

ЛАБОРАТОРНАЯ РАБОТА 5. ПРОЕКТИРОВАНИЕ ИЕРАРХИИ КЛАССОВ

1. Цель и содержание

Цель лабораторной работы: изучить механизм организации наследования классов.

Задачи лабораторной работы:

- научиться объявлять производные классы;
- научиться создавать иерархии классов;
- научиться использовать механизм полиморфизма.

2. Теоретическая часть

2.1 Наследование реализации

Наследование реализации (implementation inheritance) означает, что тип происходит от базового типа, получая от него все поля-члены и функции-члены.

Синтаксис наследования реализации:

```
class ПроизводныйКласс: БазовыйКласс
{
    // Данные – члены и функции – члены
}
```

Если при определении класса не указан базовый класс, то C# предполагает, что базовым классом является System.Object.

2.2 Создание иерархии классов

При наследовании реализации производный класс наследует реализацию каждой функции базового типа, если только в его определении не указано, что реализация функции должна быть переопределена.

Определим следующую иерархию классов (рис. 5.1) и продемонстрируем, как наследуется реализация и как переопределяются свойств и методов.



Рисунок 5.1 – Иерархия классов.

Создадим код, описывающий данную иерархию. Определим базовый класс:

```

class Человек
{
    public Человек(string Ф, string И, string О, int Возр)
    {
        Фамилия = Ф; Имя = И; Отчество = О; Возраст = Возр;
    }

    public Человек()
    {
        Фамилия = "нет данных"; Имя = ""; Отчество = "";
        Возраст = 18;
    }

    public string Фамилия, Имя, Отчество;
    private DateTime ДатаРождения;

    public virtual string ФИО
    {
        get { return Фамилия + " " + Имя + " " + Отчество; }
    }

    public int Возраст
    {
        get { return DateTime.Now.Year - ДатаРождения.Year; }
        set
        {
            int ГодРождения = DateTime.Now.Year - value;

            ДатаРождения = Convert.ToDateTime(ГодРождения.ToString()+".01.01");
        }
    }
}

```

Обратите внимание на наличие двух конструкторов, механизм хранения возраста человека и ключевое слово `virtual` у свойства «ФИО». Ключевое слово `virtual` указывает, что данное свойство будет переопределено в производном классе.

Определим производный класс «Учитель»:

```

class Учитель: Человек
{
    public Учитель()
        :base()
    {
        УченоеЗвание = УченыеЗвания.Без_Звания;
        УченаяСтепень = УченыеСтепени.Без_Степени;
    }

    public Учитель(string Ф, string И, string О, int Возр, УченыеЗвания УЗ, УченыеСтепени УС)
        :base(Ф, И, О, Возр)
    {
        УченоеЗвание = УЗ;
        УченаяСтепень = УС;
    }

    public УченыеЗвания УченоеЗвание;
    public УченыеСтепени УченаяСтепень;

    public override string ФИО
    {
        get
        {
            return УченаяСтепень.ToString() + ", " + УченоеЗвание.ToString() + ", " + base.ФИО;
        }
    }
}

```

Следует обратить внимание на использование ключевого слова `override` у свойства «ФИО», которое указывает на то, что данное свойство имеет новую реализацию, отличающуюся от реализации базового класса. Определим второй производный класс «Студент»:

```
class Студент: Человек
{
    public Студент(string Ф, string И, string О, int Возр, Специальности Спец)
        :base(Ф, И, О, Возр)
    {
        Специальность = Спец;
    }

    public Специальности Специальность;

    public override string ФИО
    {
        get
        {
            return base.ФИО + ", " + Специальность.ToString();
        }
    }
}
```

В обоих производных классах следует обратить внимание на реализацию конструкторов производных классов, использование ключевого слова `base` в коде свойства «ФИО» и при объявлении конструкторов. Также интерес представляют типы данных, используемые для членов-данных: «Специальности», «УченыеЗвания», «УченыеСтепени». Вот определения данных типов-перечислений:

```
public enum УченыеЗвания
{
    Доцент,
    Профессор,
    Академик,
    Без_Звания
}

public enum УченыеСтепени
{
    Кандидат_Технических_Наук,
    Кандидат_ФизМат_Наук,
    Кандидат_Педагогических_Наук,
    Доктор_ФизМат_Наук,
    Без_Степени
}

public enum Специальности
{
    Информационные_Системы_И_Технологии,
    Безопасность_Информационных_Систем,
    Технология_Защиты_Информации,
    Психология,
    Нанозлектроника
}
```

Наконец, продемонстрируем использование объявленной иерархии классов. В программе объявим массив объектов типа «Человек». Следует понимать, что при объявлении такого массива, его элементам можно

присваивать объекты любого производного класса, причем каждый объект будет вести себя по-своему (за счет переопределения свойств и методов).

```
static void Main(string[] args)
{
    // Объявим массив людей:
    Человек[] mas = new Человек[6];

    // Заполним массив значениями разных (!) типов
    mas[0] = new Человек("Петров", "Петр", "Петрович", 10);

    mas[1] = new Студент("Коровов", "Сергей", "Викторович",
        19,
        Специальности.Информационные_Системы_И_Технологии);

    mas[2] = new Учитель("Николаев", "Евгений", "Иванович",
        30,
        УченыеЗвания.Доцент,
        УченыеСтепени.Кандидат_Технических_Наук);

    mas[3] = new Студент("Павлова", "Марина", "Андреевна",
        20,
        Специальности.Нанозлектроника);

    mas[4] = new Учитель("Дроздова", "Виктория", "Игоревна",
        50,
        УченыеЗвания.Профессор,
        УченыеСтепени.Доктор_ФизМат_Наук);

    mas[5] = new Человек("Сидоров", "Марк", "Захарович", 12);

    // Вывод информации
    for (int i = 0; i < mas.Length; i++)
    {
        Console.WriteLine(">>>>>>>>>>" + mas[i].GetType().Name);
        Console.WriteLine(mas[i].ФИО);
        Console.WriteLine("Возраст: " + mas[i].Возраст.ToString()+"\n");
        Console.WriteLine("\n");
    }

    Console.ReadKey();
}
```

В результате работы программы будет осуществлен следующий вывод:

разделе

«Теоретическое обоснование» данной лабораторной работы.

3. Постройте свою иерархию классов в соответствии с индивидуальным заданием. В результате выполнения лабораторной работы должны быть реализованы следующие механизмы:

- использование типа-перечисления (хотя бы одного);
- использование переопределенного свойства (хотя бы одного);
- использование переопределенного метода (хотя бы одного);
- использование вызова базового конструктора;
- использование вызова любого базового метода (отличного от конструктора).

4. . Во всех классах следует переопределить метод Equals, чтобы обеспечить сравнение значений, а не ссылок.

5. Функция Main должна содержать массив из элементов базового класса, заполненный ссылками на производные классы. В этой функции должно демонстрироваться использование всех разработанных элементов классов.

6. При защите работы укажите признаки присутствия полиморфного поведения в программе (реализация полиморфизма).

Индивидуальное задание.

Спроектируйте класс, наполните его требуемой функциональностью, продемонстрируйте работоспособность класса.

Вариант 1

Создать абстрактный класс Pair (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Money (деньги) и Complex (комплексное число).

В классе Money денежная сумма представляется в виде двух целых, в которых хранятся рубли и копейки соответственно. При выводе части числа снабжаются словами «руб.» и «коп.». В классе Complex предусмотреть при выводе символ мнимой части (i).

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 2

Создать абстрактный класс Triple (тройка значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Point3D (Трехмерная точка) и Circle (окружность).

В классе Point3D координаты представляются в виде трёх целых чисел. При выводе координаты заключаются в круглые скобки и разделяются запятыми. В классе Circle два числа – координаты центра окружности, третье - радиус.

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 3

Создать абстрактный класс Pair (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Point2D (точка на плоскости) и Polar (полярные координаты).

В классе Point2D координаты представляются в виде двух вещественных чисел. В классе Polar первое число – полярный радиус, второе – полярный угол, заданный в радианах, оба – вещественные числа. При выводе координаты заключаются в круглые скобки и разделяются запятыми.

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 4

Создать абстрактный класс Triple (тройка значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Date (Дата) и Time (время).

В классе Date дата представляется в виде трёх целых чисел: год, месяц, день. При выводе части даты разделяются знаком “/”. В классе Time время представляется в виде трёх целых чисел: часы, минуты, секунды. При выводе части даты разделяются знаком “:”.

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 5

Создать абстрактный класс Pair (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Point2D (точка на плоскости) и Cell (ячейка).

В классе Point2D координаты представляются в виде двух вещественных чисел. При выводе координаты заключаются в круглые скобки и разделяются запятыми. В классе Cell номер строки представляется в виде целого числа, номер столбца – латинской буквы. При выводе первый символ – буква столбца, второй – число.

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 6

Создать абстрактный класс `Pair` (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы `Size` (габариты) и `Gender` (пол/возраст).

В классе `Size` первое число – рост, второе – масса тела, оба вещественные. При выводе к данным добавляется единица измерения: к росту – «см», к массе тела – «кг». В классе `Gender` возраст в виде целого числа, пол – букв М, Ж, Н (не определен). При выводе к возрасту добавляются единицы измерения(лет).

Создать класс `Series` (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 7

Создать абстрактный класс `Pair` (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы `Age` (возраст) и `Weight` (вес).

В классе `Age` первое число – количество полных лет, второе – количество полных месяцев, оба целые. При выводе к данным добавляется единица измерения: к годам – «лет», к месяцам – «месяцев». В классе `Weight` первое число – количество килограмм, второе – грамм, оба целые. При выводе к добавляются единицы измерения(кг, г).

Создать класс `Series` (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 8

Создать абстрактный класс `Triple` (тройка значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы `Nutrients` (нутриенты) и `Person` (человек).

В классе `Nutrients` данные представляются в виде трёх вещественных чисел: белки, жиры, углеводы. При выводе к данным добавляется единица измерения – г. В классе `Person` рост и возраст – вещественные числа, пол – одна из букв М, Ж, Н (не определен). При выводе добавляются единицы измерения (см, лет).

Создать класс `Series` (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 9

Создать абстрактный класс `Triple` (тройка значений) с виртуальными

арифметическими операциями и методом вывода на экран. На его основе реализовать классы Publication (публикация) и Person (человек).

В классе Publication данные представляются в виде трёх целых чисел: год журнала, номер выпуска, номер страницы. В классе Person хранится id автора, его индекс Хирша, и количество цитирования публикаций, все - целые.

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 10

Создать абстрактный класс Triple (тройка значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Point3D (Точка в пространстве) и Polar3D (Полярные координаты точки в пространстве).

В классе Point3D координаты представляются в виде трёх вещественных чисел. При выводе координаты заключаются в круглые скобки и разделяются запятыми. В классе Polar3D первое число – полярные радиус, второе – долгота точки, третье – широта точки. Все числа – вещественные. При выводе координаты заключаются в круглые скобки и разделяются запятыми, для долготы и широты дописывается знак «°».

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 11

Создать абстрактный класс Pair (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Fraction (дробь) и Complex (комплексное число).

В классе Fraction дробь представляется в виде двух целых чисел, в которых хранятся числитель и знаменатель соответственно. При выводе части дроби разделяются знаком «/». В классе Complex предусмотреть при выводе символ мнимой части (i).

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Вариант 12

Создать абстрактный класс Pair (пара значений) с виртуальными арифметическими операциями и методом вывода на экран. На его основе реализовать классы Sum (сумма) и Difference (разность).

В классе Sum данные представляется в виде двух вещественных чисел, в которых хранятся первое и второе слагаемое соответственно. При выводе

части суммы разделяются знаком «+». В классе Difference данные представляется в виде двух вещественных чисел, в которых хранятся уменьшаемое и вычитаемое соответственно. При выводе части разности разделяются знаком «-».

Создать класс Series (набор), содержащий список (или массив) объектов этих классов в динамической памяти. Предусмотреть возможность вывода объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

5. Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы.
2. Цели лабораторной работы.
3. Экранные формы и листинг программного кода, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы в электронном виде прикрепляется на портал преподавателю.