

1.对象的存储：①数组（基本数据类型 & 引用数据类型） ②集合（引用数据类型）

>数组存储数据的弊端：长度一旦初始化以后，就不可变；真正给数组元素赋值的个数没有现成的方法可用。

2.集合框架

Collection接口：方法：①add(Object obj),addAll(Collection coll),size(),clear(),isEmpty();

②remove(Object obj),removeAll(Collection coll),retainAll(Collection coll),equals(Object obj),contains(Object obj)
containsAll(Collection coll),hashCode()

③ iterator(),toArray();

* |-----List接口：存储有序的，可以重复的元素。---相当于“动态”数组

>新增的方法：删除remove(int index) 修改set(int index,Object obj) 获取get(int index)插入add(int index,Object obj)

>添加进List集合中的元素（或对象）所在的类一定要重写equals()方法

|-----ArrayList（主要的实现类）

|-----LinkedList（更适用于频繁的插入、删除操作）

|-----Vector（古老的实现类，线程安全的，但效率要低于ArrayList）

* |-----Set接口：存储无序的，不可重复的元素。---相当于高中的“集合”概念

>Set使用的方法基本上都是Collection接口下定义的。

>添加进Set集合中的元素所在的类一定要重写equals() 和 hashCode()。要求重写equals() 和 hashCode()方法保持一致。

>1.无序性：无序性！= 随机性。真正的无序性，指的是元素在底层存储的位置是无序的。

>2.不可重复性：当向Set中添加进相同的元素的时候，后面的这个不能添加进去。

|-----HashSet（主要的实现类）

|-----LinkedHashSet(是HashSet的子类，当我们遍历集合元素时，是按照添加进去的顺序实现的；频繁的遍历，较少的添加、

|-----TreeSet（可以按照添加进集合中的元素的指定属性进行排序）

>要求TreeSet添加进的元素必须是同一个类的！

>两种排序方式：自然排序 [

定制排序：

>两种排序方式：自然排序：①要求添加进TreeSet中的元素所在的类implements Comparable接口

②重写compareTo(Object obj)，在此方法内指明按照元素的哪个属性进行排序

③向TreeSet中添加元素即可。若不实现此接口，会报运行时异常

定制排序：①创建一个实现Comparator接口的实现类的对象。在实现类中重写Comparator的compare(Object o1,Object o2)方法

②在此compare()方法中指明按照元素所在类的哪个属性进行排序

③将此实现Comparator接口的实现类的对象作为形参传递给TreeSet的构造器中

④向TreeSet中添加元素即可。若不实现此接口，会报运行时异常

要求重写的compareTo（）或者compare（）方法与equals（）和hashCode（）方法保持一致

Map接口：存储“键-值”对的数据 ---相当于高中的“函数 $y = f(x)$ ” (x_1,y_1) (x_2,y_2)

>key是不可重复的，使用Set存放。value可以重复的，使用Collection来存放的。一个key-value对构成一个entry(Map.Entry)，entry使用Set来存放。

>添加、修改 put(Object key,Object value) 删除remove(Object key) 获取get(Object key) size() / keySet() values() entrySet()

*

|-----HashMap：主要的实现类，可以添加null键，null值

|-----LinkedHashMap：是HashMap的子类，可以按照添加进Map的顺序实现遍历

|-----TreeMap：需要按照key所在类的指定属性进行排序。要求key是同一个类的对象。对key考虑使用自然排序 或 定制排序

|-----Hashtable：是一个古老的实现类，线程安全的，不可以添加null键，null值不建议使用。

|-----子类：Properties：常用来处理属性文件

Iterator接口：用来遍历集合Collection元素

Collections工具类：操作Collection及Map的工具类，大部分为static的方法。

附：Properties的使用

Properties pros = new Properties();

pros.load(new FileInputStream(new File("jdbc.properties")));

String user = pros.getProperty("user");

System.out.println(user);

String password = pros.getProperty("password");

System.out.println(password);