Иногда преобразование типов требуется произвести во время выполнения, но не

генерировать исключение, если исход этого преобразования окажется неудачным, что

вполне возможно при приведении типов.

**Применение оператора as**

Для этой цели служит оператор as, имеющий следующую общую форму:

выражение as тип

где *выражение* обозначает отдельное выражение, преобразуемое в указанный *тип.*

Если исход такого преобразования оказывается удачным, то возвращается ссылка на

*тип,* а иначе — пустая ссылка. Оператор as может использоваться ***только*** для преобразования

ссылок, идентичности, упаковки, распаковки.

В некоторых случаях оператор as может служить удобной альтернативой оператору

is. В качестве примера рассмотрим следующую программу, в которой оператор is

используется для предотвращения неправильного приведения типов.

(***glava17\_2***)

class A { }

class B : A { }

class CheckCast

{

static void Main()

{

A a = new A();

B b = new B();

//check if can cast to B type

if (a is B)

b = (B)a;

else //if not, skip cast

b = null;

if (b == null)

Console.WriteLine("Cast types b = (B) is not allowed.");

else

Console.WriteLine("Cast types b = (B) allowed.");

}

}

Как следует из результата выполнения приведенной выше программы, тип объекта

а не совместим с типом В, и поэтому его приведение к типу В не допустимо и предотвращается

в условном операторе if. Но такую проверку приходится выполнять в два

этапа. Сначала требуется убедиться в обоснованности операции приведения типов,

а затем выполнить ее. Оба этапа могут быть объединены в один с помощью оператора

as, как демонстрирует приведенная ниже программа.

(***glava17\_2.1***)

class A { }

class B : A { }

class CheckCast

{

static void Main()

{

A a = new A();

B b = new B();

b = a as B; //do cast types, if possible

if (b == null)

Console.WriteLine("Cast types b = (B) is not allowed.");

else

Console.WriteLine("Cast types b = (B) allowed.");

}

}

В данном варианте программы в одном и том же операторе as сначала проверяется

обоснованность операции приведения типов, а затем выполняется сама операция приведения

типов, если она допустима.