**List<T> 泛型集合** **非静态 ===>对比ArraryList集合**

因为List<T>是非静态类，所以使用的时候我们需要进行实例化，创建他的对象

微信截图_20171213204449

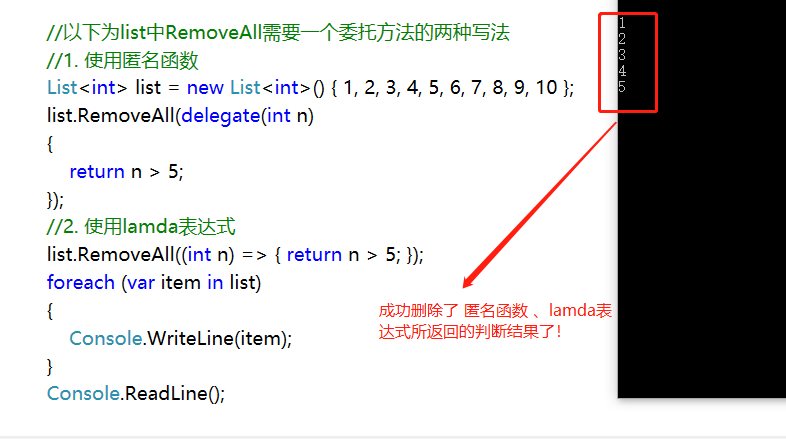
**注意：**

1. **List<T>，T代表是的是类型，因为泛型集合的类型固定性，当我们在T里面写什么类型，那么这个集合就只能存储什么类型的元素，当然，如果T写成object类型，那么就跟ArraryList一摸一样了，里面都是存放的对象了，然后输入跟输出都要遵行里式转换原则！**

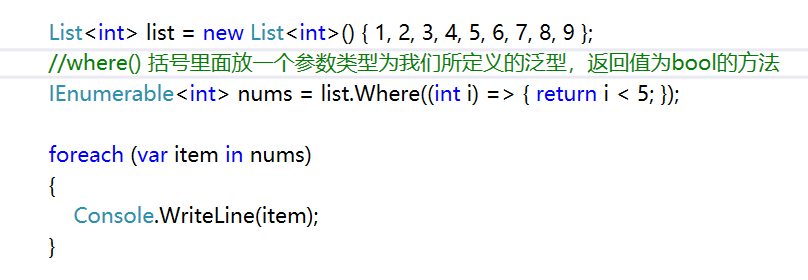
**2. 当我们为集合添加元素后，我们可以通过list[下标]来访问该下标中元素的值，也可以通过list[下标] = value来改变该下标中元素的值**

泛型集合的方法：

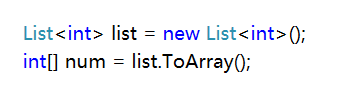
1. list.Add()；添加单个元素
2. list.AddRange()；添加一个集合/数组
3. list.Clear()；把集合中的元素清空
4. list.Remove(object value); 删除单个元素，写谁删谁，删除元素后面的元素的下标往前增
5. list.RemoveAt(int index); 删除指定下标的单个元素，下标从0开始，删除元素后面的元素的下标往前增
6. list.RemoveRange(int index,int count); 删除由指定下标开始长度为count中的元素，index下标从0开始，长度从count从1开始，删除元素后面的元素的下标往前增
7. list.RemoveAll(委托所指向的方法，参数为：T，返回值为：bool);删除委托所指向的函数所判断的需要从集合中删除的元素，函数的return结果为true则删除，false则无视



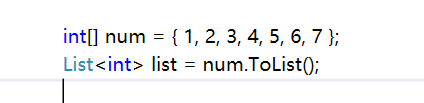
1. list.where(委托所指向的方法，参数为：T，返回值为：bool);查找委托所指向的函数所判断的需要返回的元素，函数的return结果为true则找到，false则不需要



1. list.Insert(int index,插入的元素)；在指定下标插入个元素，原先在此下标的元素，下标往后排
2. list.InsertRange(int index ,插入的集合)；在指定位置的下标插入一个集合，原先在此位置的元素的下标往后排
3. bool list.Contains()；判断该集合中是否存在该元素，返回true/false
4. list.Reverse()；把集合中的元素反向输出
5. list.Sort()；把集合中的元素升序排序
6. list.Count()；获取该集合目前的长度
7. list.CapCity()；获取该集合能包含的长度
8. **T[] list.ToArrary()；把集合转换成数组，T[]就代表返回一个数组，T就代表集合的类型，集合是什么类型，返回的数组就是什么类型**

****

1. **List<T> 数组名.ToList()；把数组转换成集合，List<T>就表示返回一个集合，T就表示该集合的类型，数组是什么类型转换的集合就是什么类型**

****