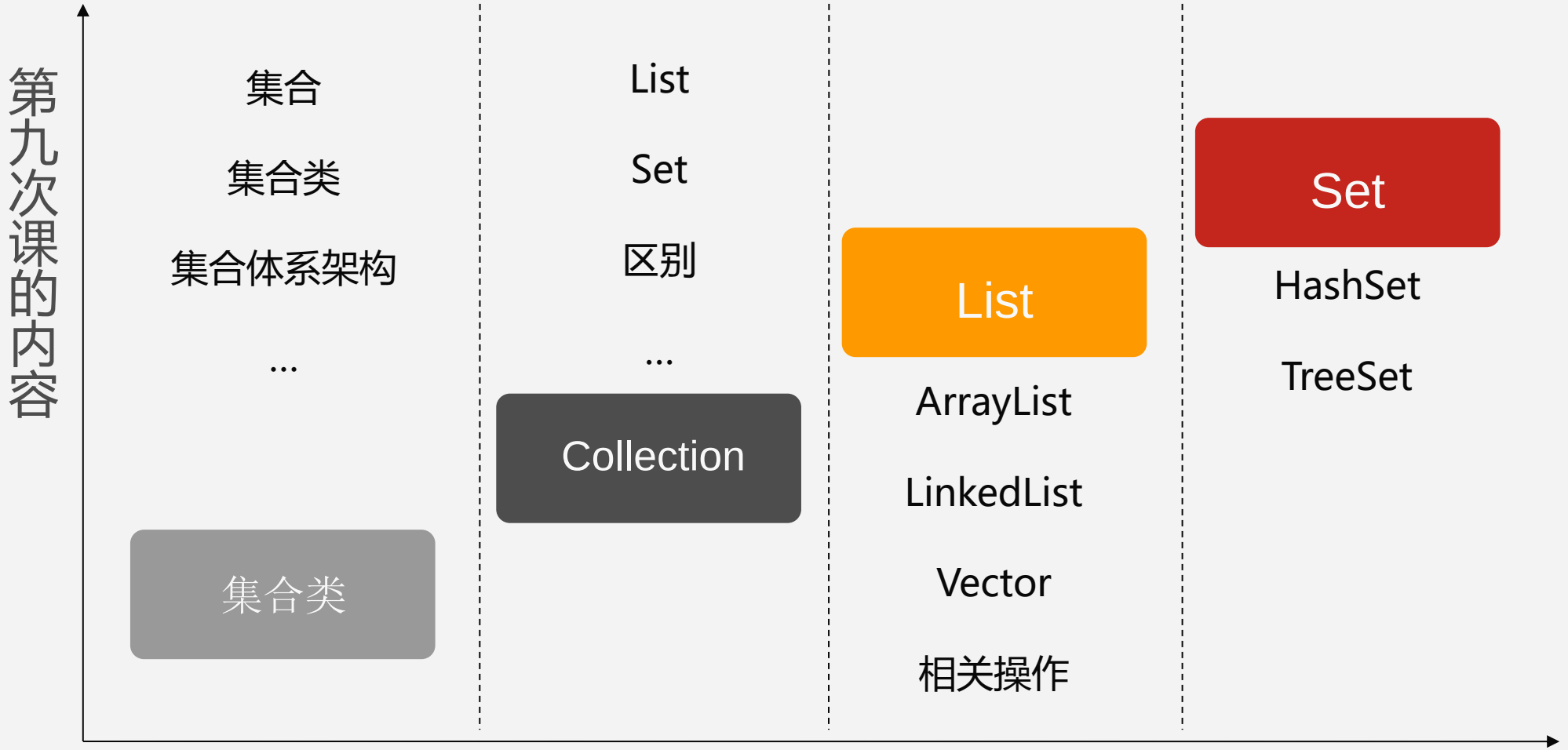
Java

面向对象程序设计

软件学院 贾伟峰

复习


第九次课的内容
集合
集合类
集合体系架构
...
集合类
List
Set
区别
...
Collection
List
ArrayList
LinkedList
Vector
相关操作
Set
HashSet
TreeSet

以ArrayList改造后的生成通话记录代码

```
13      //模拟通话记录的生成
14      void generateCommunicateRecord() {
15          //随机生成通话记录数目
16          int recordNum = new Random().nextInt(10);
17          for(int i = 0; i <= recordNum; i++) {
18              //随机生成第i条通话记录
19              //用Calendar获取当前时间
20              Calendar cal = Calendar.getInstance();
21              //随机减去若干小时（10小时以内）
22              cal.add(Calendar.HOUR, - new Random().nextInt(10));
23              //获得对应毫秒
24              long timeStart = cal.getTimeInMillis();
25              //结束时间开始后的十分钟内随机的一个时间，至少一分钟
26              long timeEnd = timeStart + 60000 + new Random().nextInt(600000);
27
28              //被叫号码
29              this.callTo = getCallToPhoneNumber();
30              //插入通话记录
31              this.communicationRecords.add(this.phoneNumber +
32                      "," + timeStart +
33                      "," + timeEnd +
34                      "," + this.callTo);
35          }
36      }
```

往ArrayList中添加记录，注意对比StringBuffer

打印的时候可以这样，每条记录需要分割。

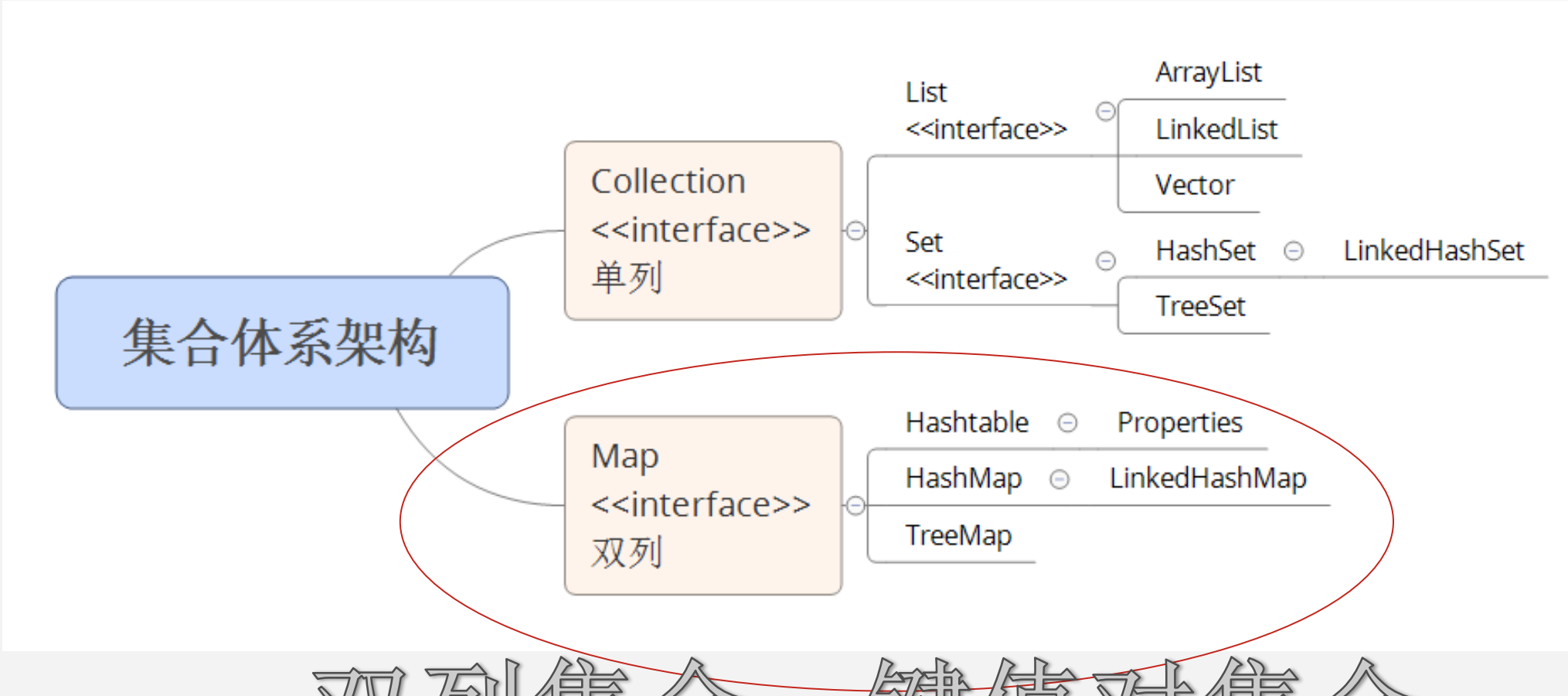
```
56  //打印通话记录
57  void printDetails() {
58      //获取记录数目，即communicationRecords集合中的元素个数
59      int arrayListSize = this.communicationRecords.size();
60
61      for(int i = 0; i < arrayListSize - 1; i++) {
62          System.out.println("----------通话记录分割线----------");
63          String [] recordField = ((String) this.communicationRecords.get(i)).
64          System.out.println("主叫：" + recordField[0]);
65          System.out.println("被叫：" + recordField[3]);
66          Date timeStart = new Date(Long.parseLong(recordField[1]));
67          Date timeEnd = new Date(Long.parseLong(recordField[2]));
68          SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy年MM月dd
69
70          //SimpleDateFormat
71          System.out.println("通话开始时间：" + simpleDateFormat.format(timeStart))
72          System.out.println("通话结束时间：" + simpleDateFormat.format(timeEnd));
73
74          System.out.println("计费："
75                  + accountFee(Long.parseLong(recordField[1]), Long.parseLong(
76                  + " 元。");
77      }
78  }
79
```

注意代码改动的地方，右边没显示完整的应该怎么补充？

能否存储通话记录的时候以这种形式存储（Key-Value键值对）？

主叫：13800138000
开始时间：2017-11-26 09:22:30
结束时间：2017-11-26 09:22:30
被叫：13803720003

Java 集合体系架构



双列集合：键值对集合

直接存储键值对，代码改如何更改呢？

使用 HashMap + ArrayList 存储通话记录

```
private ArrayList communicationRecords;
private HashMap singleRecord;
```

```
//插入通话记录
this.singleRecord = new HashMap();
this.singleRecord.put("主叫", this.phoneNumber);
this.singleRecord.put("开始时间", new Date(timeStart));
this.singleRecord.put("结束时间", new Date(timeEnd));
this.singleRecord.put("被叫号码", this.callTo);
this.singleRecord.put("计费", this.accountFee(timeStart, timeEnd)
this.communicationRecords.add(this.singleRecord);
```

打印详单的时候，怎么把HashMap中的内容打出来呢?

```
//打印通话记录
void printDetails() {
    /*
     * 使用ArrayList接口遍历。注意应将communicationRecords定义为ArrayList
     * 遍历打印每个通话、计费记录（HashMap对象）
     */
    Iterator itRecords = this.communicationRecords.iterator();
    while(itRecords.hasNext()) {
        System.out.println("----------通话记录分割线----------");
        this.singleRecord = ((HashMap)itRecords.next());
        Set keySet = this.singleRecord.keySet();
        Iterator itKey = keySet.iterator();
        while(itKey.hasNext()) {
            Object key = itKey.next();
            Object value = this.singleRecord.get(key);
            System.out.println(key + ":" + value);
        }

    }
```

取出键值对集合

取出键的集合

逐一迭代每个键

通过键获取对应的值

或者用这种办法打印，请对比

```
//打印通话记录
void printDetails() {
    /*
     * 使用ArrayList接口遍历。注意应将communicationRecords定义为ArrayList
     * 遍历打印每个通话、计费记录（HashMap对象）
     */

    /*
     * 使用entrySet遍历通话记录
     */
    Iterator itRecords = this.communicationRecords.iterator();
    while(itRecords.hasNext()) {
        System.out.println("----------通话记录分割线----------");
        this.singleRecord = ((HashMap)itRecords.next());
        Set entrySet = this.singleRecord.entrySet();
        Iterator itEntry = entrySet.iterator();
        while(itEntry.hasNext()) {
            Map.Entry entry = (Map.Entry)itEntry.next();
            Object key = entry.getKey();
            Object value = entry.getValue();
            System.out.println(key + ":" + value);
        }

    }
```

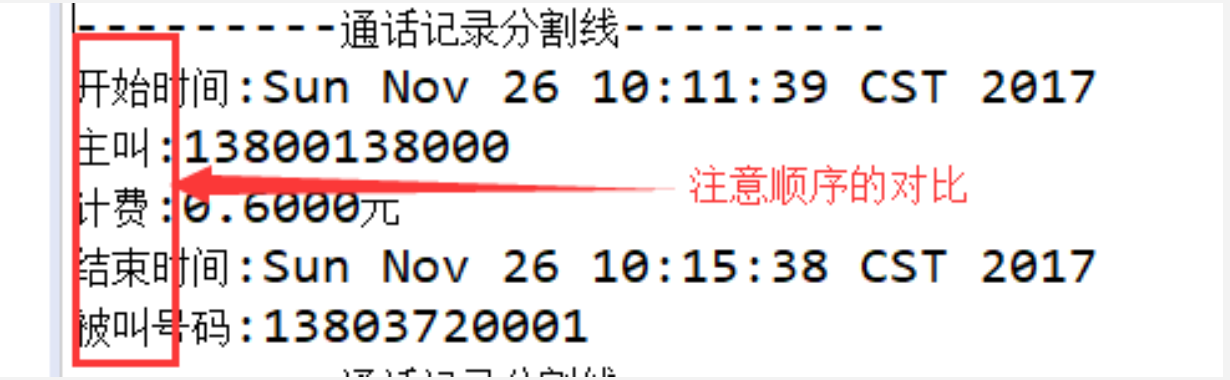
以Set的形式返回键值对集合

Entry，键值对类

拿到Entry的键和值

数据放入时的顺序、数据取出时的顺序对比

```
//插入通话记录
this.singleRecord = new HashMap();
this.singleRecord.put("主叫", this.phoneNumber);
this.singleRecord.put("开始时间", new Date(timeStart));
this.singleRecord.put("结束时间", new Date(timeEnd));
this.singleRecord.put("被叫号码", this.callTo);
this.singleRecord.put("计费", this.accountFee(timeStart, timeEnd)
this.communicationRecords.add(this.singleRecord);
```



想要保持数据插入时的顺序，怎么办？

LinkedHashMap

LinkedHashMap与HashMap的数据取出顺序

主叫:13800138000
开始时间:Sun Nov 26 11:49:40 CST 2017
结束时间:Sun Nov 26 11:54:22 CST 2017
被叫号码:13803720005
计费:0.8000元

LinkedHashMap

----------通话记录分割线---------
开始时间:Sun Nov 26 10:11:39 CST 2017
主叫:13800138000
计费:0.6000元
结束时间:Sun Nov 26 10:15:38 CST 2017
被叫号码:13803720001

注意顺序的对比

HashMap

TreeMap的有序性

假定有这样一个双列集合：（被叫号码，话费），请按号码排序

代码这样写

```
//插入通话记录
this.singleRecord = new LinkedHashMap();
this.singleRecord.put("主叫", this.phoneNumber);
this.singleRecord.put("开始时间", new Date(timeStart));
this.singleRecord.put("结束时间", new Date(timeEnd));
this.singleRecord.put("被叫号码", this.callTo);
this.singleRecord.put("计费", this.accountFee(timeStart, timeEnd)+"元");
this.communicationRecords.add(this.singleRecord);

this.treeMapCallToAndFee.put(this.callTo, this.accountFee(timeStart, timeEnd));
```

仔细想想：是否存在什么问题呢？

```
void printCallToAndFee() {
    Iterator it = this.treeMapCallToAndFee.entrySet().iterator();
    while(it.hasNext()) {
        Map.Entry entry = (Map.Entry) it.next();
        System.out.println(entry.getKey()+":"+entry.getValue());
    }
}
```

然后，打印的时候用这个方法。

执行右下角两行代码，对比结果，找问题

```
结束时间:Sun Nov 26 03:55:48 CST 2017
被叫号码:13803720009
计费:2.0000元
---------通话记录分割线---------
主叫:13800138000
开始时间:Sun Nov 26 06:45:27 CST 2017
结束时间:Sun Nov 26 06:54:45 CST 2017
被叫号码:13803720009
计费:1.8000元
---------通话记录分割线---------
主叫:13800138000
开始时间:Sun Nov 26 06:45:27 CST 2017
结束时间:Sun Nov 26 06:52:40 CST 2017
被叫号码:13803720004
计费:1.4000元
---------通话记录分割线---------
主叫:13800138000
开始时间:Sun Nov 26 08:45:27 CST 2017
结束时间:Sun Nov 26 08:52:05 CST 2017
被叫号码:13803720007
计费:1.2000元
```

重复的被叫号码

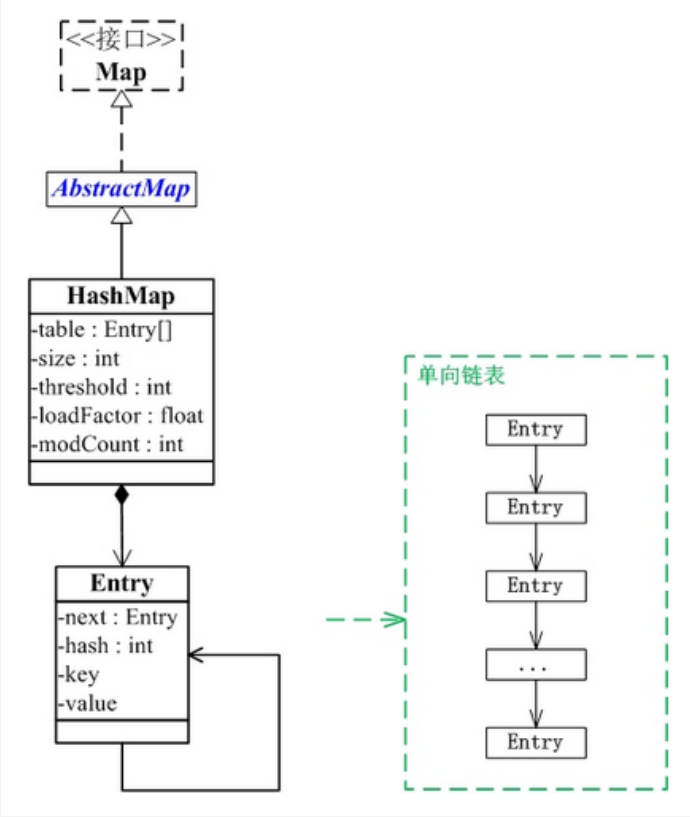
Map中的Key重复了！

```
13803720004:1.4000
13803720007:1.2000
13803720009:1.8000
```

对应的TreeMap输出结果

```
//打印通话详单
telcomUser.printDetails();
//打印被叫号码和费用
telcomUser.printCallToAndFee();
```

讨论：HashMap的key冲突处理（源码）




```java
// 将"key-value"添加到HashMap中
public V put(K key, V value) {
    // 若"key为null"，则将该键值对添加到table[0]中。
    if (key == null)
        return putForNullKey(value);
    // 若"key不为null"，则计算该key的哈希值，然后将其添加到该哈希值对应的链表中。
    int hash = hash(key.hashCode());
    int i = indexFor(hash, table.length);
    for (Entry<K,V> e = table[i]; e != null; e = e.next) {
        Object k;
        // 若"该key"对应的键值对已经存在，则用新的value取代旧的value。然后退出
        if (e.hash == hash && ((k = e.key) == key || key.equals(k))) {
            V oldValue = e.value;
            e.value = value;
            e.recordAccess(this);
            return oldValue;
        }
    }

    // 若"该key"对应的键值对不存在，则将"key-value"添加到table中
    modCount++;
    addEntry(hash, key, value, i);
    return null;
}
```

```java
// 新增Entry。将"key-value"插入指定位置，bucketIndex是位置索引。
void addEntry(int hash, K key, V value, int bucketIndex) {
    // 保存"bucketIndex"位置的值到"e"中
    Entry<K,V> e = table[bucketIndex];
    // 设置"bucketIndex"位置的元素为"新Entry"，
    // 设置"e"为"新Entry的下一个节点"
    table[bucketIndex] = new Entry<K,V>(hash, key, value, e);
    // 若HashMap的实际大小 不小于 "阈值"，则调整HashMap的大小
    if (size++ >= threshold)
        resize(2 * table.length);
}
```

Hashtable与HashMap, 类似于Vector和ArrayList

Hashtable, Vector是线程安全的

但是HashMap, ArrayList用途更广泛

Properties是Hashtable的子类

字符类型键值集合

常做应用配置：p.264

泛型：还记得这个警告吗（下图中的波浪线）？

```
13      //模拟通话记录的生成
14      void generateCommunicateRecord() {
15          //随机生成通话记录数目
16          int recordNum = new Random().nextInt(10);
17          for(int i = 0; i <= recordNum; i++) {
18              //随机生成第i条通话记录
19              //用Calendar获取当前时间
20              Calendar cal = Calendar.getInstance();
21              //随机减去若干小时（10小时以内）
22              cal.add(Calendar.HOUR, - new Random().nextInt(10));
23              //获得对应毫秒
24              long timeStart = cal.getTimeInMillis();
25              //结束时间开始后的十分钟内随机的一个时间，至少一分钟
26              long timeEnd = timeStart + 60000 + new Random().nextInt(600000);
27
28              //被叫号码
29              this.callTo = getCallToPhoneNumber();
30              //插入通话记录
31              this.communicationRecords.add(this.phoneNumber +
32                      "," + timeStart +
33                      "," + timeEnd +
34                      "," + this.callTo);
35          }
36      }
```

往ArrayList中添加记录，注意对比StringBuffer

泛　型

（1）集合类可以处理任意数据类型；

（2）集合类可以对入口不做限制（即什么都可以装进去）；

（3）集合类取出时统一成Object；

（4）集合类中的数据取出后，需确定具体类型，以便参与运算；

（5）如果知道数据类型是什么，可以进行强制类型转换。

（6）如果不知道数据类型是什么，强制转换的时候可能会出错!

能不能让集合类把好入口关？也就是说，规定集合里装入数据的类型。

泛型类型的确定

```
private ArrayList <HashMap>communicationRecords;
private LinkedHashMap singleRecord;
private TreeMap treeMapCallToAndFee;
private TreeSet callToNumbersSet;
private ArrayList callToNumbersList;
public TelcomUserBasedOnCollection(String phoneNumber) {
    this.phoneNumber = phoneNumber;
    this.communicationRecords = new ArrayList<HashMap>();
    this.callToNumbersSet = new TreeSet();
    this.callToNumbersList = new ArrayList();
    this.treeMapCallToAndFee = new TreeMap();
}
```

哪里有警告线，哪里就需确定类型。

自定义泛型：小需求——设计一个简单容器，存储一些数据，类型未知

```java
class CachePool <T>{
    T temp;
    public void save(T temp) {
        this.temp = temp;
    }
    public T get() {
        return temp;
    }
}
```

```java
public class CachePoolTest {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        CachePool <String> cp = new CachePool<String>();

        cp.save("Hello");

        System.out.println(cp.get());
    }

}
```

讨论：你觉得什么时候会用到泛型？

Collections工具类（P269 – 270）

```
ArrayList list = new ArrayList();
//一次性添加
Collections.addAll(list, "C", "Z", "B", "K");
//逆序
Collections.reverse(list);
//排序
Collections.sort(list);
//洗牌
Collections.shuffle(list);
//最大
Collections.max(list);
//最小
Collections.min(list);
//替换
Collections.replaceAll(list, "C", "D");
```

Arrays工具类

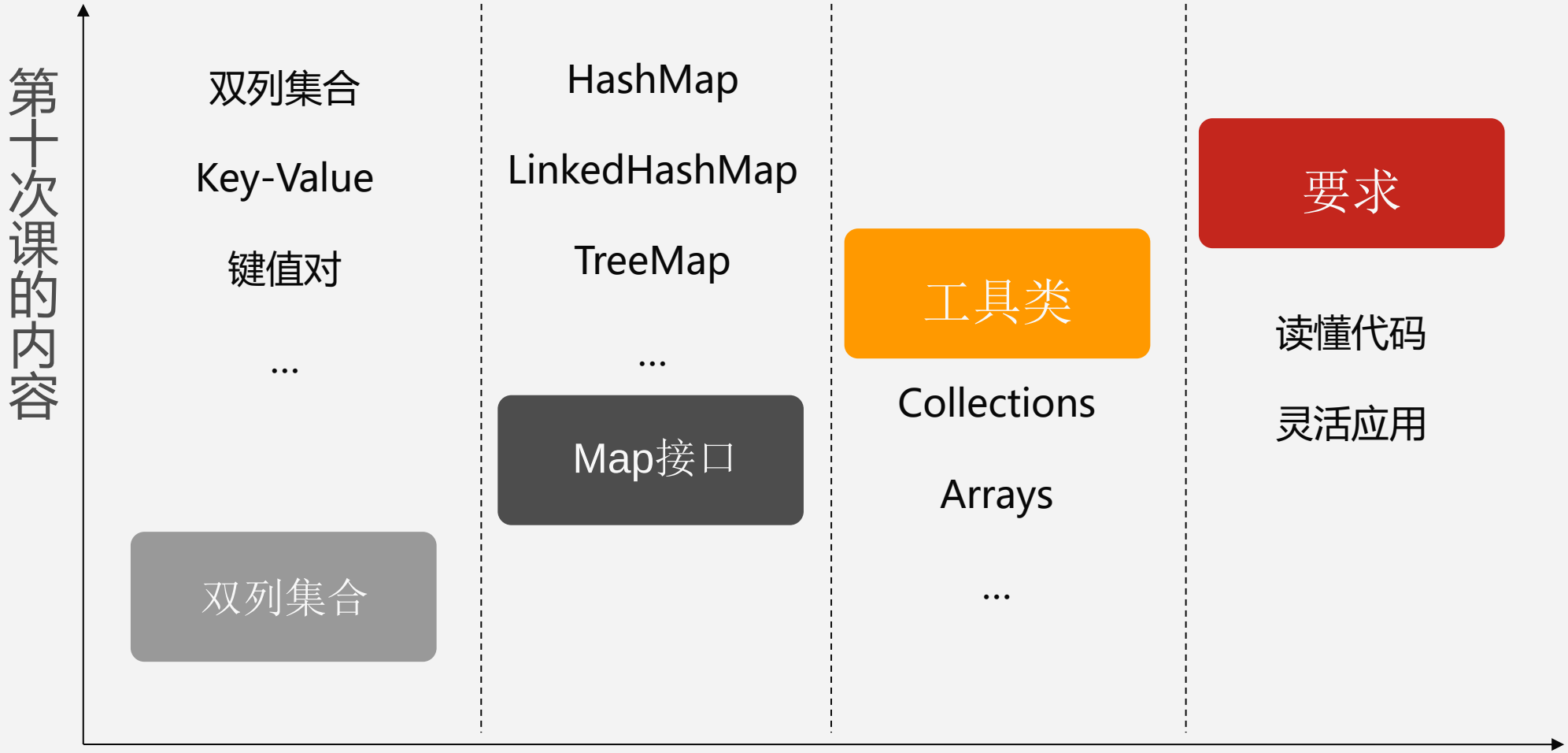
```java
int [] arr = {2,3,5,9,1};
//排序
Arrays.sort(arr);
for(int i = 0; i<arr.length; i++)
    System.out.println(arr[i]);
//查找
int index = Arrays.binarySearch(arr, 5);
System.out.println(index);
//复制
int [] subArr = Arrays.copyOfRange(arr, 1, 2);
for(int i = 0; i<subArr.length; i++)
    System.out.println(subArr[i]);
//填充
Arrays.fill(arr, 8);
for(int i = 0; i<arr.length; i++)
    System.out.println(arr[i]);
//转为字符串
String strArr = Arrays.toString(arr);
System.out.println(strArr);
```

代码是否有改进空间？

```
1
2
3
5
9
3
2
8
8
8
8
8
[8, 8, 8, 8, 8]
```

这是输出结果，能解释下吗？

总结


第十次课的内容
双列集合
Key-Value
键值对
...
双列集合
HashMap
LinkedHashMap
TreeMap
...
Map接口
工具类
Collections
Arrays
...
要求
读懂代码
灵活应用