枚举的用法：

/\*

Enumeration<String>days;

Vector<String>daynames=new Vector<String>();

daynames.add("sunday");

daynames.add("Monday");

daynames.add("Tuesday");

daynames.add("wednesday");

daynames.add("Thursday");

daynames.add("Friday");

daynames.add("Saturday");

days=daynames.elements();

while(days.hasMoreElements())

System.out.println(days.nextElement());

\*/

向量的用法：

Vector v=**new** Vector(3,2);

System.*out*.println(v.size());

System.*out*.println(v.capacity());

v.addElement(**new** Integer(1));

v.addElement(**new** Integer(2));

v.addElement(**new** Integer(3));

v.addElement(**new** Integer(4));

v.addElement(**new** Double(5.44));

System.*out*.println(v.firstElement());

System.*out*.println(v.lastElement());

Enumeration E=v.elements();

**while**(E.hasMoreElements())

System.*out*.println(E.nextElement());

对数组进行随机排序

方法1：

List<String>list=**new** ArrayList<String>();

list.add("I");

list.add("like");

list.add("lulu");

Collections.*shuffle*(list,**new** Random());//利用collections 的shuffle方法对list随机排序

System.*out*.println(list);

方法2:

. List<String> list=Arrays.*asList*(args);

Collections.*shuffle*(list);

System.*out*.println(list);

System.*out*.println(System.*getProperty*("user.dir"));//获取当前工程路径

String[] suit=**new** String[]{"梅花","方片","黑桃","红心"};

List<String> deck=**new** ArrayList<String>();

String[] rank=**new** String[]{"1","2","3","4","5","6","7","8","9","10","J","Q","K"};

**for**(**int** i=0;i<suit.length;i++)

**for**(**int** j=0;j<rank.length;j++)

deck.add(suit[i]+rank[j]);

Collections.*shuffle*(deck);

ListIterator<String> it=deck.listIterator();

// while(it.hasNext())

// System.out.println(it.next());

System.*out*.println(deck.subList(0, 13));

Java实现倒计时

**public** **class** queue\_test {

**public** **static** **void** main(String[] args) **throws** InterruptedException

{

**int** time=60;

Queue<Integer> queue=**new** LinkedList<Integer>();

**for**(**int** i=time;i>0;i--)

queue.add(i);

**while**(!queue.isEmpty()){

System.*out*.println(queue.remove()); //删除头结点并返回其值

Thread.*sleep*(1000); //线程暂停执行一秒钟

}

}

}

用优先队列对数组进行排序

**public** **static** **void** main(String[] args) **throws** InterruptedException

{

List<Integer> list=**new** ArrayList<Integer>();

List<Integer> list1=**new** ArrayList<Integer>();

list.add(11);

list.add(72);

list.add(7);

list.add(36);

list1=*heapsort*(list);

System.*out*.println(list1.subList(4, 8));

}

**static** <E> List<E> heapsort(Collection<E> c)

{

Queue<E> queue=**new** PriorityQueue<E>(c);//创建优先级队列对象，元素为c中的元素

List<E> result=**new** ArrayList<E>(c);

**while**(!queue.isEmpty())

result.add(queue.remove());

**return** result;

}

遍历Map的三种方法

Iterator key\_value=hashmap.entrySet().iterator();

Iterator keys=hashmap.keySet().iterator();

Iterator values=hashmap.values().iterator();

对数组遍历的三种操作

List<String> list=**new** ArrayList<String>();

list.add("I");

list.add("Miss");

list.add("you");

//用for循环遍历List

**for**(String str:list)

{

System.*out*.println(str);

}

//先转换为数组再进行遍历

String[] strArray=**new** String[list.size()];

list.toArray(strArray);

**for**(**int** i=0;i<strArray.length;i++)

System.*out*.println(strArray[i]);

//用迭代器进行遍历

Iterator it=list.iterator();

**while**(it.hasNext())

System.*out*.println(it.next());

对Map的四种遍历方式

Map<String,String> map=**new** HashMap<String,String>();

map.put("露露", "贵阳");

map.put("汤月", "山东");

map.put("老妹", "贵阳");

//第一种，普遍使用，二次取值

**for**(String key:map.keySet())

System.*out*.println("name:"+key+" address: "+map.get(key));

//第二种，通过Map.EntrySet使用iterator来遍历

Iterator<Map.Entry<String,String>> it=map.entrySet().iterator();

**while**(it.hasNext())

{

Map.Entry<String, String> entry=it.next();

System.*out*.println("name: "+entry.getKey()+" address:"+entry.getValue());

}

//第三种，尤其是大容量时,通过Map.entrySet遍历key 和value

**for**(Map.Entry<String, String> entry:map.entrySet())

System.*out*.println("name:" +entry.getKey()+"address:"+entry.getValue());

//第四种，只能遍历value

**for**(String v:map.values())

{

System.*out*.println(v);

}