折半查找:折半插入排序，添加元素，顺序按照从小...

疯狂代码 http://CrazyCoder.cn/ ĵ:http:/CrazyCoder.cn/Arithmetic/Article24519.html

今天在公司处理了一下排序的问题,所以就把这个东西贴出来,分享一下,随便加深一下印象;

包括两个类:PageQueueObject.java;(处理多种情况分类有用)

BinarySortedLinkedQueue.java(折半插入排序，添加元素，顺序按照从小到大的顺序)

```
/**
* 处理多种情况分类
*
* @author lake 20071103
*  * @version 1.0
*/
public class PageQueueObject {
// 分页队列的顺序ID
public final int pagination;
// 分页url
public final UrlInfo url;
//该分页对应的种类
public final String varKind;
public PageQueueObject(UrlInfo url, int pagination, String varKind) {
this.pagination = pagination;
this.url = url;
this.varKind = varKind;
}
}
```

```
/**
* 本链表队列按照从小到小大 顺序入队，永远从头部移出元素的特殊有序队列
*
* @author lake 20071103
* * @version 1.0
*/
public class BinarySortedLinkedQueue {
// 队列链表
```

```
private LinkedList<PageQueueObject> link = null;
private static byte[] lock = new byte[0];
public BinarySortedLinkedQueue() {
this.link = new LinkedList<PageQueueObject>();
}
/**
* <p>
* Title: add
* </p>
* <p>
* Description:利用折半插入排序，添加元素，顺序按照PageQueueObject.pagination从小到大的顺序
* </p>
*
* @param key：对应的Domain
* @param value：最终查询的url值
*/
public void add(PageQueueObject pqo) {
if (pqo != null) {
synchronized (lock) {
if (link != null) {
boolean exception = false;
// 如果队列没有元素
if (link.size() < 1) {
link.add(pqo);
return;
}
int low = 0;
int hig = link.size() - 1;
int mid = 0;
// comparable不是自然排序,在这里只是同样名字的方法;
int tmp = comparable(pqo, link.get(hig));
if (tmp != Integer.MAX_VALUE) {
if (tmp > 0) {// 如果比最大的还要大，则直接在尾部添加
link.add(pqo);
return;
}
```

```
} else {
exception = true;
SysConstants.loger.error("出现异常:o=" + pqo + " ,link.get(" + mid + ")" + link.get(mid));
}
tmp = comparable(pqo, link.get(low));
if (tmp != Float.MAX_VALUE) {
// 如果比最小的还要小，则直接在首部添加
if (tmp < 0) {
link.addFirst(pqo);
return;
}
} else {
exception = true;
SysConstants.loger.error("出现异常:pqo=" + pqo + " ,link.get(" + mid + ")" + link.get(mid));
}
if (!exception) {
while (low <= hig) {
mid = (low + hig) / 2;
tmp = comparable(pqo, link.get(mid));
if (tmp != Float.MAX_VALUE) {
if (tmp < 0) {
hig = mid - 1;
} else {
low = mid + 1;
}
} else {
exception = true;
SysConstants.loger.error("出现异常:pqo=" + pqo+ " ,link.get(" + mid + ")" + link.get(mid));
break;
}
}
if (!exception) {
link.add(low, pqo);
}
}
}
```

```
}
}
}
/**
* <p>
* Title: poll
* </p>
* <p>
* Description:liyong利用折半插入排序，添加元素，顺序按照PageQueueObject.pagination从小到大的顺序
* </p>
*
* @param key：对应的Domain
* @param value：最终查询的url值
*/
public PageQueueObject poll() {
synchronized (lock) {
if (link != null) {
return link.poll();
} else {
return null;
}
}
}
public PageQueueObject get(int index) {
synchronized (lock) {
if (link != null) {
return link.get(index);
} else {
return null;
}
}
}
public int size() {
synchronized (lock) {
if (link != null) {
```

```
return link.size();
} else {
return 0;
}
}
}
public void clear() {
synchronized (lock) {
if (link != null) {
link.clear();
}
}
}
private int comparable(PageQueueObject o1, PageQueueObject o2) {
if (o1 != null && o2 != null) {
return o1.pagination - o2.pagination;
}
return Integer.MAX_VALUE;
}
public static void main(String[] agrs) {
}
}
```

说明:Integer.MAX_VALUE只作为判断用,一个不可能完成的判断;　　2008-11-27 2:11:21

　　疯狂代码 http://CrazyCoder.cn/