

Классы. Полиморфизм

Цели работы:

- 1) изучение назначения и структуры классов;
- 2) изучение полиморфизма классов, виртуальных методов;
- 3) освоение области видимости членов класса;
- 4) получение навыков работы с наследуемыми классами.

Задание для выполнения лабораторной работы: разработать структуру класса и программу вычисления в соответствии с вариантом задания.

Программа должна включать:

- 1) объявление и реализация классов в соответствии с вариантом задания;
- 2) разграничение области видимости членов класса;
- 3) ввод исходных данных и их обработка;
- 4) разработку виртуальных функций;
- 5) вывод результатов выполнения программы.

Средства и оборудование: персональный компьютер, C#.

Теоретические основы:

Полиморфизм — возможность использования одного и того же имени для решения двух или более схожих, но технически разных задач.

По-умолчанию методы в классах считаются статическими методами. Статические методы используют механизм раннего связывания, то есть на этапе компиляции определяется класс, которому принадлежит метод. На следующем примере показано как работает статический метод при наследовании классов, если экземпляр класса объявлен типом базового класса:

```
class Program
{
    static void Main(string[] args)
    {
        Person s = new Student()
        {
            Surname = "Иванов";
            Group = "Б-XXX-1";
        }
        Console.WriteLine(s.Print()); // Print() вызывается из Person
    }
}
```

```

}

public class Person
{
    ... // объявление необходимых полей, методов и свойств
    public string Print()
    {
        return surname;
    }
}

public class Student
{
    ... // объявление необходимых полей, методов и свойств
    public string Print()
    {
        return string.Format("{0},{1}", base.Print(), group);
    }
}

```

Ожидалось, что метод будет вызван из класса-наследника, поэтому в объектно-ориентированное программирование было введено понятие полиморфизма. Полиморфизм вносит определенные конструкции, которые дают понять компилятору, что класс вызываемого метода будет определяться в ходе выполнения программы. Такой подход называется механизмом позднего связывания.

Полиморфизм реализуется посредством указания виртуальных методов. Для того, чтобы сделать метод виртуальным, необходимо до его объявления дописать зарезервированное слово **virtual**. Такая запись производится в классе-предке. Виртуальный метод попадает в таблицу виртуальных методов (ТВМ). Переопределяемый метод в классе-наследнике оформляется зарезервированным словом **override**. Пример использования виртуального метода:

```

class Program
{
    static void Main(string[] args)
    {
        Person s = new Student()
        {
            Surname = "Иванов";
            Group = "Б-XXX-1";
        }
        Console.WriteLine(s.Print()); // Print() вызывается из Student
    }
}

```

```

    }
}

public class Person
{
    ... // объявление необходимых полей, методов и свойств
    virtual public string Print()
    {
        return surname;
    }
}

public class Student
{
    ... // объявление необходимых полей, методов и свойств
    override public string Print()
    {
        return string.Format("{0},{1}", base.Print(), group);
    }
}

```

Абстрактный метод — метод, у которого есть только описание, но отсутствует объявление (отсутствует тело метода). Пустой метод, введенный в базовый класс, оформляется зарезервированным словом **abstract**. Только виртуальный метод могут быть абстрактными.

Абстрактный класс — класс, содержащий хотя бы один абстрактный метод. Рекомендуется в определенных случаях объявлять абстрактный класс в качестве базового класса. Запрещено объявлять экземпляры абстрактных классов. Если класс-наследник имеет хотя бы один абстрактный метод (т.е. метод не переопределялся в наследнике), то такой класс также считается абстрактным. Пример:

```

class Program
{
    static void Main(string[] args)
    {
        Circle c = new Circle();
        c.Draw();
    }
}

public class Shape // абстрактная геометрическая фигура
{

```

```

    public abstract void Draw(); // отсутствует тело метода
}

```

```

public class Circle: Shape
{
    ... // объявление необходимых полей, методов и свойств
    public override void Draw();
}

```

Полиморфизм позволяет осуществлять одно и тоже поведение для всей иерархии классов. Далее приведен пример создания списка геометрических объектов, заполненного случайным образом. Иерархия классов приведена на рисунке 1.

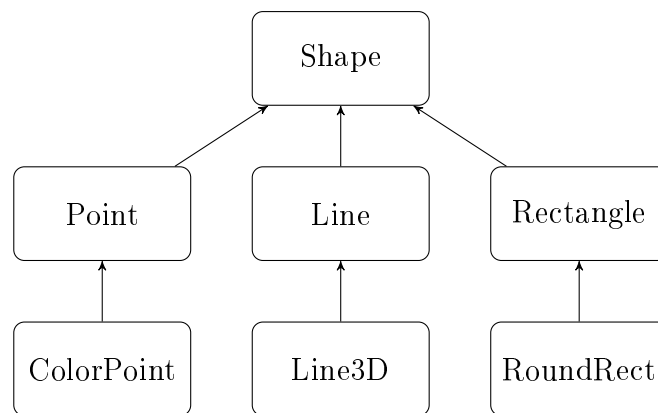


Рисунок 1 — Иерархия классов геометрических фигур

Пример. Создание списка геометрических объектов:

```

class Program
{
    static void Main(string[] args)
    {
        var rand = new Random();
        var list = new List<Shape>();
        Shape s;

        for (int i = 0; i < 100; i++)
        {
            switch(rand.Next(6))
            {
                case 0: s = new Point(); break;
                case 1: s = new ColorPoint(); break;
                case 2: s = new Line(); break;
                case 3: s = new Line3D(); break;
            }
        }
    }
}

```

```

        case 4: s = new Rectangle(); break;
        case 5: s = new RoundRect(); break;
    }
    list.Add(s);
}

foreach (var item in list)
    item.Draw();

list.Clear();
}
}

```

Варианты заданий

1. Разработать программу, демонстрирующую работу полиморфизма классов. Разработать иерархию классов и их экземпляров. Базовый класс должен содержать методы, задающие схожее поведение для всей иерархии классов. Наследуемый класс должен расширять базовый класс путем добавления как минимум одного поля и соответствующих методов. Варианты заданий взять из задания ?? предыдущей лабораторной работы.
2. Разработать иерархию классов со вложенным внутри полем типа отдельно от описанного класса. Функции-члены установки полей должны проверять корректность задаваемых параметров. Класс должен иметь: конструктор пользователя, свойства установки значений полей, функцию-член вывода на экран (перевода в тип String) по указанному шаблону, методы работы с полями. Каждый класс должен быть объявлен в отдельном модуле. Варианты заданий взять из задания ?? предыдущей лабораторной работы.

Контрольные вопросы

1. Объясните правила вызова конструкторов.
2. Расскажите назначение полиморфизма классов.
3. Какие методы называются виртуальными, а какие — статическими.
4. Расскажите об абстрактных методах.
5. В чем отличие динамических методов от виртуальных.

Распределение вариантов заданий

Таблица 1 — Распределение вариантов

		последняя цифра года поступления в университет									
		0	1	2	3	4	5	6	7	8	9
порядковый номер студента в списке группы	1	1	6	11	16	1	6	11	16	1	6
	2	2	7	12	17	2	7	12	17	2	7
	3	3	8	13	18	3	8	13	18	3	8
	4	4	9	14	19	4	9	14	19	4	9
	5	5	10	15	20	5	10	15	20	5	10
	6	6	11	16	1	6	11	16	1	6	11
	7	7	12	17	2	7	12	17	2	7	12
	8	8	13	18	3	8	13	18	3	8	13
	9	9	14	19	4	9	14	19	4	9	14
	10	10	15	20	5	10	15	20	5	10	15
	11	11	16	1	6	11	16	1	6	11	16
	12	12	17	2	7	12	17	2	7	12	17
	13	13	18	3	8	13	18	3	8	13	18
	14	14	19	4	9	14	19	4	9	14	19
	15	15	20	5	10	15	20	5	10	15	20
	16	16	1	6	11	16	1	6	11	16	1
	17	17	2	7	12	17	2	7	12	17	2
	18	18	3	8	13	18	3	8	13	18	3
	19	19	4	9	14	19	4	9	14	19	4
	20	20	5	10	15	20	5	10	15	20	5