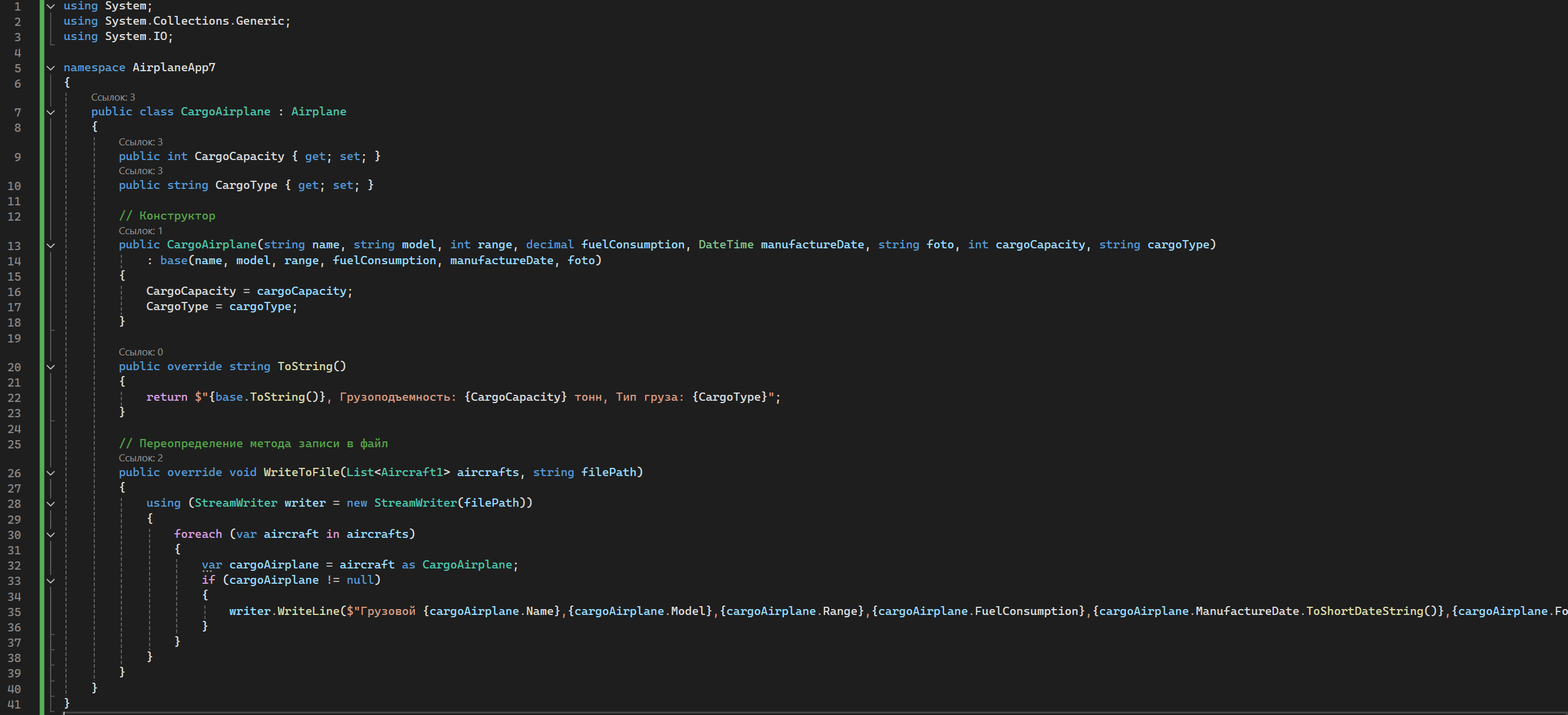


Рисунок 7 – Код дочернего класса CargoAirplane

На рисунке 8 изображен код дочернего класса PassengerAirplane:

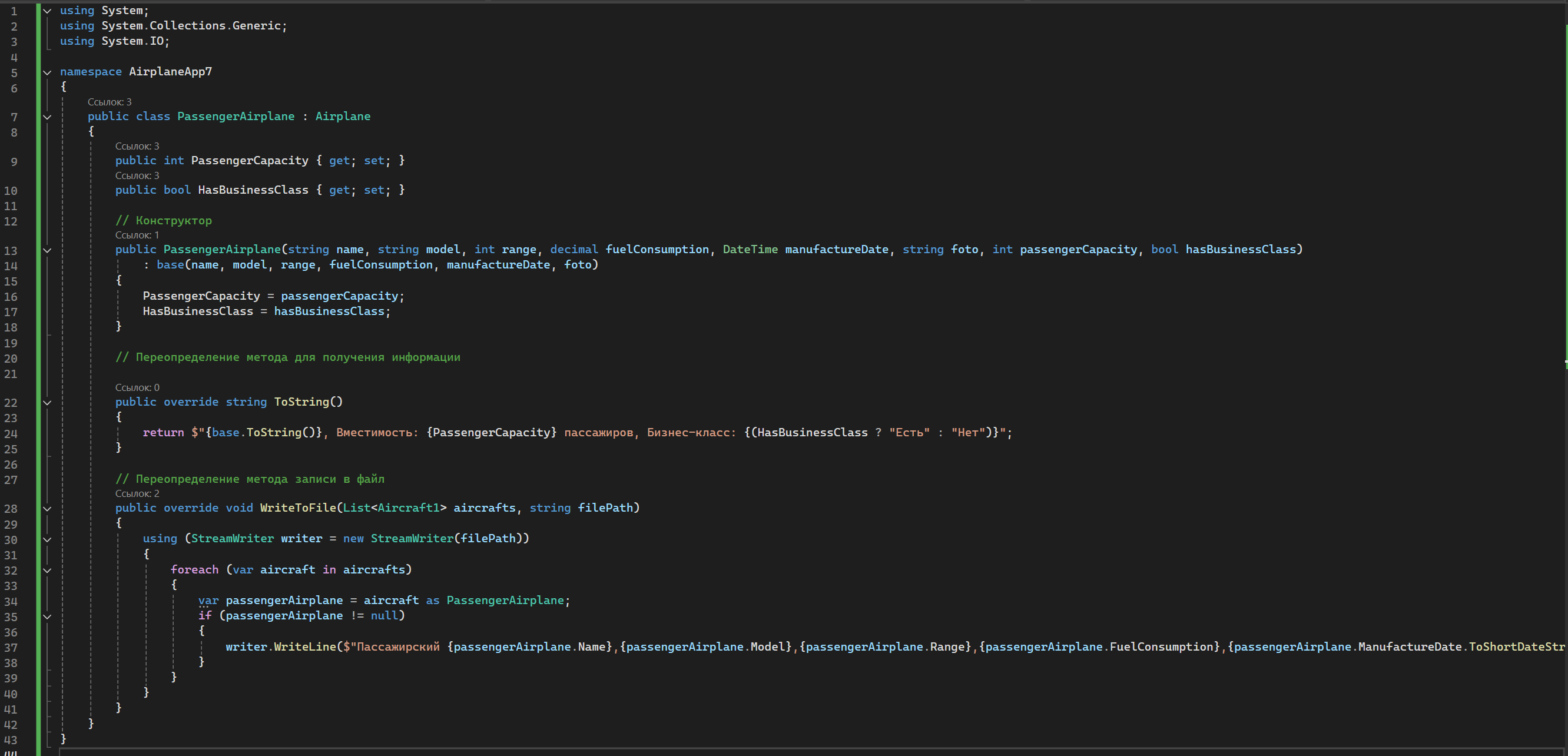


Рисунок 8 – Код дочернего класса PassengerAirplane

На рисунке 9 изображен код абстрактного класса Aircraft1:

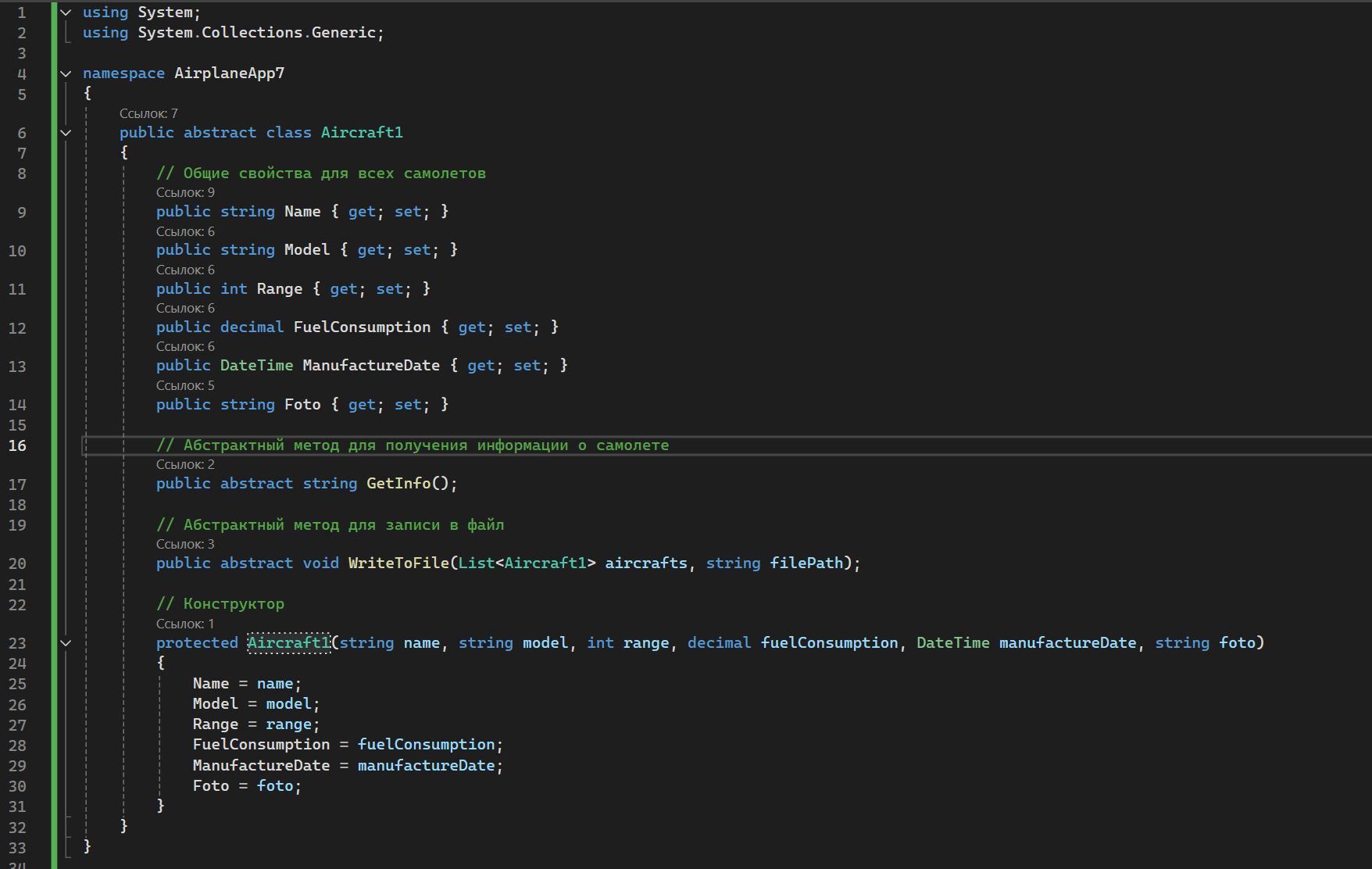


Рисунок 9 – Код абстрактного класса Aircraft1

**2.3 Тесты работы программы**

Программа была протестирована на следующих сценариях:

1. Поиск самолета

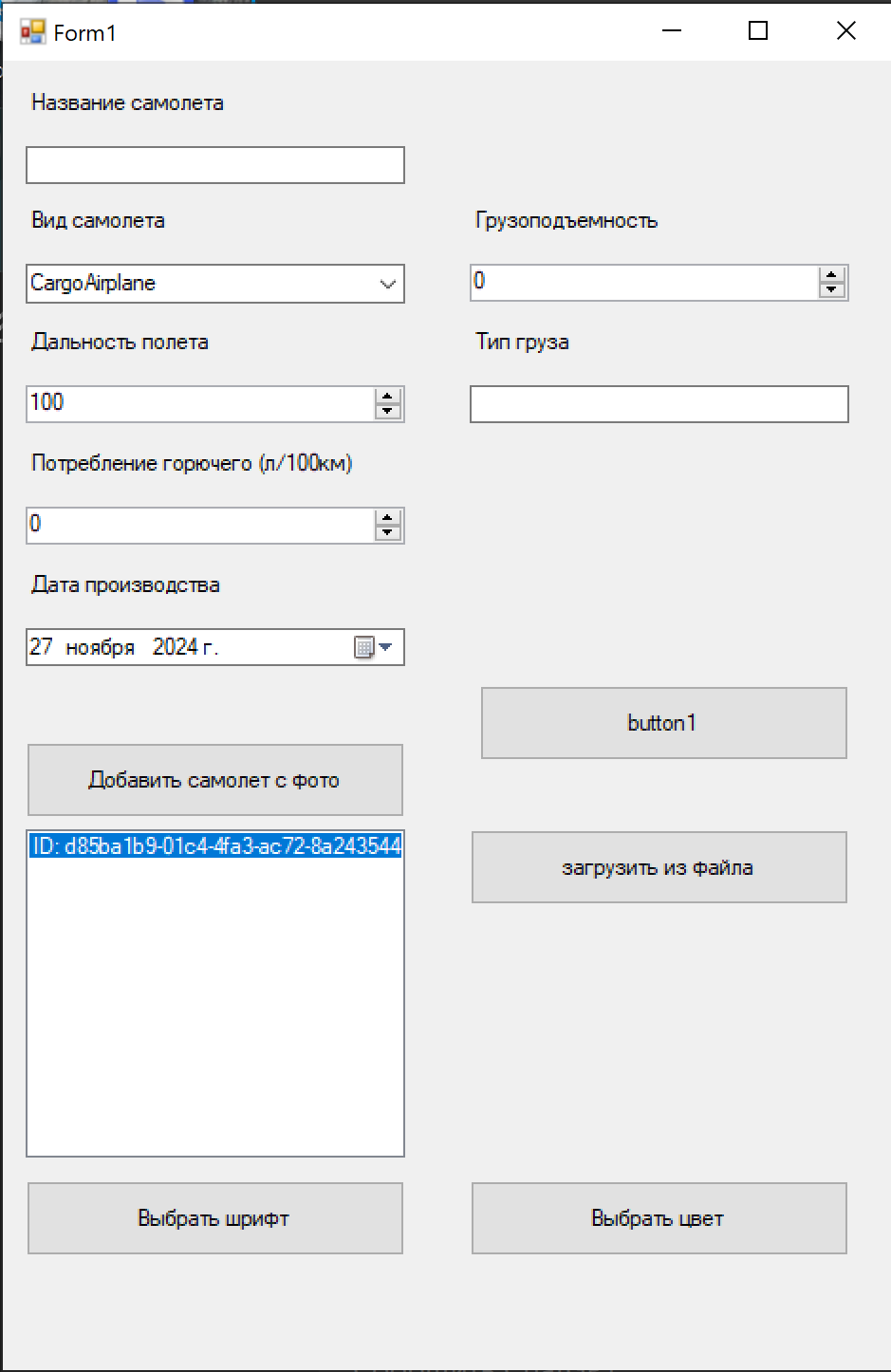


Рисунок 10 – Результат работы программы

* 1. **Интерфейс программы**

Интерфейс программы состоит из нескольких основных элементов управления, расположенных на главной форме:

Комбинированный список (ComboBox) для выбора модели самолета. Пользователь может выбрать одну из двух моделей: "CargoAirplane" (грузовой самолет) или "PassengerAirplane" (пассажирский самолет). Это значение будет использоваться для дальнейшего создания экземпляра соответствующего класса.

Текстовые поля (TextBox) для ввода информации о самолете. Например, поле для имени самолета, для типа груза, если выбран грузовой самолет, и для других характеристик. Есть также поля для ввода вместимости пассажиров, если выбран пассажирский самолет.

Числовые поля (NumericUpDown) для ввода диапазона полета, расхода топлива, вместимости груза или пассажиров. Эти элементы управляют значениями для создаваемых самолетов.

Кнопки (Button) для выполнения различных действий:

Добавить самолет с фото: при нажатии открывается диалоговое окно, позволяющее пользователю ввести информацию о самолете и добавить его в список.

Загрузить данные из файла: при нажатии открывается диалоговое окно для выбора текстового файла, из которого программа может загрузить данные о самолетах. Загруженные данные отображаются в списке.

Показать информацию о самолете: при нажатии эта кнопка позволяет получить информацию о выбранном самолете из списка, и она будет отображена в ListBox.

Список (ListBox), в который добавляются данные о каждом добавленном самолете. В нем отображаются имена и уникальные идентификаторы самолетов, а также информация о выбранном самолете.

Диалоговые окна для выбора шрифта и цвета текста, которые позволяют пользователю настроить внешний вид элементов управления.

Чекбоксы и элементы управления видимостью для отображения или скрытия дополнительных полей в зависимости от выбранной модели самолета. Например, если выбрана модель "CargoAirplane", отображаются дополнительные поля для ввода грузоподъемности и типа груза.

**3 Выводы**

В ходе лабораторной работы было создано Windows Forms приложение, которое реализует функции управления списком самолётов, их сортировки и отображения. Программа демонстрирует основные принципы объектно-ориентированного программирования, включая использование классов, перегрузку конструкторов, а также использование структур для хранения данных.