1. Термин обобщение означает параметризированный тип. Особая роль параметризированных типов состоит в том, что они позволяют создавать классы, структуры, интерфейсы, методы и делегаты, в которых обрабатываемые данные указываются в виде параметра.
2. Ошибки нет.
3. Ограничения задаются с помощью контекстного ключевого слова where
4. С параметром типа может быть связано несколько ограничений. В этом случае ограничения указываются списком через запятую.
5. В качестве ограничений мы можем использовать следующие типы:

Классы

Интерфейсы

class - универсальный параметр должен представлять класс

struct - универсальный параметр должен представлять структуру

new() - универсальный параметр должен представлять тип, который имеет общедоступный (public) конструктор без параметров

1. ограничение на базовый класс.
2. Ограничение ссылочного типа
3. Ограничение типа значения.
4. Обобщенные классы могут входить в иерархию классов аналогично необобщенным классам. Следовательно, обобщенный класс может действовать как базовый или производный класс. Главное отличие между иерархиями обобщенных и необобщенных классов заключается в том, что в первом случае аргументы типа, необходимые обобщенному базовому классу, должны передаваться всеми производными классами вверх по иерархии аналогично передаче аргументов конструктора. В производном классе следует непременно указывать параметры типа, требующиеся его обобщенному базовому классу, даже если этот производный класс не обязательно должен быть обобщенным. Разумеется, в производный класс можно свободно добавлять его собственные параметры типа, если в этом есть потребность.
5. Иногда возникает необходимость присвоить переменным универсальных параметров некоторое начальное значение, в том числе и null. Но  
   напрямую мы его присвоить не можем: В этом случае нам надо использовать оператор default(T). Он присваивает ссылочным типам в качестве значения null, а типам значений -  
   значение 0:
6. При типизации обобщенного класса определенным типом будет создаваться свой набор статических членов. Например, в классе Account определено следующее статическое поле:

class Account { public static T session;

public T Id { get; set; }

public int Sum { get; set; }

}

Теперь типизируем класс двумя типами int и string:

Account account1 = new Account { Sum = 5000 }; Account.session = 5436;

Account account2 = new Account { Sum = 4000 }; Account.session = "45245";

Console.WriteLine(Account<int>.session); // 5436 Console.WriteLine(Account<string>.session); // 45245

В итоге для Account и для Account будет создана своя переменная session.

1. public interface ISort where T : struct { void ReWrite(); }
2. Подобно классам, структуры также могут быть обобщенными. Они очень похожи на обобщенные классы, за исключением возможности наследования.
3. Классы File и FileInfo. Класс FileStream. Классы StreamReader и StreamWriter. Работа с бинарными файлами- классы BinaryWriter и BinaryReader Создание и чтение сжатых файлов - классы GZipStream и DeflateStream.