**1) Рефлексия** представляет собой процесс выявления типов во время выполнения приложения.

Получение типа:

2) Чтобы управлять типом и получать всю информацию о нем, нам надо сперва получить данный тип. Это можно сделать тремя способами: с помощью оператора typeof, с помощью метода GetType() класса Object и применяя статический метод Type.GetType().

3)Основной функционал рефлексии сосредоточен в пространстве имен System.Reflection. В нем мы можем выделить следующие основные классы:

**Assembly**: сборка манипулирование этой сборки

**AssemblyName**: информация о сборке

**EventInfo**: информация о событии

**FieldInfo**: информация о поле

**MethodInfo**: информация о методе

**PropertyInfo**: информация о свойстве

**ConstructorInfo**: информация о конструкторе

**ParameterInfo**: класс, хранящий информацию о параметре метода

4)System.type – класс, позволяет получить информацию о членах типа.

Методы:

GetConstructors();

GetEvents();

GetFields();

GetInterfaces();

GetMethods();

GetProperties();

Свойства:

IsAbstract;

IsArray;

IsClass;

IsEnum;

IsInterface;

5) Раннее связывание – связанное с формированием кода на этапе компиляции. При раннем связывании, программный код формируется на основе известной информации о типе (класс) ссылки.

Позднее связывание – связанное с формированием кода на этапе выполнения. Если в иерархии классов встречается цепочка виртуальных методов (с помощью слов virtual, override), то компилятор строит так называемое позднее связывание.

6) Чтобы динамически загрузить сборку в приложение, надо использовать статические методы Assembly. LoadFrom() или Assembly. Load().

8) Перечисление **BindingFlags** может принимать различные значения:

**DeclaredOnly:** получает только методы непосредственно данного класса, унаследованные методы не извлекаются

**Instance**: получает только методы экземпляра

**NonPublic:** извлекает не публичные методы

**Public:** получает только публичные методы

**Static:** получает только статические методы