Класс, который объявляется в пределах фигурных скобок другого класса, называется **вложенным классом**.

Объект вложенного класса можно объявить в случае, когда вложенный класс объявлен как видимый. Это значит, что видимый вложенный класс должен быть объявлен с любым модификатором доступа кроме private.

В нестатическом классе может быть объявлен статический класс (с ключевым словом static).

public class Outer

{

// внутренние переменные класса Outer

public int d;

static public int sd;

// статический вложенный класс Inner

public static class Inner

{

// только статические переменные могут объявляться в статическом классе

public static int sd; // статическая внутренняя переменная класса Inner

}

}

// доступ к статической переменной статического класса Outer.Inner

Outer.Inner.sd = 45;

// создать объект (экземпляр) класса Outer

Outer o = new Outer();

o.d = 30; // есть доступ к переменным экземпляра класса через объект

Outer.sd = 102; // доступ к статической переменной класса Outer

// создавать объект статического класса нельзя

//Outer.Inner i = new Outer.Inner(); - это ошибка!

Да, можно. В статическом классе можно объявлять нестатические вложенные классы. Однако, в статическом классе нельзя объявлять нестатические переменные и методы.

// класс Outer - статический

public static class Outer

{

// внутренние переменные класса Outer

// только статические экземпляры можно объявлять в статическом классе

// public int d; - ошибка!

static public int sd;

// нестатический вложенный класс Inner - может объявляться в статическом классе

public class Inner

{

public int d; // нестатическая переменная

public static int sd; // статическая внутренняя переменная класса Inner

}

}

// использование нестатического класса в статическом классе

// доступ к статической переменной sd класса Outer.Inner

Outer.Inner.sd = 230;

// доступ к статической переменной sd класса Outer

Outer.sd = 132;

// Нельзя создавать экземпляр (объект) статического класса

// Outer o = new Outer(); // ошибка, поскольку Outer - статический класс

// создать экземпляр вложенного класса Outer.Inner

Outer.Inner i = new Outer.Inner();

і.d = 323; // доступ к нестатической переменной d класса Outer.Inner

Для перегрузки оператора служит ключевое слово operator, определяющее операторный метод, который, в свою очередь, определяет действие оператора относительно своего класса. Существуют две формы операторных методов (operator): одна - для унарных операторов, другая - для бинарных. Ниже приведена общая форма для каждой разновидности этих методов:

// Общая форма перегрузки унарного оператора.

public static возвращаемый\_тип operator op(тип\_параметра операнд)

{

// операции

}

// Общая форма перегрузки бинарного оператора.

public static возвращаемый\_тип operator op(тип\_параметра1 операнд1,

тип\_параметра2 операнд2)

{

// операции

}

Операции не подлежащие перегрузке:

[] (но есть

() можно определить новые операторы

преобразования

+=, --=, \*=, /=, %=, &=, |=, ^=, <<=, >>= (но

получаем автоматически в случае перегрузки

бинарной операции)

&&,||

=, . ?:, ??, ->

f(x), as, checked

unchecked,

default, delegate, is, new, sizeof

typeof

Методы расширения позволяют "добавлять" методы в существующие типы без создания нового производного типа, перекомпиляции и иного изменения первоначального типа. Методы расширения представляют собой разновидность статического метода, но вызываются так же, как методы экземпляра в расширенном типе.

Суть методов расширения заключается в том, что они позволяют добавить новый функционал к уже существующим классам, при этом не модифицируя сами эти классы.

Методы расширения должны находится в статическом классе и сами по себе должны быть статическими.

Перегрузка операторов – процесс, который позволяет определить назначение оператора по отношению к создаваемому классу.

Перегружать мы можем как бинарные операторы (два параметра), так и унарные (один параметр)