*Динамическая идентификация типов* представляет собой механизм, позволяющий определить тип данных во время выполнения программы. Рефлексия — это средство для получения сведений о типе данных. Используя эти сведения, можно конструировать и применять объекты во время выполнения. Это довольно эффективное средство, поскольку оно дает возможность расширять функции программы динамически, т.е. в процессе ее выполнения. *Атрибут* описывает характеристики определенного элемента программы на С#. Атрибуты можно, в частности, указать для классов, методов и полей. Во время выполнения программы разрешается опрашивать атрибуты для получения сведений о них. Для этой цели в атрибутах используется

динамическая идентификация типов и рефлексия.

**Динамическая идентификация типов (dynamic\_cast C++)**

Динамическая идентификация типов (RTTI) позволяет определить тип объекта во время выполнения программы. Она оказывается полезной по целому ряду причин. В частности, по ссылке на базовый класс можно довольно точно определить тип объекта, доступного по этой ссылке. Динамическая идентификация типов позволяет также проверить заранее, насколько удачным будет исход приведения типов, предотвращая исключительную ситуацию в связи с неправильным приведением типов. Кроме того, динамическая идентификация типов является главной составляющей рефлексии.

Для поддержки динамической идентификации типов в С# предусмотрены три

ключевых слова: is, as и typeof. Каждое из этих ключевых слов рассматривается далее

по очереди.

**Проверка типа с помощью оператора is**

Конкретный тип объекта можно определить с помощью оператора is. Ниже приведена

его общая форма:

выражение is тип

где *выражение* обозначает отдельное выражение, описывающее объект, *тип* которого

проверяется. Если *выражение* имеет совместимый или такой же тип, как и проверяемый

*тип,* то результат этой операции получается истинным, в противном случае —

ложным. В операторе is оба типа определяются как совместимые, если

они одного и того же типа или если предусмотрено преобразование ссылок, упаковка

или распаковка.

(***glava17\_1***)

class A { }

class B : A { }

class UseIS

{

static void Main()

{

A a = new A();

B b = new B();

if (a is A)

Console.WriteLine("a has type A");

if (b is A)

Console.WriteLine("b has dependies from A, it based on A");

if (a is B)

Console.WriteLine("Wount show, a is not derived from B");

if (b is B)

Console.WriteLine("B has type B");

if (a is object)

Console.WriteLine("a has type object");

}

}