

1.ArrayList

集合和数组的区别：

- 共同点：都是存储数据的容器
- 不同点：数组的容量是固定的，集合的容量是可变的

1.1 -ArrayList的构造方法和添加方法

| | |
|--------------------------------------|-------------------|
| public ArrayList() | 创建一个空的集合对象 |
| public boolean add(E e) | 将指定的元素追加到此集合的末尾 |
| public void add(int index,E element) | 在此集合中的指定位置插入指定的元素 |

ArrayList：

- 可调整大小的数组实现
- ：是一种特殊的数据类型，泛型。

怎么用呢？

- 在出现E的地方我们使用引用数据类型替换即可
- 举例：ArrayList, ArrayList

1.2ArrayList类常用方法【应用】

成员方法：

| | |
|-----------------------------------|---------------------|
| public boolean remove(Object o) | 删除指定的元素，返回删除是否成功 |
| public E remove(int index) | 删除指定索引处的元素，返回被删除的元素 |
| public E set(int index,E element) | 修改指定索引处的元素，返回被修改的元素 |
| public E get(int index) | 返回指定索引处的元素 |
| public int size() | 返回集合中的元素的个数 |

示例代码：

```
1 public class ArrayListDemo02 {
2     public static void main(String[] args) {
3         //创建集合
4         ArrayList<String> array = new ArrayList<String>();
5     }
```

```

6      //添加元素
7      array.add("hello");
8      array.add("world");
9      array.add("java");
10
11     //public boolean remove(Object o): 删除指定的元素, 返回删除是否成功
12     //      System.out.println(array.remove("world"));
13     //      System.out.println(array.remove("javaee"));
14
15     //public E remove(int index): 删除指定索引处的元素, 返回被删除的元素
16     //      System.out.println(array.remove(1));
17
18     //IndexOutOfBoundsException
19     //      System.out.println(array.remove(3));
20
21     //public E set(int index,E element): 修改指定索引处的元素, 返回被修改的元素
22     //      System.out.println(array.set(1,"javaee"));
23
24     //IndexOutOfBoundsException
25     //      System.out.println(array.set(3,"javaee"));
26
27     //public E get(int index): 返回指定索引处的元素
28     //      System.out.println(array.get(0));
29     //      System.out.println(array.get(1));
30     //      System.out.println(array.get(2));
31     //System.out.println(array.get(3)); //? ? ? ? ? 自己测试
32
33     //public int size(): 返回集合中的元素的个数
34     System.out.println(array.size());
35
36     //输出集合
37     System.out.println("array:" + array);
38 }
39 }

```

1.3 ArrayList存储字符串并遍历

案例需求：

创建一个存储字符串的集合，存储3个字符串元素，使用程序实现在控制台遍历该集合

实现步骤：

1:创建集合对象 2:往集合中添加字符串对象 3:遍历集合，首先要能够获取到集合中的每一个元素，这个通过get(int index)方法实现 4:遍历集合，其次要能够获取到集合的长度，这个通过size()方法实现 5:遍历集合的通用格式

代码实现：

```

1  /*
2      思路：
3          1:创建集合对象
4          2:往集合中添加字符串对象
5          3:遍历集合，首先要能够获取到集合中的每一个元素，这个通过get(int index)方法实现

```

```

6      4:遍历集合，其次要能够获取到集合的长度，这个通过size()方法实现
7      5:遍历集合的通用格式
8      */
9      public class ArrayListTest01 {
10         public static void main(String[] args) {
11             //创建集合对象
12             ArrayList<String> array = new ArrayList<String>();
13
14             //往集合中添加字符串对象
15             array.add("刘正风");
16             array.add("左冷禅");
17             array.add("风清扬");
18
19             //遍历集合，其次要能够获取到集合的长度，这个通过size()方法实现
20             //      System.out.println(array.size());
21
22             //遍历集合的通用格式
23             for(int i=0; i<array.size(); i++) {
24                 String s = array.get(i);
25                 System.out.println(s);
26             }
27         }
28     }

```

1.4 ArrayList存储学生对象并遍历

案例需求：

创建一个存储学生对象的集合，存储3个学生对象，使用程序实现在控制台遍历该集合

实现步骤：

- 1:定义学生类
- 2:创建集合对象
- 3:创建学生对象
- 4:添加学生对象到集合中
- 5:遍历集合，采用通用遍历格式实现

代码实现：

```

1      /*
2          思路：
3          1:定义学生类
4          2:创建集合对象
5          3:创建学生对象
6          4:添加学生对象到集合中
7          5:遍历集合，采用通用遍历格式实现
8      */
9      public class ArrayListTest02 {
10         public static void main(String[] args) {
11             //创建集合对象

```

```

12     ArrayList<Student> array = new ArrayList<>();
13
14     //创建学生对象
15     Student s1 = new Student("林青霞", 30);
16     Student s2 = new Student("风清扬", 33);
17     Student s3 = new Student("张曼玉", 18);
18
19     //添加学生对象到集合中
20     array.add(s1);
21     array.add(s2);
22     array.add(s3);
23
24     //遍历集合，采用通用遍历格式实现
25     for (int i = 0; i < array.size(); i++) {
26         Student s = array.get(i);
27         System.out.println(s.getName() + "," + s.getAge());
28     }
29 }
30 }

```

1.5 键盘录入学生信息到集合

案例需求：

创建一个存储学生对象的集合，存储3个学生对象，使用程序实现在控制台遍历该集合

学生的姓名和年龄来自于键盘录入

实现步骤：

- 1:定义学生类，为了键盘录入数据方便，把学生类中的成员变量都定义为String类型
- 2:创建集合对象
- 3:键盘录入学生对象所需要的数据
- 4:创建学生对象，把键盘录入的数据赋值给学生对象的成员变量
- 5:往集合中添加学生对象
- 6:遍历集合，采用通用遍历格式实现

代码实现：

```

1  /*
2      思路：
3          1:定义学生类，为了键盘录入数据方便，把学生类中的成员变量都定义为String类型
4          2:创建集合对象
5          3:键盘录入学生对象所需要的数据
6          4:创建学生对象，把键盘录入的数据赋值给学生对象的成员变量
7          5:往集合中添加学生对象
8          6:遍历集合，采用通用遍历格式实现
9      */
10 public class ArrayListTest {
11     public static void main(String[] args) {

```

```

12      //创建集合对象
13      ArrayList<Student> array = new ArrayList<Student>();
14
15      //为了提高代码的复用性，我们用方法来改进程序
16      addStudent(array);
17      addStudent(array);
18      addStudent(array);
19
20      //遍历集合，采用通用遍历格式实现
21      for (int i = 0; i < array.size(); i++) {
22          Student s = array.get(i);
23          System.out.println(s.getName() + "," + s.getAge());
24      }
25  }
26
27  /*
28      两个明确：
29          返回值类型：void
30          参数：ArrayList<Student> array
31  */
32  public static void addStudent(ArrayList<Student> array) {
33      //键盘录入学生对象所需要的数据
34      Scanner sc = new Scanner(System.in);
35
36      System.out.println("请输入学生姓名:");
37      String name = sc.nextLine();
38
39      System.out.println("请输入学生年龄:");
40      String age = sc.nextLine();
41
42      //创建学生对象，把键盘录入的数据赋值给学生对象的成员变量
43      Student s = new Student();
44      s.setName(name);
45      s.setAge(age);
46
47      //往集合中添加学生对象
48      array.add(s);
49  }
50 }

```

2. 学生管理系统

2.1 学生管理系统实现步骤

- 案例需求

针对目前我们的所学内容，完成一个综合案例：学生管理系统！该系统主要功能如下：

添加学生：通过键盘录入学生信息，添加到集合中

删除学生：通过键盘录入要删除学生的学号，将该学生对象从集合中删除

修改学生：通过键盘录入要修改学生的学号，将该学生对象其他信息进行修改

查看学生：将集合中的学生对象信息进行展示

退出系统：结束程序

- **实现步骤**

1. 定义学生类，包含以下成员变量

学生类：Student成员变量：

学号：sid

姓名：name

年龄：age

生日：birthday

构造方法：

无参构造

带四个参数的构造成员方法：

每个成员变量对应给出get/set方法

2. 学生管理系统主界面的搭建步骤

- 2.1 用输出语句完成主界面的编写

- 2.2 用Scanner实现键盘录入数据

- 2.3 用switch语句完成操作的选择

- 2.4 用循环完成再次回到主界面

3. 学生管理系统的添加学生功能实现步骤

- 3.1 用键盘录入选择添加学生

- 3.2 定义一个方法，用于添加学生

显示提示信息，提示要输入何种信息

键盘录入学生对象所需要的数据

创建学生对象，把键盘录入的数据赋值给学生对象的成员变量

将学生对象添加到集合中（保存）

给出添加成功提示

- 3.3 调用方法

4. 学生管理系统的查看学生功能实现步骤

- 4.1 用键盘录入选择查看所有学生信息

- 4.2 定义一个方法，用于查看学生信息

显示表头信息

将集合中数据取出按照对应格式显示学生信息，年龄显示补充“岁”

- 4.3 调用方法

5. 学生管理系统的删除学生功能实现步骤

- 5.1 用键盘录入选择删除学生信息

- 5.2 定义一个方法，用于删除学生信息

显示提示信息

键盘录入要删除的学生学号

调用getIndex方法，查找该学号在集合的索引

如果索引为-1，提示信息不存在

如果索引不是-1，调用remove方法删除并提示删除成功

5.3 调用方法

6. 学生管理系统的修改学生功能实现步骤

6.1 用键盘录入选择修改学生信息

6.2 定义一个方法，用于修改学生信息

显示提示信息

键盘录入要修改的学生学号

调用getIndex方法，查找该学号在集合的索引

如果索引为-1，提示信息不存在

如果索引不是-1，键盘录入要修改的学生信息

集合修改对应的学生信息

给出修改成功提示

6.3 调用方法

7. 退出系统

使用System.exit(0);退出JVM

2.2 学生类的定义

```
1 package com.itheima.domain;
2
3 public class Student {
4     private String sid; // 学号
5     private String name; // 姓名
6     private int age; // 年龄
7     private String birthday; // 生日
8
9     public Student() {
10    }
11
12     public Student(String sid, String name, int age, String birthday) {
13         this.sid = sid;
14         this.name = name;
15         this.age = age;
16         this.birthday = birthday;
17     }
18
19     public String getSid() {
20         return sid;
21     }
22
23     public void setSid(String sid) {
24         this.sid = sid;
25     }
26 }
```

```

25     }
26
27     public String getName() {
28         return name;
29     }
30
31     public void setName(String name) {
32         this.name = name;
33     }
34
35     public int getAge() {
36         return age;
37     }
38
39     public void setAge(int age) {
40         this.age = age;
41     }
42
43     public String getBirthday() {
44         return birthday;
45     }
46
47     public void setBirthday(String birthday) {
48         this.birthday = birthday;
49     }
50 }

```

2.3 测试类的定义

```

1  package com.itheima.test;
2
3  import com.itheima.domain.Student;
4
5  import java.util.ArrayList;
6  import java.util.Scanner;
7
8  public class StudentManager {
9      public static void main(String[] args) {
10
11          Scanner sc = new Scanner(System.in);
12
13          // 创建集合容器对象
14          ArrayList<Student> list = new ArrayList<>();
15
16          lo:
17          while (true) {
18              // 1. 搭建主界面菜单
19              System.out.println("-----欢迎来到学生管理系统-----");
20              System.out.println("1 添加学生");
21              System.out.println("2 删除学生");
22              System.out.println("3 修改学生");
23              System.out.println("4 查看学生");

```



```

24         System.out.println("5 退出");
25         System.out.println("请输入您的选择:");
26
27         String choice = sc.next();
28
29         switch (choice) {
30             case "1":
31                 //System.out.println("添加学生");
32                 addStudent(list);
33                 break;
34             case "2":
35                 //System.out.println("删除学生");
36                 deleteStudent(list);
37                 break;
38             case "3":
39                 //System.out.println("修改学生");
40                 updateStudent(list);
41                 break;
42             case "4":
43                 // System.out.println("查看学生");
44                 queryStudents(list);
45                 break;
46             case "5":
47                 System.out.println("感谢您的使用");
48                 break;
49             default:
50                 System.out.println("您的输入有误");
51                 break;
52         }
53     }
54
55
56 }
57
58 // 修改学生的方法
59 public static void updateStudent(ArrayList<Student> list) {
60     System.out.println("请输入您要修改的学生学号:");
61     Scanner sc = new Scanner(System.in);
62     String updateSid = sc.next();
63     // 3. 调用getIndex方法, 查找该学号在集中出现的索引位置
64     int index = getIndex(list, updateSid);
65     // 4. 根据索引判断, 学号在集中是否存在
66     if (index == -1) {
67         // 不存在: 给出提示
68         System.out.println("查无信息, 请重新输入");
69     } else {
70         // 存在: 接收新的学生信息
71         System.out.println("请输入新的学生姓名:");
72         String name = sc.next();
73         System.out.println("请输入新的学生年龄:");
74         int age = sc.nextInt();
75         System.out.println("请输入新的学生生日:");
76
77         String birthday = sc.next();

```

```

77         // 封装为新的学生对象
78         Student stu = new Student(updateSid, name, age, birthday);
79         // 调用集合的set方法, 完成修改
80         list.set(index, stu);
81         System.out.println("修改成功!");
82     }
83 }
84
85 // 删除学生的方法
86 public static void deleteStudent(ArrayList<Student> list) {
87     // 1. 给出提示信息 (请输入您要删除的学号)
88     System.out.println("请输入您要删除的学生学号:");
89     // 2. 键盘接收要删除的学号
90     Scanner sc = new Scanner(System.in);
91     String deleteSid = sc.next();
92     // 3. 调用getIndex方法, 查找该学号在集合中出现的索引位置
93     int index = getIndex(list, deleteSid);
94     // 4. 根据索引判断, 学号在集合中是否存在
95     if(index == -1){
96         // 不存在: 给出提示
97         System.out.println("查无信息, 请重新输入");
98     }else{
99         // 存在: 删除
100         list.remove(index);
101         System.out.println("删除成功!");
102     }
103 }
104
105 // 查看学生的方法
106 public static void queryStudents(ArrayList<Student> list) {
107     // 1. 判断集合中是否存在数据, 如果不存在直接给出提示
108     if(list.size() == 0){
109         System.out.println("无信息, 请添加后重新查询");
110         return;
111     }
112     // 2. 存在: 展示表头数据
113     System.out.println("学号\t\t姓名\t年龄\t生日");
114     // 3. 遍历集合, 获取每一个学生对象的信息, 打印在控制台
115     for (int i = 0; i < list.size(); i++) {
116         Student stu = list.get(i);
117         System.out.println(stu.getSid() + "\t" + stu.getName() + "\t" + stu.getAge() +
118             "\t\t" + stu.getBirthday());
119     }
120 }
121
122 // 添加学生的方法
123 public static void addStudent(ArrayList<Student> list) {
124     Scanner sc = new Scanner(System.in);
125     // 1. 给出录入的提示信息
126
127     String sid;
128
129     while(true){

```

```

129         System.out.println("请输入学号:");
130         sid = sc.next();
131
132         int index = getIndex(list, sid);
133
134         if(index == -1){
135             // sid不存在, 学号可以使用
136             break;
137         }
138     }
139
140     System.out.println("请输入姓名:");
141     String name = sc.next();
142     System.out.println("请输入年龄:");
143     int age = sc.nextInt();
144     System.out.println("请输入生日:");
145     String birthday = sc.next();
146     // 2. 将键盘录入的信息封装为学生对象
147     Student stu = new Student(sid,name,age,birthday);
148     // 3. 将封装好的学生对象, 添加到集合容器当中
149     list.add(stu);
150     // 4. 给出添加成功的提示信息
151     System.out.println("添加成功!");
152 }
153
154 /*
155     getIndex : 接收一个集合对象, 接收一个学生学号
156
157     查找这个学号, 在集合中出现的索引位置
158 */
159 public static int getIndex(ArrayList<Student> list, String sid){
160     // 1. 假设传入的学号, 在集合中不存在
161     int index = -1;
162     // 2. 遍历集合, 获取每一个学生对象, 准备进行查找
163     for (int i = 0; i < list.size(); i++) {
164         Student stu = list.get(i);
165         // 3. 获取每一个学生对象的学号
166         String id = stu.getSid();
167         // 4. 使用获取出的学生学号, 和传入的学号(查找的学号)进行比对
168         if(id.equals(sid)){
169             // 存在: 让index变量记录正确的索引位置
170             index = i;
171         }
172     }
173
174     return index;
175 }
176 }

```