

Санкт-Петербургский Национальный Исследовательский
Университет Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

Лабораторная работа №7
Создание иерархии классов

Выполнил
Стафеев И.А.

Группа
К3221

Проверил
Иванов С.Е.

Санкт-Петербург,
2024

СОДЕРЖАНИЕ

	Стр.
ВВЕДЕНИЕ	3
1 Упражнение 1.....	4
2 Упражнение 2.....	7
3 Упражнение 3.....	12
4 Упражнение 4.....	18
5 Упражнение 5.....	21
6 Упражнение 6.....	24
7 Упражнение 7.....	27
ЗАКЛЮЧЕНИЕ	30

ВВЕДЕНИЕ

Цель работы: изучение наследования как важного элемента объектно-ориентированного программирования и приобретение навыков реализации иерархии классов.

Для достижения цели необходимо выполнить следующие упражнения:

1. Реализация наследования классов
2. Использование конструкторов
3. Переопределение методов
4. Применение абстрактного класса и абстрактных методов
5. Реализация модели включения
6. Реализация отношения ассоциации между классами
7. Реализация прогрессии

1 Упражнение 1

Задача: В этом упражнении вы будете использовать наследование для построения иерархии между классами, имеющих отношение типа “является”. Вы добавите новый класс `Item`, который будет являться базовым для уже имеющегося класса `Book`.

Код класса `Item` показан на рисунке 1. Поскольку метод `TakeItem` должен быть у любого объекта в библиотеке, а не только у книги, он был сразу перенесен в базовый класс, а метод `Take` сделан приватным.

```
4 class Item {
5     // класс библиотечной единицы хранения
6     protected long invNumber;
7     protected bool inLibrary;
8     public bool IsAvailable() { return inLibrary; } // наличие
9     public long GetInvNumber() { return invNumber; } //выдача ID
10    private void Take() { inLibrary = false; } // взятие книги
11    public void TakeItem() { // взятие книги при наличии
12        if (this.IsAvailable()) this.Take();
13    }
14    public void Return() { inLibrary = true; } // возврат
15    public void Show() {
16        Console.WriteLine("Информация о единице хранения:");
17        Console.WriteLine("Инвентарный номер: {0}", invNumber);
18        Console.WriteLine("В наличии: {0}", (this.IsAvailable()) ? "да" : "нет");
19    }
20 }
21 }
```

Рисунок 1 — Класс `Item`

Класс книги остался неизменным с прошлой лабораторной работы, за исключением указания на наследования и добавления ключевого слова `new` у метода вывода информации для указания на сокрытие метода базового класса (2).

```

29  class Book : Item
30  {
31      // атрибуты
32      private String author;
33      private String title;
34      private String publisher;
35      private int pages;
36      private int year;
37      private static double price = 9; // стоимость аренды фиксированная
38      public Book() { }
39      public Book(String author, String title, String publisher, int pages, int year) {...}
47      public Book(String author, String title) {...}
52      static Book() {...}
56      public static void SetPrice(double price) {...}
60      public double PriceBook(int s) {...}
64      new public void Show() {...}
73  }
74

```

Рисунок 2 — Класс Book

В методе Main создаются экземпляры базового класса и класса-наследника и вызываются методы для вывода информации (3).

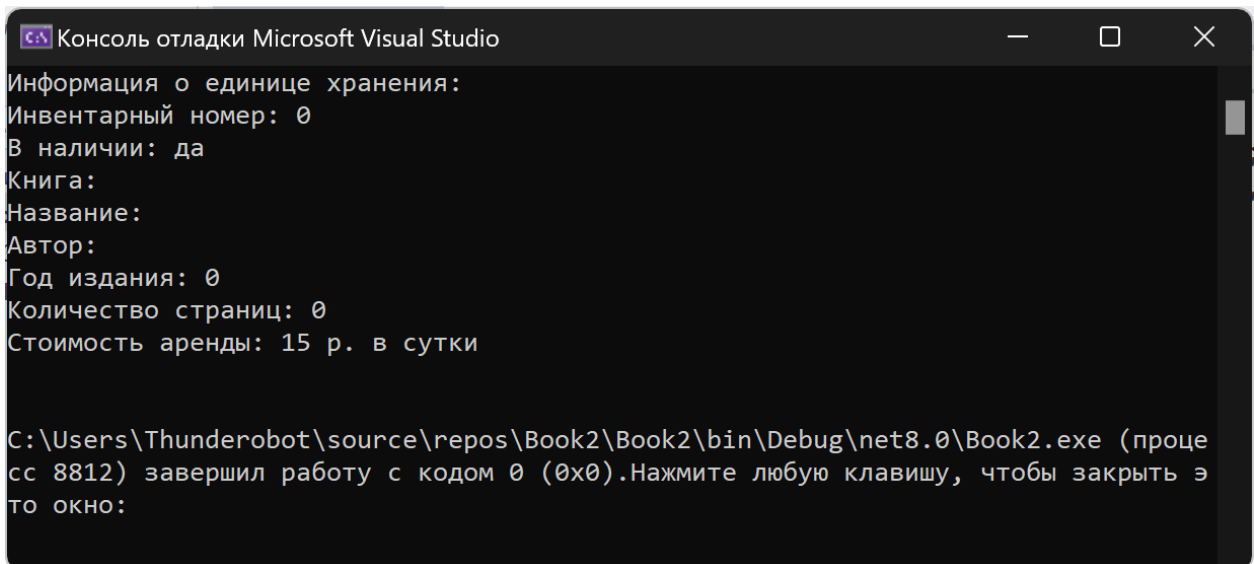
```

76  class Program {
77      public static void Main(string[] args) {
78          Item it = new Item();
79          it.Show();
80          Console.WriteLine();
81          Book b = new Book();
82          b.Show();
83      }
84  }

```

Рисунок 3 — Метод Main

Результат выполнения программы показан на рисунке 4.

The image shows a screenshot of the 'Консоль отладки Microsoft Visual Studio' (Microsoft Visual Studio Debug Console) window. The window has a dark background and a title bar with standard Windows window controls. The text inside the console is as follows:
Информация о единице хранения:
Инвентарный номер: 0
В наличии: да
Книга:
Название:
Автор:
Год издания: 0
Количество страниц: 0
Стоимость аренды: 15 р. в сутки

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (процесс 8812) завершил работу с кодом 0 (0x0).Нажмите любую клавишу, чтобы закрыть это окно:
The text is displayed in a monospaced font, with the final line being a system message about the process ending.

Рисунок 4 — Результат выполнения программы

2 Упражнение 2

Задача: В этом упражнении вы определите конструкторы в базовом и в производном классах и реализуете их выполнение в обоих классах с помощью ключевого слова `base`, которое позволяет вызвать конструктор базового класса.

В класс `Item` добавлен конструктор и конструктор по умолчанию (5).

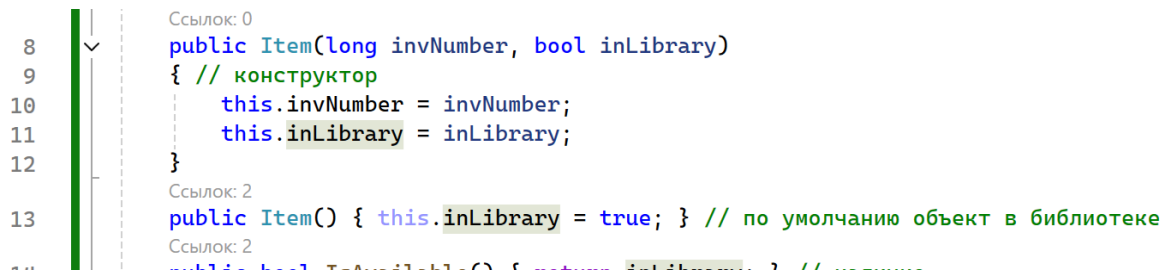
The image shows a snippet of C# code for the `Item` class. It features two constructors. The first constructor, located at lines 8-12, takes a `long` parameter `invNumber` and a `bool` parameter `inLibrary`. It is annotated with a green comment `// конструктор` and assigns the parameters to `this.invNumber` and `this.inLibrary`. The second constructor, at line 13, is a parameterless default constructor that sets `this.inLibrary` to `true`, with a green comment `// по умолчанию объект в библиотеке`. The code is displayed in a dark-themed editor with a vertical scrollbar on the left and line numbers 8, 9, 10, 11, 12, 13, and 14 visible. A green vertical bar highlights the constructor methods. A small 'v' icon is visible in the gutter between lines 8 and 9. To the right of the code, there are three lines of text: 'Ссылка: 0', 'Ссылка: 2', and 'Ссылка: 2'.

Рисунок 5 — Конструктор класса `Item`

В классе `Book` изменен конструктор: в него добавлена ссылка на конструктор базового класса, а также добавлены параметры метода базового класса (6).

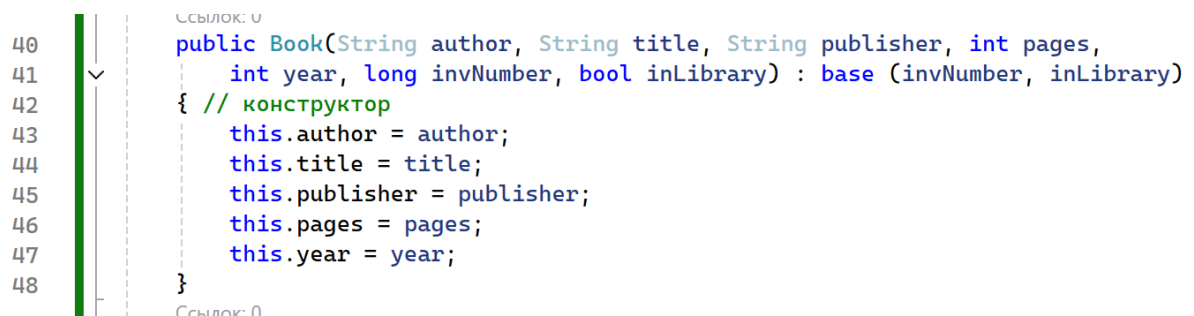
The image shows a snippet of C# code for the `Book` class. It features a single constructor at lines 40-48 that takes five parameters: `String` `author`, `String` `title`, `String` `publisher`, `int` `pages`, and `int` `year`. The constructor is annotated with a green comment `// конструктор` and includes a base class constructor call `: base (invNumber, inLibrary)` at the end of the parameter list. It then assigns each parameter to its corresponding `this` property. The code is displayed in a dark-themed editor with a vertical scrollbar on the left and line numbers 40, 41, 42, 43, 44, 45, 46, 47, and 48 visible. A green vertical bar highlights the constructor method. A small 'v' icon is visible in the gutter between lines 40 and 41. To the right of the code, there are two lines of text: 'Ссылка: 0' and 'Ссылка: 0'.

Рисунок 6 — Конструктор класса `Book`

Метод `Show` класса `Book` также изменен: добавлен вызов метода `Show` базового класса для вывода информации о книге как единице хранения (7).

```

66  new public void Show()
67  { // вывод информации о книге
68      Console.WriteLine("Книга:");
69      Console.WriteLine("Название: {0}", title);
70      Console.WriteLine("Автор: {0}", author);
71      Console.WriteLine("Год издания: {0}", year);
72      Console.WriteLine("Количество страниц: {0}", pages);
73      Console.WriteLine("Стоимость аренды: {0} р. в сутки", Book.price);
74      Console.WriteLine();
75      base.Show();
76  }

```

Рисунок 7 — Метод Show класса Book

Метод Main показан на рисунке 8. Создается экземпляр класса книги с указанием полей класса Item.

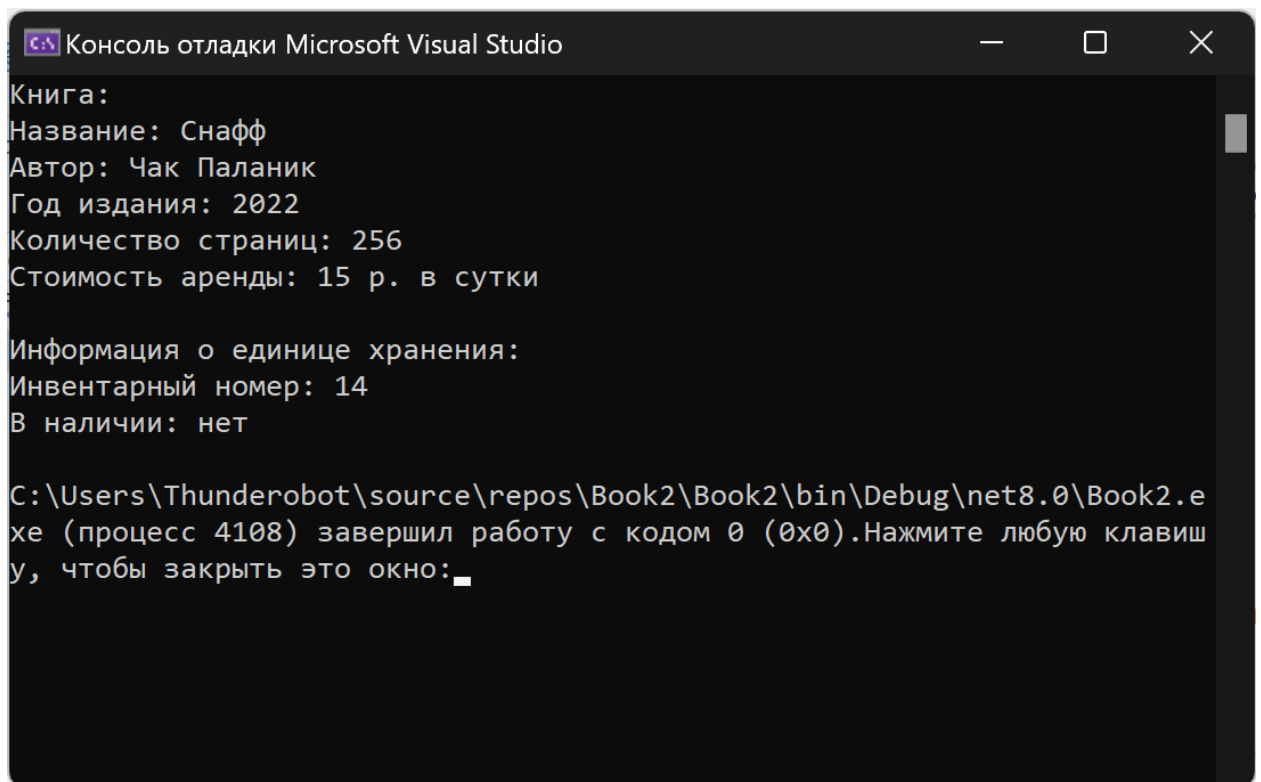
```

97  class Program {
98      public static void Main(string[] args) {
99          Book b = new Book("Чак Паланик", "Снафф", "АСТ", 256, 2022, 14, false);
100         b.TakeItem();
101         b.Show();
102     }
103 }

```

Рисунок 8 — Метод Main

Результат выполнения программы показан на рисунке 9. Можно заметить, что после вызова метода для аренды книги значение поля inLibrary стало равным false.

The image shows a screenshot of the 'Консоль отладки Microsoft Visual Studio' (Microsoft Visual Studio Debug Console) window. The window has a dark background and a light-colored title bar. The output text is as follows:

```
Книга:  
Название: Снафф  
Автор: Чак Паланик  
Год издания: 2022  
Количество страниц: 256  
Стоимость аренды: 15 р. в сутки  
  
Информация о единице хранения:  
Инвентарный номер: 14  
В наличии: нет  
  
C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.e  
xe (процесс 4108) завершил работу с кодом 0 (0x0).Нажмите любую клавиш  
у, чтобы закрыть это окно: _
```

Рисунок 9 — Результат выполнения программы

Код класса Magazine представлен на рисунке 10. По аналогии с классом Book, этот класс наследуется от Item, его конструктор содержит ссылку на базовый класс и имеет параметры базового класса, а в методе Show вызывается метод вывода информации базового класса.

```

79  class Magazine : Item {
80      // поля
81      private string volume;
82      private int number;
83      private string title;
84      private int year;
85      Ссылка: 0
86      public Magazine() { }
87      Ссылка: 0
88      public Magazine(string volume, int number, string title, int year,
89      long invNumber, bool inLibrary) : base(invNumber, inLibrary) {
90      // конструктор
91      this.volume = volume;
92      this.number = number;
93      this.title = title;
94      this.year = year;
95      }
96      Ссылка: 0
97      new public void Show() { // вывод информации
98      Console.WriteLine("Журнал:");
99      Console.WriteLine("Том: {0}", volume);
100     Console.WriteLine("Номер: {0}", number);
101     Console.WriteLine("Название: {0}", title);
102     Console.WriteLine("Год выпуска: {0}", year);
103     base.Show();
104 }
105 }

```

Рисунок 10 — Код класса Magazine

В методе Main создается экземпляр нового класса, затем выводится информация о нем (11).

```

105  class Program {
106      Ссылка: 0
107      public static void Main(string[] args) {
108      Magazine mag = new Magazine("УДК 323 Внутренняя политика", 3, "ВЕСТНИК ОГУ", 2013, 100, true);
109      mag.Show();
110  }
111 }

```

Рисунок 11 — Метод Main

Результат выполнения программы показан на рисунке 12.

```
Консоль отладки Microsoft Visual Studio

Журнал:
Том: УДК 323 Внутренняя политика
Номер: 3
Название: ВЕСТНИК ОГУ
Год выпуска: 2013

Информация о единице хранения:
Инвентарный номер: 100
В наличии: да

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (проц
есс 12524) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 12 — Результат выполнения программы

Диаграмму классов можно увидеть на рисунке 13.

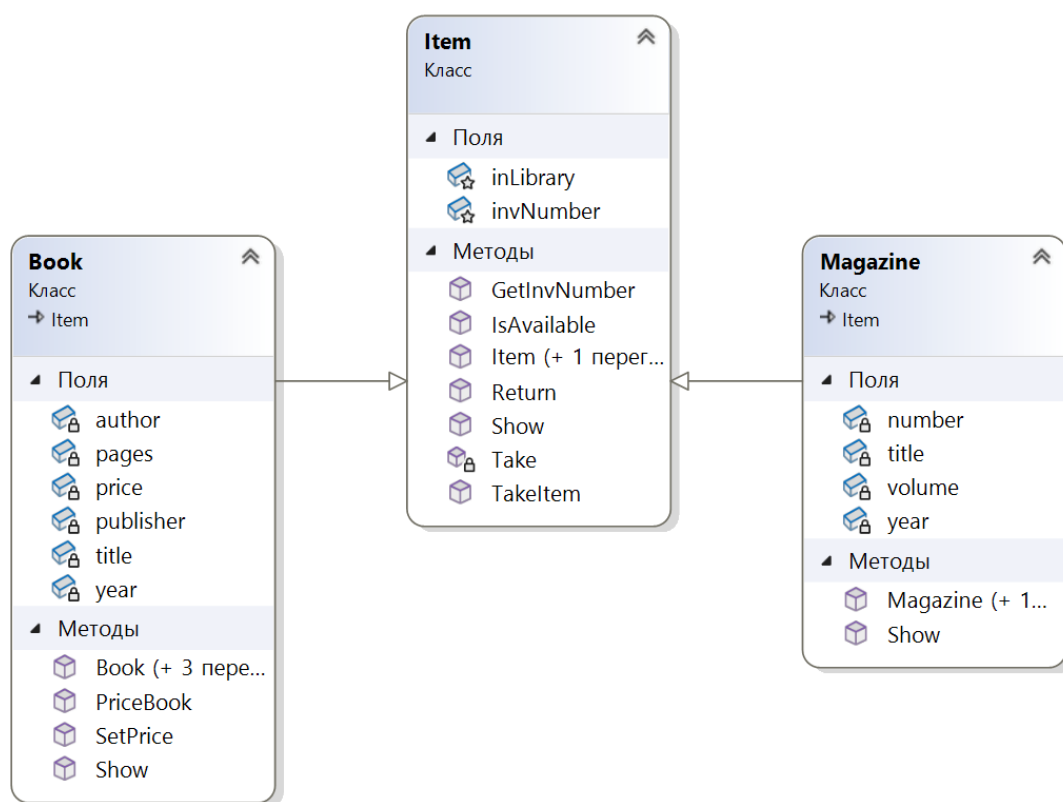


Рисунок 13 — Диаграмма классов

3 Упражнение 3

Задача: В этом упражнении вы реализуете механизм полиморфизма с помощью виртуальных методов и их переопределения в производных классах. Необходимо определить методы Show и Return базового класса как доступные к переопределению и переопределить их в производных классах.

В методе Return базового класса добавляется ключевик слово **virtual** (14).

```

20  public virtual void Return() {
21      // возврат
22      inLibrary = true;
23  }

```

Рисунок 14 — Виртуальный метод Return

В класс книги добавлено поле `returnedOnTime`, указывающее, была ли возвращена книга вовремя или нет (15).

```

33 class Book : Item
34 {
35     // атрибуты
36     private String author;
37     private String title;
38     private String publisher;
39     private int pages;
40     private int year;
41     private bool returnedOnTime;
42     private static double price = 9; // c

```

Рисунок 15 — Новое поле класса Book

Реализация возврата книги осуществляется с помощью методов `ReturnOnTime`, устанавливающего значение `true` соответствующего поля, и переопределенного с помощью ключевого слова **`override`** метода `Return`, который вызывает сдачу книги только в том случае, если она возвращена в срок (16).

```

63  ✓
64
65
66
67
68  ✓
69
70
71

```

```

Ссылка: 1
public void ReturnOnTime() {
    // указание на возврат в срок
    returnedOnTime = true;
}
Ссылка: 5
public override void Return()
{ // возврат книги (должна быть в срок)
    if (returnedOnTime) base.Return();
}
Ссылка: 0

```

Рисунок 16 — Методы `ReturnOnTime` и `Return` класса `Book`

Надобности в переопределении метода `Return` в классе `Magazine` нет, так как для журналов не учитывается, когда он был сдан.

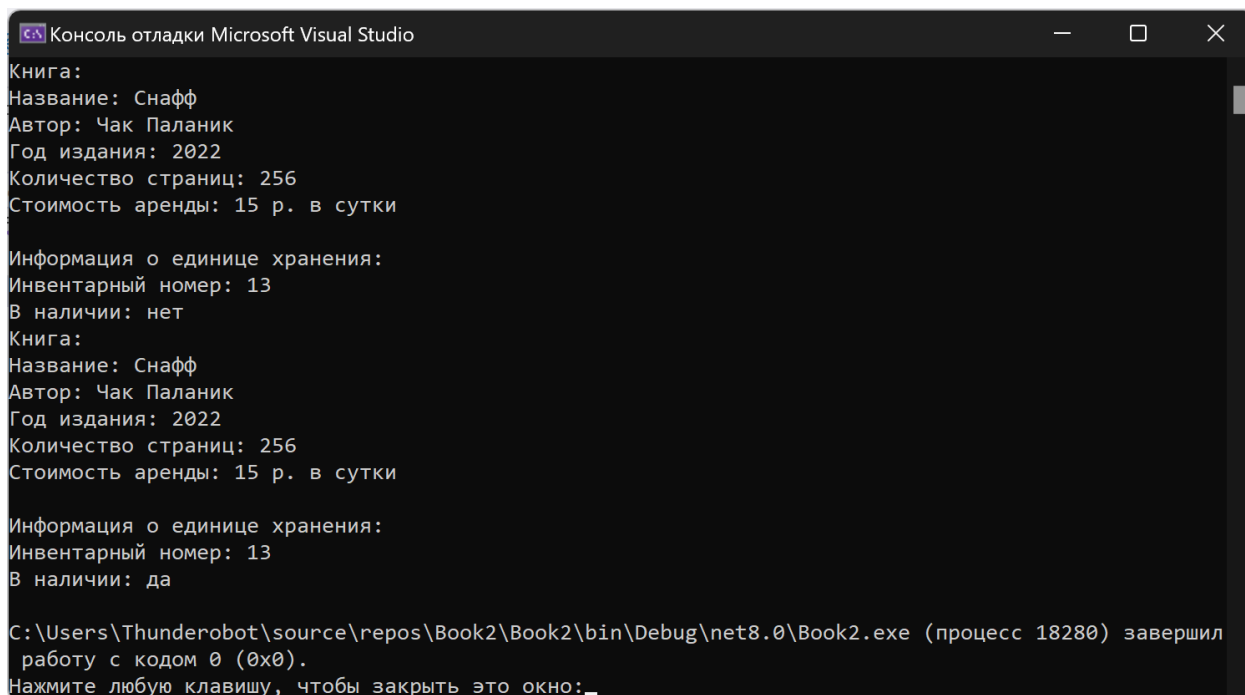
В работоспособности переопределенного метода `Return` у класса книги можно убедиться, посмотрев на картинки 17 и 18. Книга не считалась в наличии, пока не был вызван метод возвращения в срок.

```

Ссылка: 0
119  ✓ class Program {
Ссылка: 0
120  ✓ public static void Main(string[] args) {
121      Book b = new Book("Чак Паланик", "Снафф", "АСТ", 256, 2022, 13, true);
122      b.TakeItem();
123      b.Return();
124      b.Show();
125      b.ReturnOnTime();
126      b.Return();
127      b.Show();
128

```

Рисунок 17 — Тест переопределенного метода `Return`



```
Консоль отладки Microsoft Visual Studio
Книга:
Название: Снафф
Автор: Чак Паланик
Год издания: 2022
Количество страниц: 256
Стоимость аренды: 15 р. в сутки

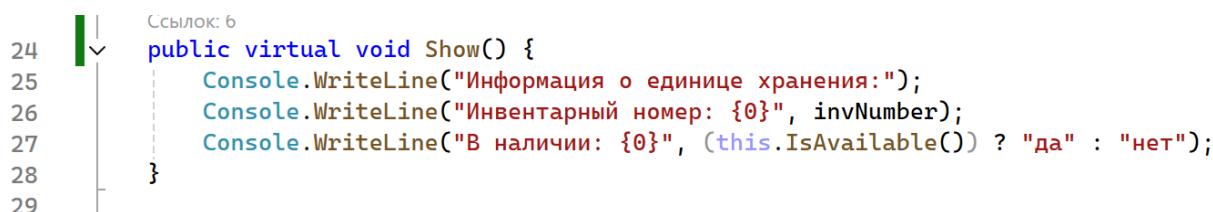
Информация о единице хранения:
Инвентарный номер: 13
В наличии: нет
Книга:
Название: Снафф
Автор: Чак Паланик
Год издания: 2022
Количество страниц: 256
Стоимость аренды: 15 р. в сутки

Информация о единице хранения:
Инвентарный номер: 13
В наличии: да

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (процесс 18280) завершил
работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

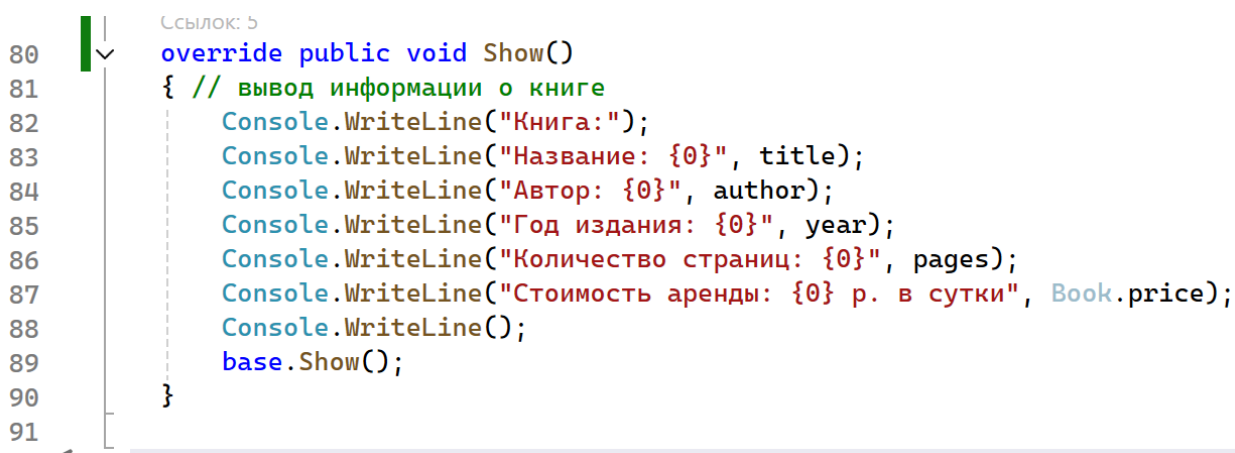
Рисунок 18 — Результат выполнения программы

Метод Show сделан виртуальным в базовом классе (19) и переопределенным в производных (20, 21).



```
Ссылка: 6
24 public virtual void Show() {
25     Console.WriteLine("Информация о единице хранения:");
26     Console.WriteLine("Инвентарный номер: {0}", invNumber);
27     Console.WriteLine("В наличии: {0}", (this.IsAvailable()) ? "да" : "нет");
28 }
29
```

Рисунок 19 — Виртуальный метод Show



```
Ссылка: 5
80 override public void Show()
81 { // вывод информации о книге
82     Console.WriteLine("Книга:");
83     Console.WriteLine("Название: {0}", title);
84     Console.WriteLine("Автор: {0}", author);
85     Console.WriteLine("Год издания: {0}", year);
86     Console.WriteLine("Количество страниц: {0}", pages);
87     Console.WriteLine("Стоимость аренды: {0} р. в сутки", Book.price);
88     Console.WriteLine();
89     base.Show();
90 }
91
```

Рисунок 20 — Переопределенный у класса Book метод Show

```

108  override public void Show() { // вывод информации
109      Console.WriteLine("Журнал:");
110      Console.WriteLine("Том: {0}", volume);
111      Console.WriteLine("Номер: {0}", number);
112      Console.WriteLine("Название: {0}", title);
113      Console.WriteLine("Год выпуска: {0}\n", year);
114      base.Show();
115  }
116

```

Рисунок 21 — Переопределенный у класса Magazine метод Show

Метод Main для тестирования полиморфизма выглядит следующим образом (22).

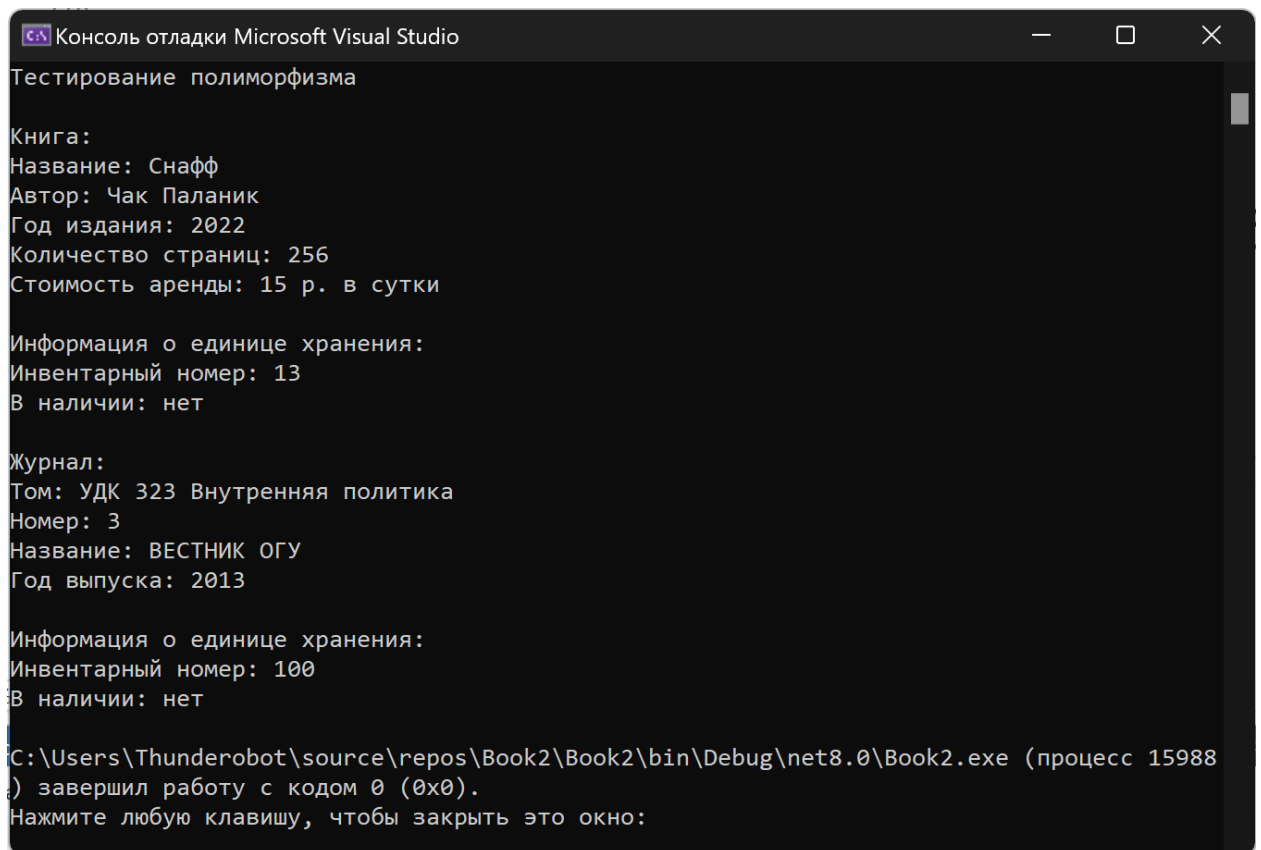
```

119  class Program {
120      public static void Main(string[] args) {
121          Console.WriteLine("Тестирование полиморфизма\n");
122          Item it;
123          Book b = new Book("Чак Паланик", "Снафф", "АСТ", 256, 2022, 13, true);
124          Magazine mag = new Magazine("УДК 323 Внутренняя политика", 3, "ВЕСТНИК ОГУ", 2013, 100, true);
125          it = b;
126          it.TakeItem();
127          it.Show();
128
129          it = mag;
130          it.TakeItem();
131          it.Show();
132      }
133  }

```

Рисунок 22 — Метод Main

Результат выполнения программы показан на рисунке 23.



```
Консоль отладки Microsoft Visual Studio
Тестирование полиморфизма

Книга:
Название: Снафф
Автор: Чак Паланик
Год издания: 2022
Количество страниц: 256
Стоимость аренды: 15 р. в сутки

Информация о единице хранения:
Инвентарный номер: 13
В наличии: нет

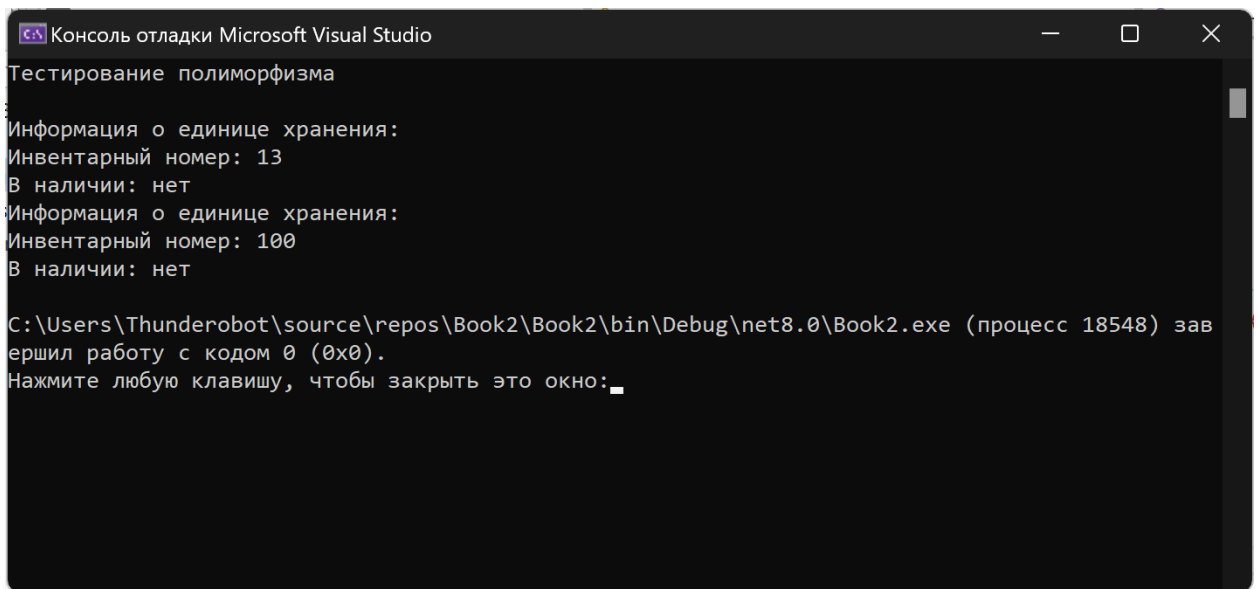
Журнал:
Том: УДК 323 Внутренняя политика
Номер: 3
Название: ВЕСТНИК ОГУ
Год выпуска: 2013

Информация о единице хранения:
Инвентарный номер: 100
В наличии: нет

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (процесс 15988
) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 23 — Тест полиморфизма

Для сравнения: если не выполнить переопределение метода Show, то результат будет такой, как на рисунке 24. Вызывается метод Show у базового класса, хотя предполагается вызов метода у класса Book и Magazine соответственно.



```
Консоль отладки Microsoft Visual Studio
Тестирование полиморфизма
Информация о единице хранения:
Инвентарный номер: 13
В наличии: нет
Информация о единице хранения:
Инвентарный номер: 100
В наличии: нет

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (процесс 18548) за-
вершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 24 — Тест при отсутствии переопределения метода Show

4 Упражнение 4

Задача: В этом упражнении вы сделаете базовый класс и метод Return абстрактным. Создавать объекты класса Item не имеет смысла, его назначение – быть базовым, и создавать экземпляры этого класса нельзя.

Класс Item и его метод Return сделаны абстрактными с помощью ключевого слова **abstract** (25).

```

4  abstract class Item {
5      // класс библиотечной единицы хранения
6      protected long invNumber;
7      protected bool inLibrary;
8      public Item(long invNumber, bool inLibrary) {...}
13     public Item() { this.inLibrary = true; } // по умолчанию объект в библиотеке
14     public bool IsAvailable() { return inLibrary; } // наличие
15     public long GetInvNumber() { return invNumber; } //выдача ID
16     private void Take() { inLibrary = false; } // взятие книги
17     public void TakeItem() {...}
20     abstract public void Return();
21     public virtual void Show() {...}
26
27 }
```

Рисунок 25 — Абстрактный класс Item

Затем метод Return в классе Book изменен (удалено обращения к методу базового класса) (26), а в классе Magazine метод переопределен (27).

```

65  public override void Return()
66  { // возврат книги (должна быть в срок)
67      if (returnedOnTime) inLibrary = true;
68  }
```

Рисунок 26 — Измененный метод Return класса Book

```

---
114      ✓
115      |
116      |
117      |
118      | }

Ссылка: 1
public override void Return()
{ // возврат журнала
  inLibrary = true;
}

```

Рисунок 27 — Переопределенный метод Return класса Magazine

Метод Main для тестирования абстрактного класса показан на рисунке 28.

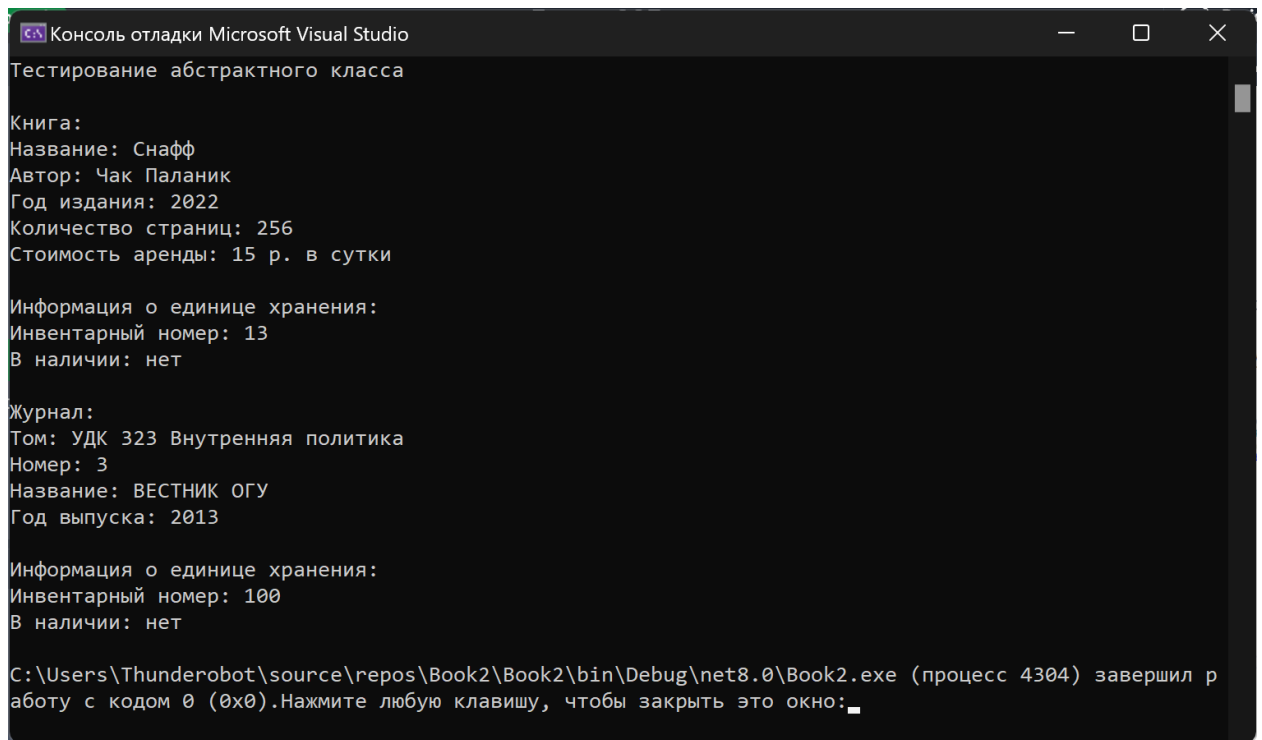
```

Ссылка: 0
121  ✓ class Program {
    Ссылка: 0
122  ✓  public static void Main(string[] args) {
123      Console.WriteLine("Тестирование абстрактного класса\n");
124      Item it;
125      Book b = new Book("Чак Паланик", "Снафф", "АСТ", 256, 2022, 13, true);
126      Magazine mag = new Magazine("УДК 323 Внутренняя политика", 3, "ВЕСТНИК ОГУ", 2013, 100, true);
127      it = b;
128      it.TakeItem();
129      it.Return();
130      it.Show();
131      Console.WriteLine();
132
133      it = mag;
134      it.Return();
135      it.TakeItem();
136      it.Show();
137  }
138  }

```

Рисунок 28 — Метод Main

Результат выполнения программы показан на рисунке 29. Видно, что метод Return вызывается и выполняется, поскольку определен в производных классах, а не вызывает ошибку, как если бы был вызван у объекта класса Item.



```
Консоль отладки Microsoft Visual Studio
Тестирование абстрактного класса

Книга:
Название: Снафф
Автор: Чак Паланик
Год издания: 2022
Количество страниц: 256
Стоимость аренды: 15 р. в сутки

Информация о единице хранения:
Инвентарный номер: 13
В наличии: нет

Журнал:
Том: УДК 323 Внутренняя политика
Номер: 3
Название: ВЕСТНИК ОГУ
Год выпуска: 2013

Информация о единице хранения:
Инвентарный номер: 100
В наличии: нет

C:\Users\Thunderobot\source\repos\Book2\Book2\bin\Debug\net8.0\Book2.exe (процесс 4304) завершил р
аботу с кодом 0 (0x0).Нажмите любую клавишу, чтобы закрыть это окно:■
```

Рисунок 29 — Результат выполнения программы

5 Упражнение 5

Задача: реализовать отношение агрегации между классами. Необходимо создать класс точки в ДСК и класс отрезка, состоящего из двух точек.

Код класса точки Point показан на рисунке 30. Полями являются два вещественных числа - координаты, которые задаются в конструкторе. Определены методы для вывода информации о точке, нахождения расстояния до другой точки и приведения к строке.

```
4  class Point {  
5      // класс точки  
6      private double x, y;  
7      public Point() { }  
8      public Point(double x, double y) {  
9          // конструктор  
10         this.x = x; this.y = y;  
11     }  
12     public void Show() {  
13         // вывод информации  
14         Console.WriteLine("Точка с координатами ({0}, {1})", x, y);  
15     }  
16     public double GetDistance(Point p) {  
17         // расстояние между двумя точками  
18         return Math.Sqrt((x - p.x) * (x - p.x) + (y - p.y) * (y - p.y));  
19     }  
20     public override string ToString()  
21     { // приведение к строке  
22         return "(" + x + "; " + y + " ";  
23     }  
24 }
```

Рисунок 30 — Код класса Point

Код класса отрезка представлен на рисунке 31. Его полями являются объект созданного ранее класса Point. Определен метод для нахождения длины отрезка, использующий метод точки для нахождения расстояния, и метод для вывода информации.

```

26  class Line {
27      // класс линии
28      private Point pStart, pEnd;
29      public Line() { }
30      public Line(Point pStart, Point pEnd) {
31          this.pStart = pStart; this.pEnd = pEnd;
32      }
33      public void Show() {
34          // вывод информации
35          Console.WriteLine("Отрезок с координатами {0} - {1}", pStart, pEnd);
36      }
37      public double GetLength() {
38          // получение длины
39          return pStart.GetDistance(pEnd);
40      }
41  }

```

Рисунок 31 — Код класса Line

Код метода Main для теста классов показан на рисунке 32. В нем создаются две точки и линия, их содержащая. О каждом объекте выводится информация, а для отрезка выводится и его длина.

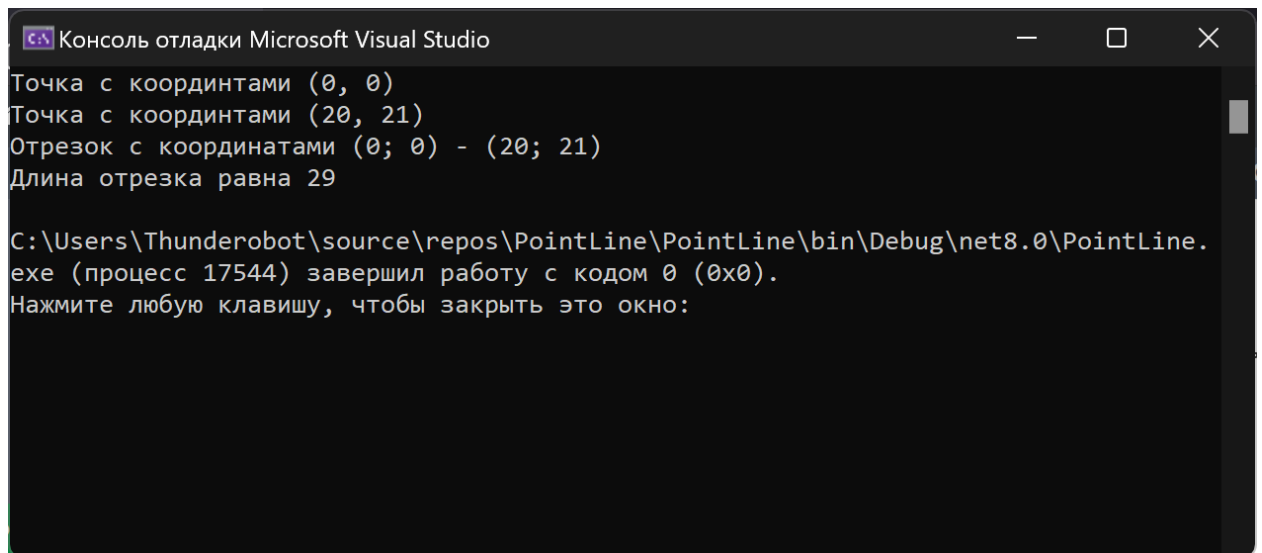
```

43  class Program {
44      public static void Main(string[] args) {
45          Point p1 = new Point();
46          p1.Show();
47          Point p2 = new Point(20, 21);
48          p2.Show();
49          Line line = new Line(p1, p2);
50          line.Show();
51          Console.WriteLine("Длина отрезка равна {0}", line.GetLength());
52      }
53  }

```

Рисунок 32 — Метод Main

Результат выполнения программы можно увидеть на рисунке 33.



```
Консоль отладки Microsoft Visual Studio
Точка с координатами (0, 0)
Точка с координатами (20, 21)
Отрезок с координатами (0; 0) - (20; 21)
Длина отрезка равна 29

C:\Users\Thunderobot\source\repos\PointLine\PointLine\bin\Debug\net8.0\PointLine.exe (процесс 17544) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 33 — Метод Main

6 Упражнение 6

Задача: реализовать модель ассоциации между классами. Создать класс игральной кости и игрока, которые связаны ассоциацией - отдельный сеанс игры связан с конкретным игроком, бросающим кости.

Класс игровой кости Dice представлен на рисунке 34. Поле класса является объект класса Random, который создается внутри конструктора. Метод Throw возвращает случайное число от 1 до 6.

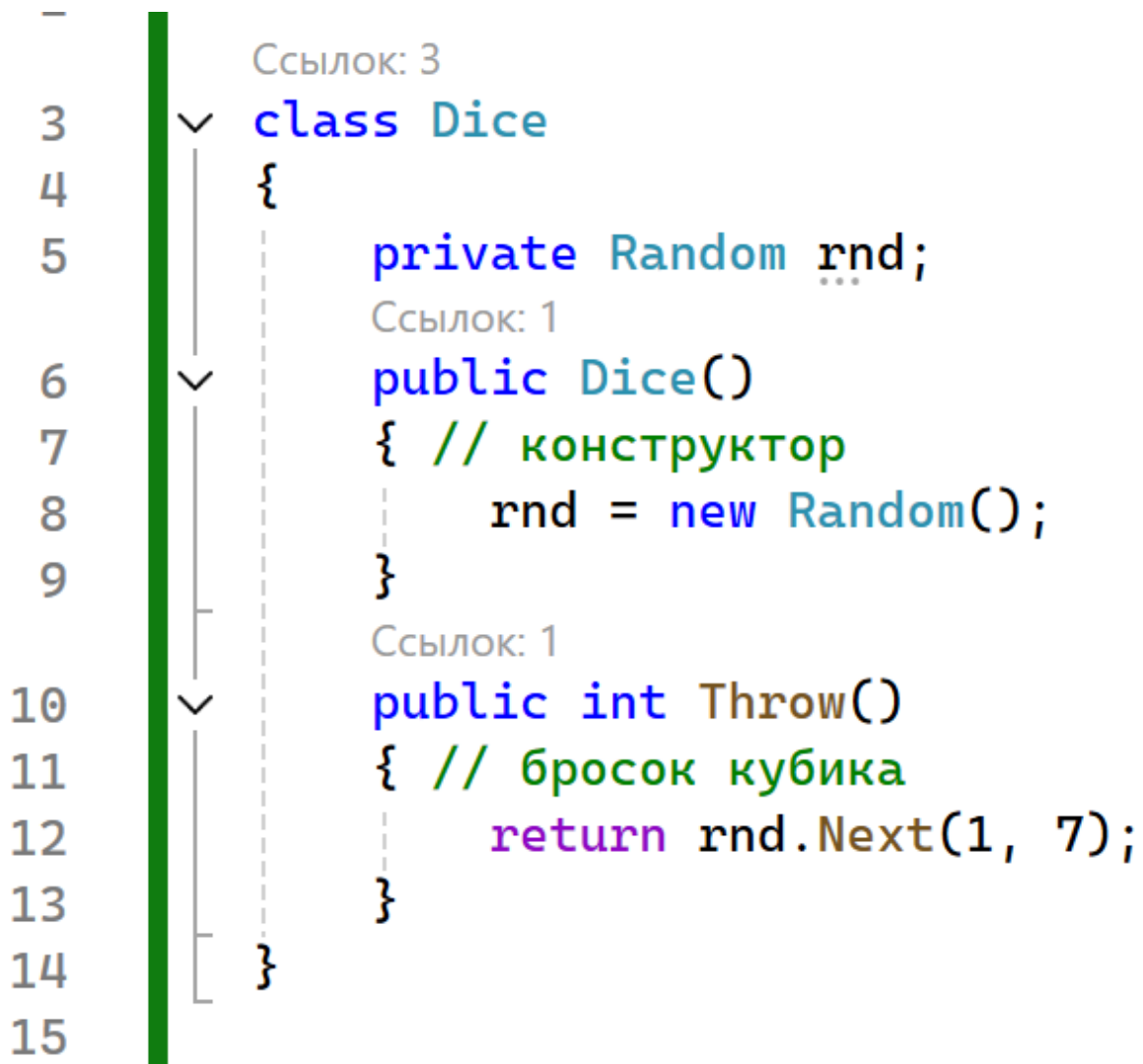


Рисунок 34 — Класс Dice

Класс игрока Player представлен на рисунке 35. Поле класса является объект созданного ранее метода Dice. Метод Play обозначает одну игровую сессию, то есть один бросок кубика.


```

16  class Player
17  {
18      private string name;
19      private Dice dice;
20      public Player(string name)
21      {
22          this.name = name;
23          this.dice = new Dice();
24      }
25      public int Play()
26      { // выполнить бросок кубика игроком
27          return dice.Throw();
28      }
29      public override string ToString()
30      {
31          return name;
32      }
33  }
34

```

Ссылки: 3 (к строке 16), 1 (к строке 20), 1 (к строке 25), 0 (к строке 29)

Рисунок 35 — Класс Player

Код метода Main для теста классов показан на рисунке 36. В нем создается экземпляр класса игрока, и в цикле делаются броски и выводится о них информация.

```

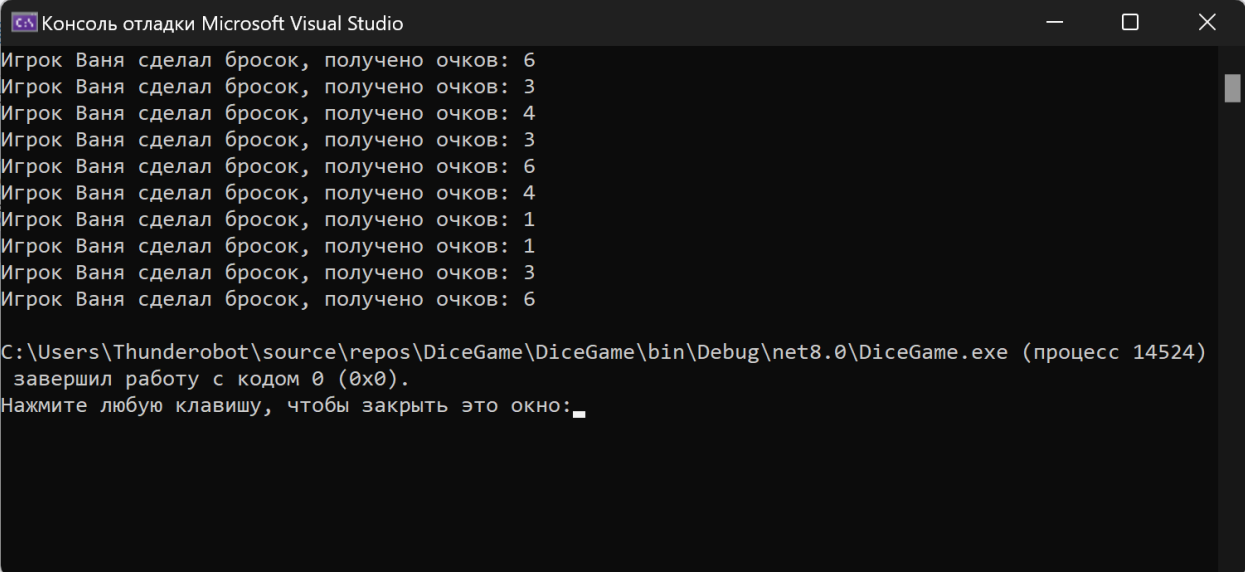
35  class Program {
36      public static void Main(string[] args) {
37          Player p = new Player("Ваня");
38          for (int i = 0; i < 10; i++) {
39              Console.WriteLine("Игрок {0} сделал бросок, получено очков: {1}", p, p.Play());
40          }
41      }
42  }

```

Ссылки: 0 (к строке 35), 0 (к строке 36)

Рисунок 36 — Метод Main

Результат выполнения программы показан на рисунке 37.



```
Консоль отладки Microsoft Visual Studio
Игрок Ваня сделал бросок, получено очков: 6
Игрок Ваня сделал бросок, получено очков: 3
Игрок Ваня сделал бросок, получено очков: 4
Игрок Ваня сделал бросок, получено очков: 3
Игрок Ваня сделал бросок, получено очков: 6
Игрок Ваня сделал бросок, получено очков: 4
Игрок Ваня сделал бросок, получено очков: 1
Игрок Ваня сделал бросок, получено очков: 1
Игрок Ваня сделал бросок, получено очков: 3
Игрок Ваня сделал бросок, получено очков: 6

C:\Users\Thunderobot\source\repos\DiceGame\DiceGame\bin\Debug\net8.0\DiceGame.exe (процесс 14524)
завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 37 — Результат выполнения программы

7 Упражнение 7

В этом упражнении требуется:

- определить абстрактный класс `Progression`, описывающий прогрессии
- определить два производных класса `ArithmeticProgression` и `GeometricProgression`, описывающие арифметическую и геометрическую прогрессии.
- в каждом из классов необходимо определить конструктор, задающий параметры прогрессии и перегрузить унаследованный метод `GetElement`.

Код абстрактного класса показан на рисунке 38. Полями класса являются `firstElement` - первый элемент прогрессии - и `shift` - изменение прогрессии (шаг или знаменатель в зависимости от типа прогрессии). Добавлен конструктор и абстрактный класс получения элемента `GetElement`.

```
3  |  Ссылка: 6
4  |  ✓ abstract class Progression {
5  |      private protected double firstElement;
6  |      private protected double shift;
7  |
8  |      Ссылка: 0
9  |      public Progression() { }
10 |      Ссылка: 2
11 |      ✓ public Progression(double firstElement, double shift)
12 |      { // конструктор
13 |          this.firstElement = firstElement;
14 |          this.shift = shift;
15 |      }
16 |      Ссылка: 4
17 |      abstract public double GetElement(int k);
18 |  }
```

Рисунок 38 — Код абстрактного класса `Progression`

Класс арифметической прогрессии наследуется от базового (39). В нем переопределен метод нахождения k -го члена прогрессии, а конструктор ссылается на базовый класс.

```

17  class ArithmeticProgression : Progression {
18
19      Ссылка: 1
20      public ArithmeticProgression(double firstElement, double shift) : base(firstElement, shift) { }
21      Ссылка: 2
22      public override double GetElement(int k)
23      { // получение k-го члена
24          return firstElement + (k - 1) * shift;
25      }

```

Рисунок 39 — Код класса ArithmeticProgression

Аналогичным образом создан класс геометрической прогрессии (40).

```

26  class GeometricProgression : Progression {
27      Ссылка: 1
28      public GeometricProgression(double firstElement, double shift) : base(firstElement, shift) { }
29      Ссылка: 2
30      public override double GetElement(int k)
31      { // получение k-го члена
32          checked
33          {
34              return firstElement * Math.Pow(shift, k - 1);
35          }
36      }

```

Рисунок 40 — Код класса GeometricProgression

Метод Main для тестирования прогрессий представлен на рисунке 41. Программа запрашивает у пользователя первые члены и изменения для двух прогрессий, а также два числа k - номера искоемых членов. Затем программа выводит найденные члены прогрессий.

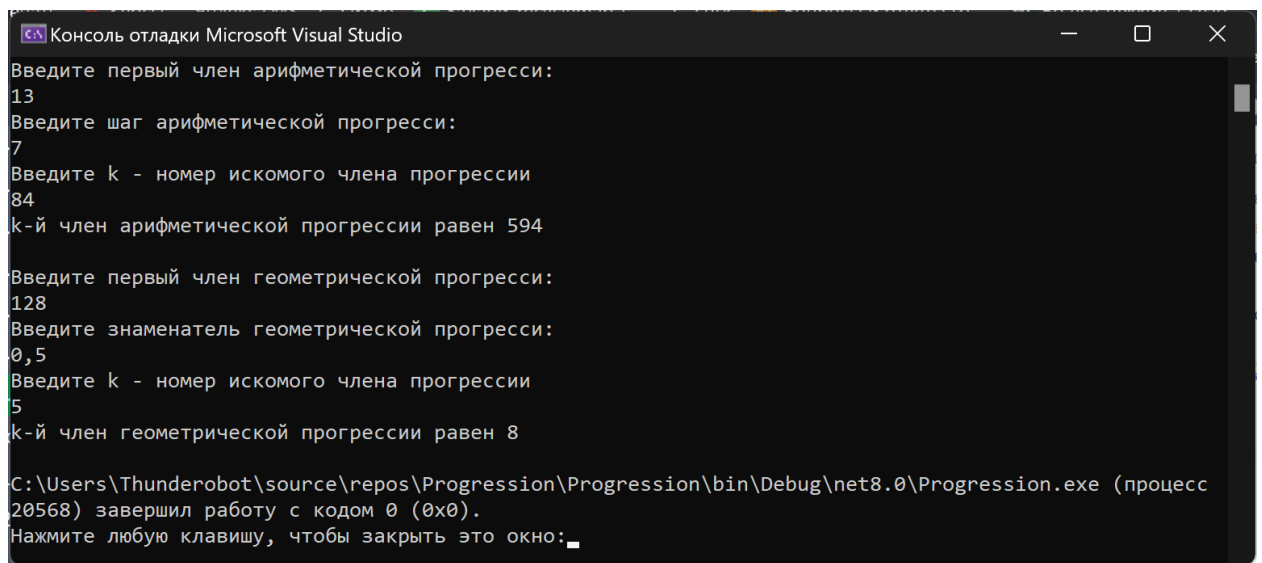
```

37  class Program {
38      Ссылка: 0
39      public static void Main(string[] args) {
40          Console.WriteLine("Введите первый член арифметической прогрессии:");
41          double a = double.Parse(Console.ReadLine());
42          Console.WriteLine("Введите шаг арифметической прогрессии:");
43          double d = double.Parse(Console.ReadLine());
44          ArithmeticProgression arprog = new ArithmeticProgression(a, d);
45          Console.WriteLine("Введите k - номер искомого члена прогрессии");
46          int k = int.Parse(Console.ReadLine());
47          Console.WriteLine("k-й член арифметической прогрессии равен {0}\n", arprog.GetElement(k));
48
49          Console.WriteLine("Введите первый член геометрической прогрессии:");
50          double b = double.Parse(Console.ReadLine());
51          Console.WriteLine("Введите знаменатель геометрической прогрессии:");
52          double q = double.Parse(Console.ReadLine());
53          GeometricProgression geoprogr = new GeometricProgression(b, q);
54          Console.WriteLine("Введите k - номер искомого члена прогрессии");
55          k = int.Parse(Console.ReadLine());
56          Console.WriteLine("k-й член геометрической прогрессии равен {0}", geoprogr.GetElement(k));
57      }

```

Рисунок 41 — Метод Main

Пример выполнения программы показан на рисунке 42.



```
Консоль отладки Microsoft Visual Studio
Введите первый член арифметической прогрессии:
13
Введите шаг арифметической прогрессии:
7
Введите k - номер искомого члена прогрессии
84
k-й член арифметической прогрессии равен 594

Введите первый член геометрической прогрессии:
128
Введите знаменатель геометрической прогрессии:
0,5
Введите k - номер искомого члена прогрессии
5
k-й член геометрической прогрессии равен 8

C:\Users\Thunderobot\source\repos\Progression\Progression\bin\Debug\net8.0\Progression.exe (процесс
20568) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 42 — Пример выполнения программы

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были выполнены все требуемые упражнения. Цель работы достигнута. Получены знания о наследовании в языке C# и приобретены навыки реализации иерархии классов.