1.ArrayList

1.1ArrayList类概述【理解】

- 什么是集合

  提供一种存储空间可变的存储模型，存储的数据容量可以发生改变

- ArrayList集合的特点

  底层是数组实现的，长度可以变化

- 泛型的使用

  用于约束集合中存储元素的数据类型

1.2ArrayList类常用方法【应用】

1.2.1构造方法

方法名	说明
public ArrayList()	创建一个空的集合对象

1.2.2成员方法

方法名	说明
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
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

1.2.3示例代码

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

        //添加元素
        array.add("hello");
        array.add("world");
```

```
        array.add("java");

        //public boolean remove(Object o)：删除指定的元素，返回删除是否成功
//        System.out.println(array.remove("world"));
//        System.out.println(array.remove("javaee"));

        //public E remove(int index)：删除指定索引处的元素，返回被删除的元素
//        System.out.println(array.remove(1));

        //IndexOutOfBoundsException
//        System.out.println(array.remove(3));

        //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
//        System.out.println(array.set(1,"javaee"));

        //IndexOutOfBoundsException
//        System.out.println(array.set(3,"javaee"));

        //public E get(int index)：返回指定索引处的元素
//        System.out.println(array.get(0));
//        System.out.println(array.get(1));
//        System.out.println(array.get(2));
        //System.out.println(array.get(3)); //？？？？？？ 自己测试

        //public int size()：返回集合中的元素的个数
        System.out.println(array.size());

        //输出集合
        System.out.println("array:" + array);
    }
}
```

1.3ArrayList存储字符串并遍历【应用】

1.3.1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

1.3.2代码实现

```
/*
    思路：
        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
 */
public class ArrayListTest01 {
    public static void main(String[] args) {
        //创建集合对象
        ArrayList<String> array = new ArrayList<String>();
```

```
        //往集合中添加字符串对象
        array.add("刘正风");
        array.add("左冷禅");
        array.add("风清扬");

        //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
//        System.out.println(array.size());

        //遍历集合的通用格式
        for(int i=0; i<array.size(); i++) {
            String s = array.get(i);
            System.out.println(s);
        }
    }
}
```

1.4ArrayList存储学生对象并遍历【应用】

1.4.1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

1.4.2代码实现

```
/*
    思路：
        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
 */
public class ArrayListTest02 {
    public static void main(String[] args) {
        //创建集合对象
        ArrayList<Student> array = new ArrayList<>();

        //创建学生对象
        Student s1 = new Student("林青霞", 30);
        Student s2 = new Student("风清扬", 33);
        Student s3 = new Student("张曼玉", 18);

        //添加学生对象到集合中
        array.add(s1);
        array.add(s2);
        array.add(s3);

        //遍历集合，采用通用遍历格式实现
        for (int i = 0; i < array.size(); i++) {
            Student s = array.get(i);
            System.out.println(s.getName() + "," + s.getAge());
        }
    }
```

```
}
```

1.5ArrayList存储学生对象并遍历升级版【应用】

1.5.1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

1.5.2代码实现

```
/*
    思路：
        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
 */
public class ArrayListTest {
    public static void main(String[] args) {
        //创建集合对象
        ArrayList<Student> array = new ArrayList<Student>();

        //为了提高代码的复用性，我们用方法来改进程序
        addStudent(array);
        addStudent(array);
        addStudent(array);

        //遍历集合，采用通用遍历格式实现
        for (int i = 0; i < array.size(); i++) {
            Student s = array.get(i);
            System.out.println(s.getName() + "," + s.getAge());
        }
    }

    /*
        两个明确：
            返回值类型：void
            参数：ArrayList<Student> array
     */
    public static void addStudent(ArrayList<Student> array) {
        //键盘录入学生对象所需要的数据
        Scanner sc = new Scanner(System.in);

        System.out.println("请输入学生姓名:");
        String name = sc.nextLine();

        System.out.println("请输入学生年龄:");
        String age = sc.nextLine();
```

```
            //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
            Student s = new Student();
            s.setName(name);
            s.setAge(age);

            //往集合中添加学生对象
            array.add(s);
        }
    }
```

2.学生管理系统

2.1学生管理系统实现步骤【理解】

- 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

     private String sid // 学生id

     private String name // 学生姓名

     private String age // 学生年龄

     private String address // 学生所在地

  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实现键盘输入 2.3 用switch语句完成选择的功能 2.4 用循环完成功能结束后再次回到主界面

  3. 学生管理系统的添加学生功能实现步骤

     3.1 定义一个方法，接收ArrayList集合 3.2 方法内完成添加学生的功能 ①键盘录入学生信息 ②根据录入的信息创建学生对象 ③将学生对象添加到集合中 ④提示添加成功信息 3.3 在添加学生的选项里调用添加学生的方法

  4. 学生管理系统的查看学生功能实现步骤

     4.1 定义一个方法，接收ArrayList集合 4.2 方法内遍历集合，将学生信息进行输出 4.3 在查看所有学生选项里调用查看学生方法

  5. 学生管理系统的删除学生功能实现步骤

     5.1 定义一个方法，接收ArrayList集合 5.2 方法中接收要删除学生的学号 5.3 遍历集合，获取每个学生对象 5.4 使用学生对象的学号和录入的要删除的学号进行比较,如果相同，则将当前学生对象从集合中删除 5.5 在删除学生选项里调用删除学生的方法

  6. 学生管理系统的修改学生功能实现步骤

6.1 定义一个方法，接收ArrayList集合 6.2 方法中接收要修改学生的学号 6.3 通过键盘录入学生对象所需的信息，并创建对象 6.4 遍历集合，获取每一个学生对象。并和录入的修改学生学号进行比较.如果相同，则使用新学生对象替换当前学生对象 6.5 在修改学生选项里调用修改学生的方法

7. 退出系统

   使用System.exit(0);退出JVM

2.2学生类的定义【应用】

```
public class Student {
    //学号
    private String sid;
    //姓名
    private String name;
    //年龄
    private String age;
    //居住地
    private String address;

    public Student() {
    }

    public Student(String sid, String name, String age, String address) {
        this.sid = sid;
        this.name = name;
        this.age = age;
        this.address = address;
    }

    public String getSid() {
        return sid;
    }

    public void setSid(String sid) {
        this.sid = sid;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAge() {
        return age;
    }

    public void setAge(String age) {
        this.age = age;
    }
```

```
    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }
}
```

2.3测试类的定义【应用】

```
public class StudentManager {
    /*
        1:用输出语句完成主界面的编写
        2:用Scanner实现键盘录入数据
        3:用switch语句完成操作的选择
        4:用循环完成再次回到主界面
    */
    public static void main(String[] args) {
        //创建集合对象，用于保存学生数据信息
        ArrayList<Student> array = new ArrayList<Student>();

        //用循环完成再次回到主界面
        while (true) {
            //用输出语句完成主界面的编写
            System.out.println("--------欢迎来到学生管理系统--------");
            System.out.println("1 添加学生");
            System.out.println("2 删除学生");
            System.out.println("3 修改学生");
            System.out.println("4 查看所有学生");
            System.out.println("5 退出");
            System.out.println("请输入你的选择：");

            //用Scanner实现键盘录入数据
            Scanner sc = new Scanner(System.in);
            String line = sc.nextLine();

            //用switch语句完成操作的选择
            switch (line) {
                case "1":
                    addStudent(array);
                    break;
                case "2":
                    deleteStudent(array);
                    break;
                case "3":
                    updateStudent(array);
                    break;
                case "4":
                    findAllStudent(array);
                    break;
                case "5":
```

```
                System.out.println("谢谢使用");
                System.exit(0); //JVM退出
            }
        }
    }

    //定义一个方法，用于添加学生信息
    public static void addStudent(ArrayList<Student> array) {
        //键盘录入学生对象所需要的数据,显示提示信息，提示要输入何种信息
        Scanner sc = new Scanner(System.in);

        String sid;

        while (true) {
            System.out.println("请输入学生学号：");
            sid = sc.nextLine();

            boolean flag = isUsed(array, sid);
            if (flag) {
                System.out.println("你输入的学号已经被占用，请重新输入");
            } else {
                break;
            }
        }

        System.out.println("请输入学生姓名：");
        String name = sc.nextLine();

        System.out.println("请输入学生年龄：");
        String age = sc.nextLine();

        System.out.println("请输入学生居住地：");
        String address = sc.nextLine();

        //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
        Student s = new Student();
        s.setSid(sid);
        s.setName(name);
        s.setAge(age);
        s.setAddress(address);

        //将学生对象添加到集合中
        array.add(s);

        //给出添加成功提示
        System.out.println("添加学生成功");
    }

    //定义一个方法，判断学号是否被使用
    public static boolean isUsed(ArrayList<Student> array, String sid) {
        //如果与集合中的某一个学生学号相同，返回true;如果都不相同，返回false
        boolean flag = false;
```

```
        for(int i=0; i<array.size(); i++) {
            Student s = array.get(i);
            if(s.getSid().equals(sid)) {
                flag = true;
                break;
            }
        }

        return flag;
    }


    //定义一个方法，用于查看学生信息
    public static void findAllStudent(ArrayList<Student> array) {
        //判断集合中是否有数据，如果没有显示提示信息
        if (array.size() == 0) {
            System.out.println("无信息，请先添加信息再查询");
            //为了让程序不再往下执行，我们在这里写上return;
            return;
        }

        //显示表头信息
        //\t其实是一个tab键的位置
        System.out.println("学号\t\t\t姓名\t\t年龄\t\t居住地");

        //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”
        for (int i = 0; i < array.size(); i++) {
            Student s = array.get(i);
            System.out.println(s.getSid() + "\t" + s.getName() + "\t" + s.getAge() +
"岁\t\t" + s.getAddress());
        }
    }

    //定义一个方法，用于删除学生信息
    public static void deleteStudent(ArrayList<Student> array) {
        //键盘录入要删除的学生学号,显示提示信息
        Scanner sc = new Scanner(System.in);

        System.out.println("请输入你要删除的学生的学号：");
        String sid = sc.nextLine();

        //在删除/修改学生操作前，对学号是否存在进行判断
        //如果不存在，显示提示信息
        //如果存在，执行删除/修改操作

        int index = -1;

        for (int i = 0; i < array.size(); i++) {
            Student s = array.get(i);
            if (s.getSid().equals(sid)) {
                index = i;
                break;
            }
```

```
        }

        if (index == -1) {
            System.out.println("该信息不存在，请重新输入");
        } else {
            array.remove(index);
            //给出删除成功提示
            System.out.println("删除学生成功");
        }
    }

    //定义一个方法，用于修改学生信息
    public static void updateStudent(ArrayList<Student> array) {
        //键盘录入要修改的学生学号，显示提示信息
        Scanner sc = new Scanner(System.in);

        System.out.println("请输入你要修改的学生的学号：");
        String sid = sc.nextLine();

        //键盘录入要修改的学生信息
        System.out.println("请输入学生新姓名：");
        String name = sc.nextLine();
        System.out.println("请输入学生新年龄：");
        String age = sc.nextLine();
        System.out.println("请输入学生新居住地：");
        String address = sc.nextLine();

        //创建学生对象
        Student s = new Student();
        s.setSid(sid);
        s.setName(name);
        s.setAge(age);
        s.setAddress(address);

        //遍历集合修改对应的学生信息
        for (int i = 0; i < array.size(); i++) {
            Student student = array.get(i);
            if (student.getSid().equals(sid)) {
                array.set(i, s);
            }
        }

        //给出修改成功提示
        System.out.println("修改学生成功");
    }
}
```