数组在内存中是连续的内存空间,并且长度固定
集合的长度不固定,可以装任意多个元素.

常用泛型集合

命名空间:using System.Collections.Generic;

List < T >

列表-按顺序存储数序,通过索引访问元素

```
static void Main(string[] args)
        {
            List<int> list = new List<int>();//定义一个用于存储int的列表
            list.Add(2);
            list.Add(4);
            list.Add(1);
            Console.WriteLine(list[0]);
            //遍历输出元素
            for (int i = 0; i < list.Count; i++)
            {
                Console.WriteLine(list[i]);
            }
        }
```

```
List常用API:
list.Remove();//移除元素 ,导致后面的元素全部往前移,性能不好
list.RemoveAt();//根据下标移除元素
list.Insert();//插入,导致后面的元素往后移,性能不好
list.InsertRange();//插入一批数据
list.Clear();//移除所有元素
list.IndexOf();//返回元素的索引
list.Contains();//是否包含某个元素
```

DIctionary< T >

字典-以键值对存储数据

```
static void Main(string[] args)
        {
            //创建字典对象
            Dictionary<string, string> dic = new Dictionary<string, string
            //添加元素
            dic.Add("主机","192.168.2.254");
            dic.Add("用户名","the9");
            dic.Add("密码","admin");
            dic.Add("端口","21");
            //获取数据
            Console.WriteLine(dic["主机"]);
            //修改数据
            dic["主机"] = "192.168.1.254";
            //遍历
            foreach (var item in dic)
            {
                Console.WriteLine(item.Key);//输出键
                Console.WriteLine(item.Value);//输出值
            }
        }
```

```
    Dictonary常用API:
    ContainsKey();//返回知否包含某个键
    ContainsValue();//是否包含某个值
    Remove();//移除
    Clear();//全部移除
TryGetValue();//尝试获取值 返回bool 通过out参数接收结果
```

Stack< T >

栈-先进后出

```
        static void Main(string[] args)
        {
            Stack<int> stack = new Stack<int>();
```

```
            //进栈
            stack.Push(4);
            stack.Push(5);
            stack.Push(9);
            //出栈
            //Console.WriteLine(stack.Pop());//输出9    (后进先出)
            //Console.WriteLine(stack.Pop());//输出5
                                //Console.WriteLine(stack.Peek());//返回栈
            //循环出栈
            while (stack.Count>0)
            {
                Console.WriteLine(stack.Pop());// 连续输出 9 5 4
            }
        }
```

Queue< T >

队列-先进先出

```
static void Main(string[] args)
        {
            Queue<string> queue = new Queue<string>();
            //进队列
            queue.Enqueue("A");
            queue.Enqueue("X");
            queue.Enqueue("K");
            //出队列
            //Console.WriteLine(queue.Dequeue());//输出A
            //Console.WriteLine(queue.Dequeue());//输出X
            //Console.WriteLine(queue.Peek());//返回队列第一个元素但不移除
            //循环出队
            while (queue.Count>0)
            {
                Console.WriteLine(queue.Dequeue()); //连续输出 A X K
            }
        }
```

常用非泛型集合 (用Object存储数据)

> 命名空间:using System.Collections;
> 所有数据以Object类型存储,使用的时候需要转成具体类型,如果存储值类型数据会发生装箱和拆箱操作

ArrayList

> 和List<T>很相似

```
        static void Main(string[] args)
        {
            ArrayList list = new ArrayList(); //创建ArrayList对象
            list.Add(5);
            list.Add(6);
            list.Add("ABC");//可以同时存储多种数据类型
            int sum = (int)list[0] + (int)list[1];//使用的时候需要转成具体类

            /*面试常问:List和ArrayList的区别
             * 1.List是泛型集合可以在定义时候通过泛型参数指定要存储的数据类型 A
             * 数据以Object类型存储.
             * 2.ArrayList存储值类型数据 会发生装拆箱操作 而 List不会
            */
        }
```

HashTable

> 和Dictonary<T>相似

```
  static void Main(string[] args)
        {
            Hashtable table = new Hashtable();
            table.Add("主机", "192.168.2.254");
            table.Add("用户名", "the9");
            table.Add("端口", 21);
            table.Add("密码", "admin");
            string ip = (string)table["主机"];//使用的时候需要转成具体类型
```

```
            int port = (int)table["端口"];
        }
```

Stack

和Stack<T>相似

Queue

和Queue<T>相似

作业

> 1.使用数组实现一个存整数的MyList类 功能类似 List
> 至少包含如下方法
> Add()
> RemoveAt()
> Contains()
> 2.自学索引器并为自己的集合添加索引器
> 快捷键 index tab tab