引言:

如果选择用数组存储数据会存在以下问题？

1.数组有类型限制

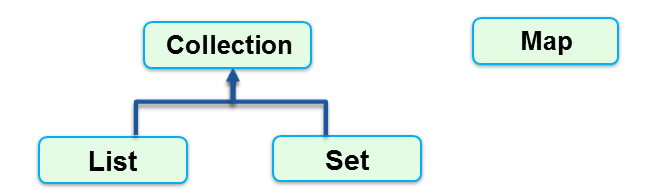
2.数组长度限制(如果存储内容不确定，导致数组浪费空间或才空间不足)

使用集合解决以上问题

1. java集合框架
2. java集合框架(java 集合API)提供一套性能优良、使用方便的接口和类。它们位于java.util包中. 注意:api是接口和类的统称
3. java集合框架的介绍

collection接口用于存储一组值的集合、而map用于存储一组键值的集合.

2．1java集合框架的接口分类及介绍:



* **Collection 接口存储一组不唯一，无序的对象**
* **List 接口存储一组不唯一，有序（插入顺序）的对象**
* **Set 接口存储一组唯一，无序的对象**
* **Map接口存储一组键值对象，提供key(不能重复)到value的映射**

1. **java集合的使用步骤**

**3.1创建集合对象**

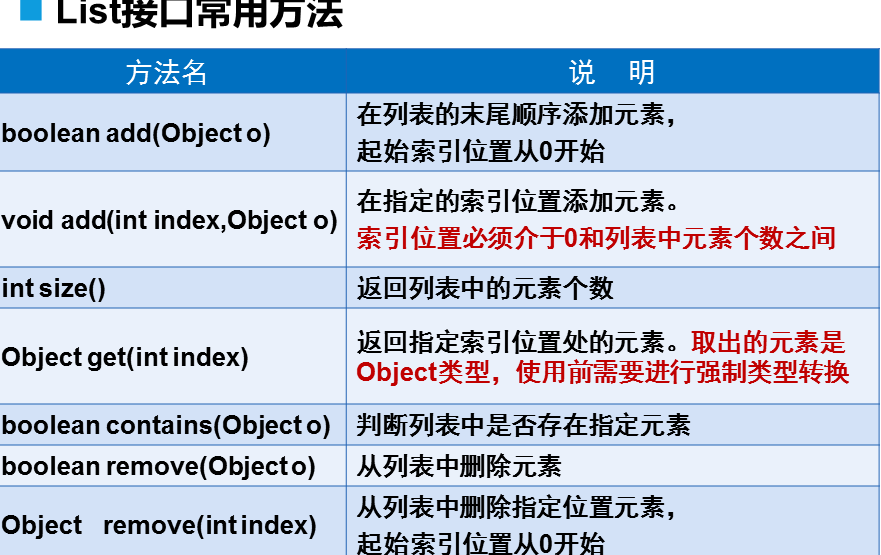
**3.2通过集合对象的方法操作集合**

1. **list接口的使用**
2. list接口的实现类:ArrayList、LinkedList

**ArrayList实现了长度可变的数组，在内存中分配连续的空间。遍历元素和随机访问元素的效率比较高**

**LinkedList采用链表存储方式。插入、删除元素时效率比较高**

1. List接口常用方法



示例: 使用ArrayList集合类

|  |
| --- |
| *//1.创建ArrayList集合对象 /// ArrayList list=new ArrayList();* List list=**new** ArrayList(); *//面向接程口编程  //2.使用集合的方法操作集合  //2.1添加元素* list.add(**"张三"**);  list.add(**"李四"**);  list.add(**"马明伍"**);  *//2.2获取集合的长度* **int** len=list.size();  System.***out***.println(**"长度是:"**+len);  *//2.3获取集合元素* Object o=list.get(0);  System.***out***.println(o);  *//2.4获取所有元素* **for**(**int** i=0;i<list.size();i++){  System.***out***.println(list.get(i));  }  *//2.5判断某个元素在集合中是否出现* **if**(list.contains(**"张三1"**))  System.***out***.println(**"已存在"**);  **else** System.***out***.println(**"不存在"**);  *//2.6移除元素  //list.remove(值); //通过元素移除  //list.remove(0); //通过位置移除* list.remove(1); |

1. (了解)LinkedList特殊的方法:提供一组首尾添加，移除、取值的方法



1. set接口的使用. （类似于List接口 ）
2. set接口实现类:HashSet、TreeSet
3. 遍历集合的方法
4. (了解)使用迭代器遍历集合的步骤

|  |
| --- |
| *//1.1获取集合迭代器对象* Iterator iterator=集合对象.iterator(); *//1.2使用迭代器遍历元素* **while**(iterator.hasNext()){ *//判断有没有下一个元素* System.***out***.println(iterator.next()); *//获取当前位置的值，并将指针下移* } |

1. 使用foreach遍历集合
   1. *foreach循环的语法结构***for** (元素类型 变量名:集合|数组名) {  
       循环体.....  
      }
2. (推荐)泛型集合
3. 什么是泛型:泛指某一类型
4. 定义泛型集合

定义List泛型集合

List<Student> list=**new** ArrayList<Student>(); *//类型安全*

注意:泛型只能是类类型

3. *包（封）装类:每个基本类型都对应一个类类型  
Integer int  
 Character char  
 Short short  
 Double double  
 Long long  
 Float float  
 Boolean boolean  
 Byte byte*