

**Министерство науки и высшего образования Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ**  
**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**  
**ITMO University**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8**

**По дисциплине** Объектно-ориентированное программирование

**Тема работы** Использование интерфейсов при реализации иерархии классов

**Обучающийся** Буров Глеб Максимович

**Факультет** факультет инфокоммуникационных технологий

**Группа** К3223

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и системы связи

**Образовательная программа** Программирование в инфокоммуникационных системах

**Обучающийся**

\_\_\_\_\_  
(дата)

\_\_\_\_\_  
(подпись)

Буров Г.М.  
(Ф.И.О.)

**Руководитель**

\_\_\_\_\_  
(дата)

\_\_\_\_\_  
(подпись)

Иванов С.Е.  
(Ф.И.О.)

# СОДЕРЖАНИЕ

Стр.

|                             |          |
|-----------------------------|----------|
| <b>ВВЕДЕНИЕ .....</b>       | <b>3</b> |
| <b>1   Ход работы .....</b> | <b>4</b> |
| 1.1   Упражнение 1 .....    | 4        |
| 1.2   Упражнение 2 .....    | 5        |
| 1.3   Упражнение 3 .....    | 6        |
| <b>ЗАКЛЮЧЕНИЕ .....</b>     | <b>8</b> |

## **ВВЕДЕНИЕ**

Целью данной лабораторной работы является использование интерфейсов при реализации иерархии классов как важного элемента объектно-ориентированного программирования и приобретение навыков реализации интерфейсов.

# 1 Ход работы

## 1.1 Упражнение 1

В первом упражнении нужно было создать интерфейс, определяющий поведение классов, которые будут его реализовывать.

Для этого в проекте MyClass из прошлого упражнения был создан интерфейс IPubs. В интерфейсе были объявлены его функциональные члены — метод для проверки оформлена ли подписка на издание Subs и свойство IfSubs для оформления подписки (рис. 1.1).

```
internal interface IPubs
{
    Ссылка: 2
    void Subs();
    Ссылка: 3
    bool IfSubs { get; set; }
}
```

Рисунок 1.1 — Интерфейс IPubs

В классе Magazine в список наследования добавился интерфейс IPubs, свойство и метод IfSubs, объявленные в интерфейсе, были реализованы. Также был реализован метод Subs (рис. 1.2).

```
Ссылка: 3
public bool IfSubs { get; set; }
Ссылка: 2
public void Subs()
{
    Console.WriteLine("Подписка на журнал \"{0}\": {1}.", title, IfSubs);
}
```

Рисунок 1.2 — Изменения в классе Magazine

В методе Main была протестирована новая функциональность (рис. 1.3).

|  |  |
|--|--|
| <pre>Console.WriteLine("\n Тестирование полим<br/><br/>Item it;<br/>it = b2;<br/>it.TakeItem();<br/>it.Return();<br/>it.Show();<br/><br/>it = mag1;<br/>it.TakeItem();<br/>it.Return();<br/>it.Show();<br/>mag1.IfSubs = true;<br/>mag1.Subs();<br/>/*</pre> | <pre>Журнал:<br/>Том: 0 природе<br/>Номер: 5<br/>Название: Земля и мы<br/>Год выпуска: 2014<br/>Состояние единицы хранения:<br/>Инвентарный номер: 1235<br/>Наличие: True<br/>Подписка на журнал "Земля и мы": True.<br/>Для продолжения нажмите любую клавишу . . .</pre> |
|--|--|

Рисунок 1.3 — Работа программы из упражнения 1

## 1.2 Упражнение 2

Во втором упражнении нужно было применить стандартный интерфейс `Comparable`, который задает метод сравнения объектов по принципу больше и меньше, что позволяет переопределить соответствующие операции в рамках класса, наследующего интерфейс `Comparable`.

В нашем случае сравнение и дальнейшая сортировка будет реализована по полю `invNumber`. Для этого в объявление абстрактного класса `Item` было добавлено наследование интерфейса `Comparable`. Также в классе был явно реализован метод `CompareTo`, реализующий сравнение по инвентарному номеру (рис. 1.4).

```
Ссылка: 0  
int Comparable.CompareTo(object obj)  
{  
    Item it = (Item)obj;  
    if (this.invNumber == it.invNumber) return 0;  
    else if (this.invNumber > it.invNumber) return 1;  
    else return -1;  
}
```

Рисунок 1.4 — Изменения в классе `Item`

В методе `Main` был создан список объектов класса `Item`, включивший в себя созданные ранее книги и журналы. После была выполнена сортировка этого массива (рис. 1.5).

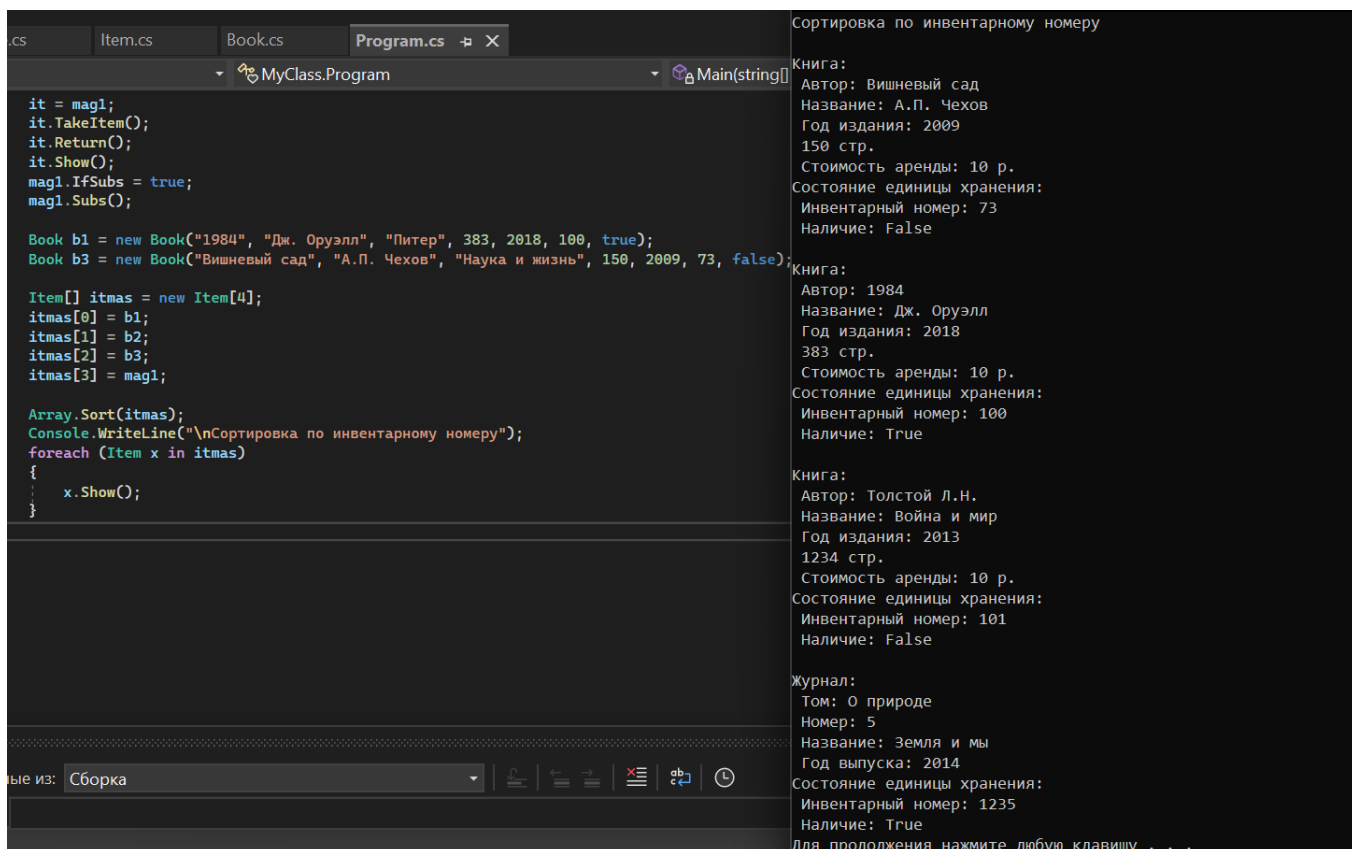


Рисунок 1.5 — Пример работы программы из упражнения 2

### 1.3 Упражнение 3

В последнем упражнении нужно было заменить абстрактный класс Progression из предыдущей лабораторной работы на интерфейс IProgression, определяющий поведение классов ArithmeticProgression и GeometricProgression.

Интерфейс IProgression также состоял из одного метода GetElement (рис. 1.6).

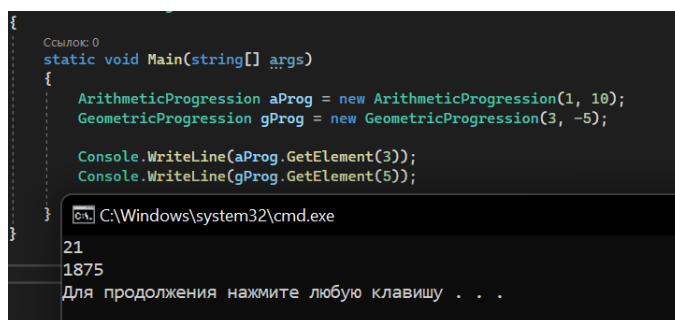
```

interface IProgression
{
    Ссылка: 4
    double GetElement(int k);
}

```

Рисунок 1.6 — Интерфейс IProgression

Изменять классы `ArithmeticProgression` и `GeometricProgression` не потребовалось. Результат работы программы на примере, который был в прошлой лабораторной работе показан на рис. 1.7.



```
{
    Ссылка: 0
    static void Main(string[] args)
    {
        ArithmeticProgression aProg = new ArithmeticProgression(1, 10);
        GeometricProgression gProg = new GeometricProgression(3, -5);

        Console.WriteLine(aProg.GetElement(3));
        Console.WriteLine(gProg.GetElement(5));
    }
}
```

C:\Windows\system32\cmd.exe

21  
1875  
Для продолжения нажмите любую клавишу . . .

Рисунок 1.7 — Пример работы программы из упражнения 3

## **ЗАКЛЮЧЕНИЕ**

В ходе этой лабораторной работы я ознакомился с интерфейсами и свойствами, расширив функциональность программы из прошлой лабораторной работы. Все задания были выполнены.