

# 学习任务一 技能节报名小程序

## 子任务 1.3 信息排序与查找

### 1、任务引入（微课学习内容）

我们将数据存储在集合中，最终的目的是为了排序和查找。例如我们在集合中存储了技能节选手的信息，就是为了选手可以查询到自己的分数，最后能看到分数的排名。在 Java 中集合常常用来临时存储来自外部的数据，这个子任务就是要将报名所得的学生信息存储在集合中，方便我们一步的修改和查询。

集合信息的查找，首先要学会遍历集合，也就是从头到尾的把集合的数据访问一遍，如下图所示：

```
public class ListDemo {  
    public static void main(String[] args) {  
        List<String> aList = new LinkedList<String>();  
        // aList = new ArrayList<String>();  
        aList.add("张三锋1");  
        aList.add("张三锋2");  
        aList.add("张三锋3");  
        aList.add("张三锋4");  
        for(String iString : aList){  
            System.out.println(iString);  
        }  
    }  
}
```

当然，对于 Map 类型的结合，可以直接通过键值快速查找对象，如下图所示：

```
public void mapdemo(){  
    Map<String,String> map = new HashMap<String, String>();  
    map.put("aa", "11");  
    map.put("bb", "22");  
    map.put("cc", "33");  
    //Map的第一种遍历方式：先获得key,再获得value  
    Set<String> sett = map.keySet();  
    for (String s : sett) {  
        System.out.println(s+" "+map.get(s));  
    }  
    System.out.println("-----");  
    //Map的第二种遍历方式：获得键值对  
    for (Map.Entry<String, String> entry : map.entrySet()) {  
        System.out.println(entry.getKey()+" : "+entry.getValue());  
    }  
}
```

集合信息的排序，需要先为放入排序集合的对象类型设计好比较器，让这种对象具有比较的能力，这样当这个对象被添加到排序集合中的时候，集合就会先比较一下新成员和老成员，并为新成员安排好合适的位置，这样排序集合中的对象一直是排序状态。

例如，我们先为学生设计如下的比较器，其实就是实现一个比较接口。

```
public class Student implements Comparable<Student> {  
    private int age;  
    private float height;  
    private String name;
```

利用接口，让学生对象本身具备比较的能力

```
@Override  
public int compareTo(Student student) {  
    // TODO Auto-generated method stub  
    int resultAge = this.age - student.age;  
    if(resultAge!=0){  
        return resultAge;  
    }  
    float resultHeight = this.height - student.height;  
    if(resultHeight != 0){  
        return Float.floatToIntBits(resultHeight);  
    }  
    return this.name.compareTo(student.name);  
}
```

然后，就可以将学生对象放入 TreeSet 类型的集合中，再进行输出，如下图。

```
public class MyText3 {  
    public static void main(String[] args) {  
        TreeSet<Student> treeset = new TreeSet<>();  
        treeset.add(new Student( name: "李1", age: 40));  
        treeset.add(new Student( name: "李12", age: 28));  
        treeset.add(new Student( name: "李123", age: 22));  
        treeset.add(new Student( name: "李1234", age: 33));  
        treeset.add(new Student( name: "李12345", age: 24));  
        treeset.add(new Student( name: "李123456", age: 25));  
        for (Student student : treeset) {  
            System.out.println(student);  
        }  
    }  
}
```

那么 TreeSet 里的学生就会自动按照年龄排序了。

#### 【思考题】

1、对象如何具有比较的能力？

- 2、Comparable 接口中需要实现的方法是什么？
  - 3、哪些集合具有排序的能力？
  - 4、Collections 里面用来排序的方法如何使用？
- 

## 2、技能训练

### 1) 任务需求

请结合之前学习的集合相关内容，完成如下程序设计任务：

```
请输入学生的学号： 12
请输入学生的姓名： 123
请输入学生的成绩： 80
*****
已经报名的学生信息如下：
学号   姓名   成绩
14     ee    85
12     123   80
13     aa    78
*****
请输入你要查询的学号： 12
您要查询的信息如下：
学号   姓名   成绩
12     123   80
*****
```

### 2) 任务拓展

看看自己是否有能力完成进阶版的技能节计分系统，需求如下：

```
——请选择您要的操作：——  
——1 、报名：——  
——2、计分：——  
——3 、查分：——  
——4 、退出：——  
  
1  
*****  
请输入您的学号  
201  
请输入您的姓名  
zs  
请输入您的班级  
1  
请输入您的系部  
ai  
*****报名完成*****
```

```
——请选择您要的操作：——  
——1 、报名：——  
——2、计分：——  
——3 、查分：——  
——4 、退出：——  
  
2  
*****  
请输入您要加分的学号  
201  
请输入您要加的分  
20  
zs当前分数是：20  
——请选择您要的操作：——  
——1 、报名：——  
——2、计分：——  
——3 、查分：——  
——4 、退出：——  
  
3  
学号      姓名      班级      系部      成绩  
2         2         2         2         200  
1         1         1         1         100  
201      zs         1         ai         20  
1         1         1         1         0  
2         2         2         2         0
```