**中软国际考题-笔试**

1. 单选题

1、以下哪个是枚举接口？（C）

A. Runnable B. Throwable C. Enumeration D. List

2、以下代码的执行结果是？（B）

public class Example {

public static void main(String[] args) {

String s1 = "abc";

String s2 = "def";

Stack<String> stack = new Stack<String>();

stack.push(s1);

stack.push(s2);

try {

String s3 = stack.pop() + stack.peek();

System.out.println(s3);

} catch (Exception e) {

// TODO: handle exception

}

}

}

A. abcdef B. defabc C. abcabc D. defdef

3、 单列集合的顶层接口是（B）

A、 java.util.Map B、 java.util.Collection C、 java.util.List D、 java.util.Set

4、 下列代码运行的结果是（A）

ArrayList<String> al=new ArrayList<String>();

al.add(true);

al.add(123);

al.add(“abc”);

System.out.println(al);

A、 编译失败 B、 [true,123] C、 [true,123,abc] D、 [abc]

5、关于迭代器说法错误的是()

A、迭代器是取出集合元素的方式 B、迭代器的hasNext()方法返回值是布尔类型

C、List集合有特有迭代器 D、next()方法将返回集合中的上一个元素.

6、.实现下列哪个接口，可以启用比较功能( )

A、Runnable接口 B、Iterator接口 C、Serializable接口 D、Comparator接口

7、集合类的上级接口（B）

A.Collections B.Collection C.List D.Iterator

8、ArrayList类的底层数据结构是( A )

A、数组结构 B、链表结构 C、哈希表结构 D、红黑树结构

9、 以下关于集合的说法错误的是（D）。

A、ArrayList采用了和数组相同的存储方式，在内存中分配连续的空间

B、LinkedList的优点在于插入、删除元素时效率比较高

C、ArrayList的优点在于遍历元素和随机访问元素的效率比较高

D、ArrayList和LinkedList都不允许存储重复数据

10、阅读如下Java代码，出现错误的行号是（ C ）。

import java.util.\*;

public class Test {

public static void main(String[] args) {

List <String> list =new LinkedList<String>(); //1

list.add("b"); //2

list.addFirst("a"); //3

for(String a:list) //4

System.out.println(a);

}

}

1. 1 B.2 C.3 D.4

11、下列说法中关于Collection 和Collections 的区别，不正确的是？(C)

A、Collection 是java.util 下的接口

B、Collection 是各种集合的父接口，继承于它的接口主要有Set 和List；

C、Collections 是个java.lang 下的类，

D、Collections 是针对集合的帮助类，提供一系列静态方法实现对各种集合的 搜索、排序、线程安全化等操作。

12、下列哪个属于集合接口？（D）

A.Tree B.Stack C.Array D.Map

1. 多选题
2. 下列哪些项是泛型的优点？（A、C）
3. 不用向下强制类型转换 B. 代码容易编写 C. 类型安全
4. 运行速度快 E. 以上都不是

2、Java中的集合类包括ArrayList、LinkedList、HashMap等类，下列关于集合类描述正确的是（ABD）

A、 ArrayList和LinkedList均实现了List接口B、 ArrayList的查询速度比LinkedList快

C、 添加和删除元素时，ArrayList的表现更佳

D、 HashMap实现Map接口，它允许任何类型的键和值对象，并允许讲null用作键或值

3、 下面的代码用于输出字符数组ch中每个字符出现的次数

public static void main(String[] args){

char[] ch={‘a’,’b’,’a’,’b’,’c’,’b’};

HashMap map=new HashMap();

for(int i=0;i<ch.length;i++){

//位置1

}

System.out.println(map);

}

应填入位置1的代码是（不允许使用工具运行程序）(CD)

A、 if(map.contains(ch[i])){

map.put(ch[i],map.get(ch[i])+1);

}

else{

map.put(ch[i],1);

}

B、 if(map.contains(ch[i])){

map.put(ch[i],(Integer)map.get(ch[i])+1);

}

else{

map.put(ch[i],1);

}

C、 if(map.containsKey(ch[i])){

map.put(ch[i],(int)map.get(ch[i])+1);

}

else{

map.put(ch[i],1);

}

D、 if(map.containsKey(ch[i])){

map.put(ch[i],(Integer)map.get(ch[i])+1);

}

else{

map.put(ch[i],1);

}

4、下面关于HashMap和Hashtable的区别，说法正确的是( )

A.Hashtable线程不安全，效率高

B.HashMap线程安全，效率低

C.HashMap允许null键和值

D.Hashtable不允许null键和值

5、Java中的集合类包括ArrayList、LinkedList、HashMap等类，下列关于集合类描述正确的是（ ）

A.ArrayList和LinkedList均实现了List接口

B.ArrayList的查询速度比LinkedList快

C.添加和删除元素时，ArrayList的表现更佳

D.HashMap实现Map接口，它允许任何类型的键和值对象，并允许将null用作键或值

6、下面哪些实现类属于List接口（A、B）

A. ArrayList B. LinkedList C. Lists

D. HashSet; E. TreeSet F. Collections

7、Collection继承自它的接口有（A、B）

A. List B. Set; C. HashMap D. TreeMap

8、 下列哪些方法是ArrayList和LinkedList集合中都定义的( ACD )

(A) add(Object o)(B) removeFirst()(C) remove(Object o)(D) add(int index,Object o)

9、关于集合下面哪些说法不正确？（A、B）

A、 java集合提供了一系列的接口，如List，HashSet

B、 Map接口是以键值对存储的，键和值都可以重复

C、 TreeSet存储的数据时有序的

D、 HashMap的key的集合类型是Set类型

10、 Iterator是什么,以下说法正确的是？（A、B、C、D）

A、Iterator接口提供遍历任何Collection的接口。

B、我们可以从一个Collection中使用迭代器方法来获取迭代器实例。

C、迭代器取代了Java集合框架中的Enumeration。

D、迭代器允许调用者在迭代过程中移除元素。

11、哪些集合是同步的？（B、C）

A.TreeSet B.Hashtable C. Vector D.LinkedList

1. 填空题
2. 集合类的两个根接口是\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_。(Collection、Map)
3. ArrayList（数组结构）的优点是\_\_\_\_，缺点是\_\_\_\_。（查询的速度快、添加删除的速度慢）
4. List集合的实现子类有哪些，列举三个\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_。（ArrayList、LinkedList、Vector）
5. Java集合的知识中，\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_类是线程安全的。(Vector、hashtable)
6. Collection框架中实现比较要实现什么接口\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、结束。(comparable、comparator)
7. List接口中的实现类分别是：\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_。(ArrayList、LinkedList、vector)
8. Java中集合框架List接口的特点 \_\_\_\_\_\_\_\_\_\_\_，Set接口的特点\_\_\_\_\_\_\_\_\_( 有序不唯一，无序唯一 )
9. 说出java集合中重要的接口和类：\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、结束。(list、set、map)
10. 集合接口\_\_\_\_\_\_\_\_\_支持通过字符串主键检索对象？(Map)
11. 判断题
12. ArrayList是线程安全的。（ F ）
13. Map集合中存储的是键值对，键不能重复，值可以重复。（ T ）
14. Iterator的remove方法结合next()方法使用比如集合中每隔一项删除一项，Iterator的remove()效率更低（ F ）
15. Hashtable不允许null键和值。（ T ）
16. hashtable的方法都提供了同步机制。（ T ）
17. Vector是线程安全的吗?（ F ）
18. Map接口继承Collection 接口。（ F ）
19. Collection和Collections是一样的，都是集合接口（ F ）
20. Iterator迭代器比for循环的效率高（ F）
21. Set支持元素排序。（ F ）
22. Map是Collection接口的子接口。（ F ）

五、简答题

1、List、Map、Set三个接口，存取元素时，各有什么特点?

答案：

List 以特定次序来持有元素，可有重复元素。

Set 无法拥有重复元素,内部排序。

Map 保存key-value值，value可多值。

2、HashMap的实现原理

答案：通过put和get存储和获取对象，存储对象时，我们将K/V传给put方法时，它调用hashcode计算hash从而得到bucket位置，进一步存储，HashMap会根据当前bucket的占用情况自动调整容量。获取对象时，我们将K传递给get,它调用hashcode计算hash从而得到bucket位置，并进一步调用equals()方法确认键值对。

3、List 和 Map 区别？（数据结构，存储特点）

答案：这个要从两个方面来回答，一方面是List和Map的数据结构，另一方面是存储数据的特点。在数据结构方面，List存储的是单列数据的集合，而Map存储的是key、value类型的数据集合。在数据存储方面，List存储的数据是有序且可以重复的，而Map中存储的数据是无序且key值不能重复（value值可以重复）

* 1. ArrayList和Vector的区别（是否有序、是否重复、数据结构、底层实现）

ArrayList和Vector都实现了List接口，他们都是有序集合，并且存放的元素是允许重复的。它们的底层都是通过数组来实现的，因此列表这种数据结构检索数据速度快，但增删改速度慢。

而ArrayList和Vector的区别主要在两个方面：

第一，线程安全。Vector是线程安全的，而ArrayList是线程不安全的。因此在如果集合数据只有单线程访问，那么使用ArrayList可以提高效率。而如果有多线程访问你的集合数据，那么就必须要用Vector，因为要保证数据安全。

第二，数据增长。ArrayList和Vector都有一个初始的容量大小，当存储进它们里面的元素超过了容量时，就需要增加它们的存储容量。ArrayList每次增长原来的0.5倍，而Vector增长原来的一倍。ArrayList和Vector都可以设置初始空间的大小，Vector还可以设置增长的空间大小，而ArrayList没有提供设置增长空间的方法。

5、集合框架中的泛型有什么优点？（5分）

答案：泛型允许我们为集合提供一个可以容纳的对象类型，因此，如果你添加其它类型的任何元素，它会在编译时报错。这避免了在运行时出现ClassCastException，因为你将会在编译时得到报错信息。泛型也使得代码整洁，我们不需要使用显式转换和instanceOf操作符。它也给运行时带来好处，因为不会产生类型检查的字节码指令。

6、List,Set,Collection,Collections分别是什么，各有什么特点？

答案：List和Set都是接口，他们都继承于接口Collection,List是一个有序的可重复的集合，而Set的无序的不可重复的集合。Collection是集合的顶层接口，Collections是一个封装了众多关于集合操作的静态方法的工具类,因为构造方法是私有的，所以不能实例化。

List接口实现类有ArrayList,LinkedList,Vector。ArrayList和Vector是基于数组实现的,所以查询的时候速度快，而在进行增加和删除的时候速度较慢LinkedList是基于链式存储结构，所以在进行查询的时候速度较慢但在进行增加和删除的时候速度较快。又因为Vector是线程安全的，所以他和ArrayList相比而言，查询效率要低。

7、ArrayList和LinkedList的区别。(分别写出2条即可)（5分）

答案：1.ArrayList是实现了动态数组的数据结构，LinkedList基于链表的数据结构。

　　 2.对于随机访问get和set，很明显ArrayList的性能要优于LinkedList。

　　 3.对于增加和删除操作add和remove，LinkedList占优势。

　　 当操作是在一列数据的后面添加数据而不是在前面或中间，并且需要随机地访问其中的元素时，使用ArrayList性能比较好；当操作是在一列数据的前面或中间添加或删除数据，并且按照顺序访问其中的元素时，就应该使用LinkedList了。

8、List 和 Map 区别？（数据结构，存储特点）

答案：这个要从两个方面来回答，一方面是List和Map的数据结构，另一方面是存储数据的特点。在数据结构方面，List存储的是单列数据的集合，而Map存储的是key、value类型的数据集合。在数据存储方面，List存储的数据是有序且可以重复的，而Map中存储的数据是无序且key值不能重复（value值可以重复）。

9、HashMap和Hashtable的区别

答案：HashMap和Hashtable都实现了Map接口，并且都是key-value的数据结构。它们的不同点主要在三个方面：第一，Hashtable是Java1.1的一个类，它基于陈旧的Dictionary类。而HashMap是Java1.2引进的Map接口的一个实现。

第二，Hashtable是线程安全的，也就是说是线程同步的，而HashMap是线程不安全的。也就是说在单线程环境下应该用HashMap，这样效率更高。

第三，HashMap允许将null值作为key或value，但Hashtable不允许（会抛出NullPointerException）。

10、Collection 和Collections 的区别？

答：Collection 是java.util 下的接口，它是各种集合的父接口，继承于它的接口主要有Set 和List；Collections 是个java.util 下的类，是针对集合的帮助类，提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

11、Arraylist 与Vector 区别？

答：就ArrayList 与Vector 主要从二方面来说：

1）同步性：Vector 是线程安全的（同步），而ArrayList 是线程序不安全的；

2）数据增长：当需要增长时,Vector 默认增长一倍，而ArrayList 却是一半。

12、List、Map、Set 三个接口，存取元素时，各有什么特点？

答：List 以特定次序来持有元素，可有重复元素。Set 无法拥有重复元素,内部排序。Map 保存key-value 值，value 可多值。

13、1、 集合有哪四种主要的接口，分别什么作用？

答案：1、Collection：代表一组对象，每一个对象都是它的子元素。

2、Set：不包含重复元素的Collection。

3、List：有顺序的collection，并且可以包含重复元素。

4、Map：可以把键(key)映射到值(value)的对象，键不能重复。

14、 说出ArrayList、Vector、LinkedList的存储性能和特性。

答案：ArrayList 采用的是数组形式来保存对象的，这种方式将对象放在连续的位置中，所以最大的缺点就是插入删除时非常麻烦

LinkedList 采用的将对象存放在独立的空间中，而且在每个空间中还保存下一个链接的索引 但是缺点就是查找非常麻烦 要丛第一个索引开始

ArrayList和Vector都是用数组方式存储数据,此数组元素数要大于实际的存储空间以便进行元素增加和插入操作,他们都允许直接用序号索引元素,但是插入数据元素涉及到元素移动等内存操作,所以索引数据快而插入数据慢.

Vector使用了sychronized方法(线程安全),所以在性能上比ArrayList要差些.

LinkedList使用双向链表方式存储数据,按序号索引数据需要前向或后向遍历数据，所以索引数据慢,是插入数据时只需要记录前后项即可,所以插入的速度快.

15、 一组从小到大的数据存在ArrayList中，写个函数通过折半查找迅速找到对应的索引值。

答案：package Algorithm\_analysis;

public class Bisearch {

static int[] array={-3,-2,0,4,5,7,12,64};

public static void main(String args[]){

int left=0;

int right=array.length;

int center=0;

int k=7;

while(left<=right){

center=(right+left)/2;

if ((array[center]-k)==0){

System.out.print(center);

break;

} else{

if((array[center]-k)>0){

right=center;

}

else{

left=center;

}

}

}

}

}

16、 实现单向链表的反转，如：a->b->c->d，反转后变为 d->c->b->a

答案：public Node reverse(Node current){

if (current == null || current.next == null)

return current;

Node nextNode = current.next;

current.next = null;

Node reverseRest = reverse(nextNode);

nextNode.next = current;

}

17、 请说一下你熟悉的集合，并说一下它们是如何取值的。

答案：ArrayList是一个数组实现的列表，由于数据是存入数组中的。

List<String> exampleList=new ArrayList<String>();

　第一种：List集合遍历的最基础的方式：for循环，指定下标长度，根据List集合的size()长度，for循环遍历；

//i的操作，小于或者小于等于集合的长度，根据自己的需求，可以

for(int i=0;i<exampleList.size();i++){

Example example = exampleList.get(i);//获取每一个Example对象

String name = example.getName();

System.out.print("第"+i+"个=？"+name);

}

　第二种：非常简单的写法：直接根据List集合的长度自动遍历，但是不能操作第几个；

for(Example example : exampleList){

String name = example.getName();//直接操作Example对象

System.out.print(("Name:"+name);

}

第三种：利用迭代器Iterator遍历：也是直接根据List集合的自动遍历，知道遍历完整个List；

for(Iterator iterators = exampleList.iterator();iterators.hasNext();){

Example example = (Example) iterators.next();//获取当前遍历的元素，指定为Example对象

String name = example.getName();

System.out.print("Name:"+name);

}

18、 说明Map、HashMap、LinkMap、TreeMap的关系和区别。

HashMap,LinkedHashMap,TreeMap都属于Map

Map 主要用于存储键(key)值(value)对，根据键得到值，因此键不允许键重复,但允许值重复。

HashMap 是一个最常用的Map,它根据键的HashCode 值存储数据,根据键可以直接获取它的值，具有很快的访问速度。HashMap最多只允许一条记录的键为Null;允许多条记录的值为Null;HashMap不支持线程的同步，即任一时刻可以有多个线程同时写HashMap;可能会导致数据的不一致。如果需要同步，可以用Collections的synchronizedMap方法使HashMap具有同步的能力。

LinkedHashMap LinkedHashMap也是一个HashMap,但是内部维持了一个双向链表,可以保持顺序

TreeMap 不仅可以保持顺序，而且可以用于排序

19、 HashMap与HashTable有什么区别？什么时候使用Hashtable，什么时候使用 HashMap？

1.HashTable的方法是同步的，HashMap未经同步，所以在多线程场合要手动同步HashMap这个区别就像Vector和ArrayList一样。

2.HashTable不允许null值(key和value都不可以),HashMap允许null值(key和value都可以)。

3.HashTable有一个contains(Object value)，功能和containsValue(Object value)功能一样。

4.HashTable使用Enumeration，HashMap使用Iterator。

以上只是表面的不同，它们的实现也有很大的不同。

5.HashTable中hash数组默认大小是11，增加的方式是 old\*2+1。HashMap中hash数组的默认大小是16，而且一定是2的指数。

6.哈希值的使用不同，HashTable直接使用对象的hashCode，代码是这样的：

int hash = key.hashCode();

int index = (hash & 0x7FFFFFFF) % tab.length;

而HashMap重新计算hash值，而且用与代替求模：

int hash = hash(k);

int i = indexFor(hash, table.length);

20、 Java 中 Set 与 List 有什么不同?

• Set是一个不允许重复元素存在的集合

• Set没有索引

• Set仅仅允许一个null值

• Set有类：HashSet、LinkedHashMap、TreeSet

• List有索引

• List允许N个null值

• List可以按插入顺序显示

• List有类：Vector、ArrayList、LinkedList

21、 如何保证一个集合线程安全？

• Vector, Hashtable, Properties 和 Stack 都是同步的类，所以它们都线程安全的，可以被使用在多线程环境中

• 使用Collections.synchronizedList(list)) 方法，可以保证list类是线程安全的

• 使用java.util.Collections.synchronizedSet()方法可以保证set类是线程安全的。

22、 是否可以往 TreeSet 或者 HashSet 中添加 null 元素？

• 可以往 hashset 中添加一个 null

• TreeSet 也允许一个 null值

23、ArrayList、LinkedList的区别？

答案：ArrayList底层是采用数组实现的，LinkedList底层是由双向循环链表实现的，ArrayList做遍历效率高，LinkedList插入删除效率高

24. HashSet和HashMap的区别？

答案：HashSet是set接口的实现类，数据方式是唯一无序的，HashMap是Map接口的实现类，key唯一，value不唯一

25、说出ArrayList,Vector, LinkedList 的存储性能和特性？

答：ArrayList 和Vector 都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，它们都允许直接按序号索引元素，但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，Vector 由于使用了synchronized 方法（线程安全），通常性能上较ArrayList 差，而LinkedList 使用双向链表实现存储，按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

26、.List、Map、Set 三个接口，存取元素时，各有什么特点？

答：List 以特定次序来持有元素，可有重复元素。Set 无法拥有重复元素,内部排序。Map 保存key-value 值，value 可多值。

27、arraylist和linkedlist的区别？

1.ArrayList是实现了基于动态数组的数据结构，LinkedList基于链表的数 据结构。

2.对于随机访问get和set，ArrayList觉得优于LinkedList，因为LinkedList要移动指针。

3.对于新增和删除操作add和remove，LinedList比较占优势，因为ArrayList要移动数据。 这一点要看实际情况的。若只对单条数据插入或删除，ArrayList的速度反而优于LinkedList。但若是批量随机的插入删除数据，LinkedList的速度大大优于ArrayList. 因为ArrayList每插入一条数据，要移动插入点及之后的所有数据。

28、Hashtable和HashMap的区别

答案：Hashtable和HashMap类有三个重要的不同之处。第一个不同主要是历史原因。Hashtable是基于陈旧的Dictionary类的，HashMap是Java 1.2引进的Map接口的一个实现。

　　也许最重要的不同是Hashtable的方法是同步的，而HashMap的方法不是。这就意味着，虽然你可以不用采取任何特殊的行为就可以在一个多线程的应用程序中用一个Hashtable，但你必须同样地为一个HashMap提供外同步。一个方便的方法就是利用Collections类的静态的synchronizedMap()方法，它创建一个线程安全的Map对象，并把它作为一个封装的对象来返回。这个对象的方法可以让你同步访问潜在的HashMap。这么做的结果就是当你不需要同步时，你不能切断Hashtable中的同步（比如在一个单线程的应用程序中），而且同步增加了很多处理费用。

　　第三点不同是，只有HashMap可以让你将空值作为一个表的条目的key或value。HashMap中只有一条记录可以是一个空的key，但任意数量的条目可以是空的value。这就是说，如果在表中没有发现搜索键，或者如果发现了搜索键，但它是一个空的值，那么get()将返回null。如果有必要，用containKey()方法来区别这两种情况。

29、Vector、ArrayList、LinkedList的区别

答案：ArrayList 和Vector是采用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，都允许直接序号索引元素，但是插入数据要设计到数组元素移动等内存操作，所以索引数据快插入数据慢，Vector由于使用了synchronized方法（线程安全）所以性能上比ArrayList要差，LinkedList使用双向链表实现存储，按序号索引数据需要进行向前或向后遍历，但是插入数据时只需要记录本项的前后项即可，所以插入数度较快！

30、List、Map、Set三个接口，存取元素时，各有什么特点？

答案：List 以特定次序来持有元素，可有重复元素。Set 无法拥有重复元素,内部排序。Map 保存key-value值，value可多值。

31、Collection和Collections的区别，Array和Arrays的区别？

答案：java.util.Collection 是一个集合接口。它提供了对集合对象进行基本操作的通用接口方法。Collection接口在Java 类库中有很多具体的实现。Collection接口的意义是为各种具体的集合提供了最大化的统一操作方式。Collections是针对集合类的一个帮助类，他提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

Array类提供了动态创建和访问 Java 数组的方法。

Arrays包含用来操作数组（比如排序和搜索）的各种方法。此类还包含一个允许将数组作为列表来查看的静态工厂。

六、程序实现题

1、集合类的排序：定义User实体类，包含姓名与生日（Date）属性，创建若干个User对象并加入到集合类中，在集合类中根据User对象的生日进行排序，按照年龄从小到大排序集合类。

（10分）

|  |
| --- |
| package test;  import java.util.ArrayList;  import java.util.Collections;  import java.util.Comparator;  import java.util.Date;  import java.util.List;  class User {  private String name;  private Date birthday;  public User(String name, Date birthday) {  super();  this.name = name;  this.birthday = birthday;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public Date getBirthday() {  return birthday;  }  public void setBirthday(Date birthday) {  this.birthday = birthday;  }    }  public class TestSort {  public static void main(String[] args) {  User u1 = new User("zhangsan", new Date(90, 1, 2));  User u2 = new User("lisi", new Date(89, 1, 2));  User u3 = new User("wangwu", new Date(91, 1, 2));  List<User> users = new ArrayList<User>();  users.add(u1);  users.add(u2);  users.add(u3);  Collections.sort(users, new Comparator<User>() {  public int compare(User u1, User u2) {  return (int) (u1.getBirthday().getTime() - u2.getBirthday().getTime());  }  });  System.out.println(users.get(0).getName());  }  } |

2、遍历一个集合，删除所有与s相同的元素

（10分）

|  |
| --- |
| /\*\*  删除集合中的指定元素  @parm list 集合  @parm s 指定的元素  \*\*/  public static void removes(List<String> list,String s){  Iterator iterator=list.iterator();  while(iterator.hasNext()){  String str=(String) iterator.next();  if(str.equals(s)){  iterator.remove();  }  }  } |

3、已知数组存放一批QQ号码，QQ号码最长为11位，最短为5位String[] strs = {"12345","67891","12347809933","98765432102","67891","12347809933"}。

将该数组里面的所有qq号都存放在LinkedList中，将list中重复元素删除，将list中所有元素分别用迭代器和增强for循环打印出来。

（10分）

|  |
| --- |
| import java.util.Iterator;  import java.util.LinkedHashSet;  import java.util.LinkedList;  /\*  \* 已知数组存放一批QQ号码，QQ号码最长为11位，最短为5位String[] strs = {"12345","67891","12347809933","98765432102","67891","12347809933"}。  将该数组里面的所有qq号都存放在LinkedList中，将list中重复元素删除，将list中所有元素分别用迭代器和增强for循环打印出来。  \* \*/  public class Topic3 {  public static void main(String[] args) {  String[] strs = {"12345","67891","12347809933","98765432102","67891","12347809933"};  LinkedHashSet<String> set = method01(strs);  //方式1  // System.out.println(set);  //方式2  // System.out.println(method02(strs));  //迭代器打印  Iterator<String> it = method01(strs).iterator();  while (it.hasNext()){  String value = it.next();  System.out.print(value+" ");  }  System.out.println();  //增强for打印  for (String s : method02(strs)) {  System.out.print(s+" ");  }  }  //方式2  private static LinkedList<String> method02(String[] strs) {  LinkedList<String> list = new LinkedList<>();  for (int i = 0; i < strs.length; i++) {  list.add(strs[i]);  }  int flag =1;// 0为找不到,1为找到  //如果此元素在,除了本次位置的其他元素内找到,则删除该元素.  for (int i = 0; i < list.size(); i++) {  //判断这个元素之后的元素 是否与这个元素相等. 是则删除,不是则继续执行.  for(int j=i+1;j<list.size();j++){  if (list.get(i).equals(list.get(j))){  list.remove(j);  }  }  }  return list;  }  //方式1  private static LinkedHashSet<String> method01(String[] strs) {  LinkedHashSet<String> set = new LinkedHashSet<>();  for (int i = 0; i < strs.length; i++) {  set.add(strs[i]);  }  return set;  }  } |

4、如何判断两个集合是否有交集，并打印出他们的交集

（10分）

|  |
| --- |
| public static void main(String[] args) {  HashSet<String> hs=new HashSet<String>();  hs.add("George");  hs.add("Jim");  hs.add("Blake");  hs.add("Kevin");  hs.add("Mecheal");  hs.add("John");  HashSet<String> hs2=new HashSet<String>();  hs2.add("George");  hs2.add("Kate");  hs2.add("Kevin");  hs2.add("Mecheal");  hs2.add("Ryan");  hs.retainAll(hs2);//retainAll（）的方法返回时boolean，但不能作为又没有交集的判断，判断有没有交集要通过hs的size（）。  if (hs.size()==0){  System.out.println("没有交集");  }else{  System.out.println("有交集");  }  for (String s : hs) {  System.out.println(s);  }  } |

5、遍历一个集合，删除所有与s相同的元素

注意：不要使用for循环遍历删除，会出现删除不干净的情况

|  |
| --- |
| /\*\*  删除集合中指定的元素  @param list 集合  @param s 指定的元素  \*/  public static void removes(List<String> list, String s) {  Iterator iterator = list.iterator();  while (iterator.hasNext()) {  String str = (String) iterator.next();  if (str.equals(s)) {  iterator.remove();  }  }  } |

6、遍历一个集合，删除所有与s相同的元素（注意：不要使用for循环遍历删除，会出现删除不干净的情况）

（10分）

|  |
| --- |
| /\*\*  删除集合中指定的元素  @param list 集合  @param s 指定的元素  \*/  public static void removes(List<String> list, String s) {  Iterator iterator = list.iterator();  while (iterator.hasNext()) {  String str = (String) iterator.next();  if (str.equals(s)) {  iterator.remove();  }  }  } |

7、如何判断两个集合是否有交集，并打印出他们的交集

|  |
| --- |
| public static void main(String[] args) {  HashSet<String> hs=new HashSet<String>();  hs.add("George");  hs.add("Jim");  hs.add("Blake");  hs.add("Kevin");  hs.add("Mecheal");  hs.add("John");  HashSet<String> hs2=new HashSet<String>();  hs2.add("George");  hs2.add("Kate");  hs2.add("Kevin");  hs2.add("Mecheal");  hs2.add("Ryan");  hs.retainAll(hs2);//retainAll（）的方法返回时boolean，但不能作为又没有交集的判断，判断有没有交集要通过hs的size（）。  if (hs.size()==0){  System.out.println("没有交集");  }else{  System.out.println("有交集");  }  for (String s : hs) {  System.out.println(s);  }  } |

8、List集合的遍历方法

|  |
| --- |
| public static void main(String[] args) {  List<String> list=new ArrayList<>();  list.add("George");  list.add("Jim");  list.add("Blake");  list.add("Kevin");  list.add("Mecheal");  list.add("John");  //for循环  for (int i = 0; i < list.size(); i++) {  String s = list.get(i);  System.out.println(s);  }  //foreach  for (String s : list) {  System.out.println(s);  }  //迭代器遍历  Iterator<String> iterator = list.iterator();  while (iterator.hasNext()) {  System.out.println(iterator.next());  }  } |

9、Set集合的遍历方法

|  |
| --- |
| public static void main(String[] args) {  Set<String> list=new HashSet<>();  list.add("George");  list.add("Jim");  list.add("Blake");  list.add("Kevin");  list.add("Mecheal");  list.add("John");  //foreach  for (String s : list) {  System.out.println(s);  }  //迭代器遍历  Iterator<String> iterator = list.iterator();  while (iterator.hasNext()) {  System.out.println(iterator.next());  } |

10、Map 集合的遍历方法

|  |
| --- |
| public static void main(String[] args) {  Map<String, String> map = new HashMap<>();  map.put("name", "Tom");  map.put("age", "20");  map.put("address", "beijing");  for (String key : map.keySet()) {  String value = map.get(key);  System.out.println(key+"==="+value);  }  Iterator<Map.Entry<String, String>> iterator = map.entrySet().iterator();  while (iterator.hasNext()) {  Map.Entry<String, String> entry = iterator.next();  String key = entry.getKey();  String value = entry.getValue();  System.out.println(key+"==="+value);  }  for (Map.Entry<String,String> entry: map.entrySet()) {  String key = entry.getKey();  String value = entry.getValue();  System.out.println(key+"==="+value);  }  for (String value:map.values()) {  System.out.println(value);  }  } |

11、用java实现双向链表结构及实现插入数据结点的方法。（10分）

|  |
| --- |
| //添加到链表表头  public boolean addFirst(Object o)  {  addAfter(new Node(o), head);  return true;  }  //将元素添加到链表表尾  public boolean addLast(Object o)  {  addBefore(new Node(o), head);  return true;  }  //在某元素之前添加元素  private void addBefore(Node newNode, Node node)  {  newNode.prev = node.prev;  newNode.next = node;  newNode.next.prev = newNode;  newNode.prev.next = newNode;  size++;  }  //在某元素之后添加元素  private void addAfter(Node newNode, Node node)  {  newNode.prev = node;  newNode.next = node.next;  newNode.next.prev = newNode;  newNode.prev.next = newNode;  size++;  } |

12、使用增强for遍历如下List

List list = new ArrayList();（10分）

List.add(“1”);

List.add(“2”);

|  |
| --- |
| public class Test {  public static void main(String[] args) {  List list = new ArrayList();（10分）  List.add(“1”);  List.add(“2”);  for(Object o:list){  String s = (String) o;  System.out.println(s);  }  }  } |

13、使用迭代器遍历如下Map

HashMap<Integer,String> map=new HashMap<Integer,String>();（10分）

map.put(1,”test1”);

map.put(2,”test2”);

|  |
| --- |
| public static void main(String[] args) {  HashMap<Integer,String>map=new HashMap<Integer,String>();  map.put(1, "test1");  map.put(2, "test1");    Set<Integer> set = map.keySet();  Iterator<Integer> it = set.iterator();  while(it.hasNext()){  Integer key = it.next();  String str = map.get(key);  System.out.println(str);  }  } |

14、用程序给出随便大小的10 个数，序号为1-10，按从小到大顺序输出，并输出相应的序号。

（10分）

|  |
| --- |
| **public** **class** RandomSort {  **public** **static** **void** printRandomBySort() {  Random random = **new** Random();  // 创建随机数生成器List list = new ArrayList();  // 生成10 个随机数，并放在集合list 中  **for** (**int** i = 0; i < 10; i++) {  list.add(random.nextInt(1000));  }  Collections.*sort*(list); // 对集合中的元素进行排序Iterator it = list.iterator();int count = 0;while (it.hasNext()) { // 顺序输出排序后集合中的元素System.out.println(++count + ": " + it.next());  }  } |

15、建立国家英文简称和中文全名间的键值映射，并通过key对value进行操作，应该如何实现数据的存储和操作呢？根据键查找对应的值

|  |
| --- |
| public class Country {  private String mo;  private String name;  public Country(String mo, String name) {  super();  this.mo = mo;  this.name = name;  }  @Override  public String toString() {  // TODO Auto-generated method stub  return "英文缩写："+mo+"中文全名："+name;  }  public String getMo() {  return mo;  }  public String getName() {  return name;  }  }  public class CountryTest {  public static void main(String[] args) {  // TODO Auto-generated method stub  HashMap<String,Country> hm = new HashMap<String,Country>();  Country coun1 = new Country("US","美利坚");  Country coun2 = new Country("RU","俄罗斯");  Country coun3 = new Country("CN","中华人民共和国");  hm.put(coun1.getMo(), coun1);  hm.put(coun2.getMo(), coun2);  hm.put(coun3.getMo(), coun3);  System.out.println(hm);  System.out.println("========================");  Set<String> keys = hm.keySet();  for(String key:keys){  //System.out.println(key);  //System.out.println(hm.get(key));  System.out.println(hm.get(key).getName());  }  System.out.println("========================");  String mo = "CN";  for(String key:keys){  if(hm.get(key).getMo().equals(mo)){  System.out.println(hm.get(key).getName());  }  }  }  } |