Ref1

Metaclasses, References to classes, Class methods(static methods), virtual

constructors, RTTI.

**Метаклассы**

Метаклассы – это классы , экземплярами которых являются классы. Некоторые языки поддерживающие метаклассы:

Lisp,

Delphi/Object Pascal(изучаемый в данной дисциплине),

Objective C,

Ruby,

Smalltalk.

Также есть специальные языки, ориентированные на использование этой концепции для метапрограммирования, например: MPS от JetBrains, R# и другие.

**Метапрограммирование,** в свою очередь, - это вид программирования, связанный с созданием программ, которые порождают другие программы в результате своей работы, а также управляют ими. Данная концепция, как и языки специально для этого созданные, не получили широкого распространения.

В языке Delphi/Objective Pascal для поддержки метаклассов существует специальный тип данных: ссылка на класс.

type Ttype = class of TAnyType;

значением экземпляра метакласса могут быть все потомки TAnyType включая его самого.   
Все ссылки на классы приводятся к типу TClass, являющимся предком для любого типа-метакласса.

Концептуально метакласс является частью системы метабъектов, описанным в Common Lisp Object System, на ряду с ссылками на методы и интерфейсами.

**Методы классов** в Delphi оперируют не экземпляром объекта, а непосредственно классом, неявный параметр Self, который передается в метод класса содержит ссылку на класс, (механизм описанный ранее), в теле метода класса есть доступ лишь к другим методам класса. Однако методы класса доступны также и экземплярам класса. В языках типа C# и С++ схожими являются статические методы, однако в отличие от Delphi их нельзя объявить виртуальными.

Из этого также вытекает следующая особенность: Delphi позволяет использовать **виртуальные конструкторы**, с их помощью можно создавать объекты классов, тип которых неизвестен на этапе компиляции. В С++ и С# это запрещено конструктор может быть вызван только у конкретного типа.

**RTTI (runtime type information)** – механизм позваляющий получить информацию о типе объекта на этапе выполнения программы. Специфичен для языков семейства C, а также Delphi Это могут быть как стандартные типы так и пользовательские а так же обобщенные(generic). RTTI это одна из реализаций коцепции **Интерспецкии типов**. Суть которой заключается как раз в возможности получать информацию о типе, свойствах, методах объекта на этапе выполнения программы. В С++ RTTI доступна лишь для полиморфных классов, содержащих хотя бы один виртуальный метод. Type interspection(Интроспекции типов) в с++ реализована через dynamic\_cast, позволяющий произвести безопасное приведение типа например:

if (Person \*p = dynamic\_cast<Person \*>(obj)) { p->walk(); }

typeid позволяет произвести проверку типов. Не во всех компиляторах поддержка RTTI включена по умолчанию.

В Delphi все классы унаследованы от базового TObject. TObject реализует возможность интроспекции типов с помощью свойств ClassType, ClassName  
Оператор Is реализует проверку типов, As – безопасное приведение типов

Delphi в своей библиотеке форм повсеместно использует возможности интроспекций типов, и работы с их метаданными.

Механизм интроспекции типов в C# пошел куда дальше. Он получил название Reflection, и в допольнение ко всему позволяет также изменять метаданные во время выполнения.

Источники:

Статья о R# и метапрограммировании на RSDN: <http://www.rsdn.ru/projects/rsharp/article/rsharp_mag.xml>  
CommonLisp Object System: <http://alu.org/mop/index.html>

Bjarne Stroustrup: A history of c++

MSDN