day07【ArrayList集合】

今日内容

- ArrayList集合

教学目标

- [ ] 能够知道集合和数组的区别
- [ ] 能够使用ArrayList集合存储数据
- [ ] 能够使用ArrayList集合中常用的方法
- [ ] 能够使用ArrayList集合存储字符串并遍历
- [ ] 能够使用ArrayList集合存储自定义对象并遍历

第一章 ArrayList集合

1.1 引入—对象数组

使用学生数组，存储三个学生对象，代码如下：

```
public class Student {
  private String name;
  private int age;
  public Student() {
  }
  public Student(String name, int age) {
    this.name = name;
    this.age = age;
  }
  public String getName() {
    return name;
  }
  publicvoid setName(String name) {
    this.name = name;
  }
  publicint getAge() {
    return age;
  }
  publicvoid setAge(int age) {
    this.age = age;
  }
}
public class TestO1StudentArray {
  public static void main(String[] args) {
    //创建学生数组
    Student[] students = new Student[3];

    //创建学生对象
    Student s1 = new Student("曹操",40);
```

```
        Student s2 = new Student("刘备",35);
        Student s3 = new Student("孙权",30);

        //把学生对象作为元素赋值给学生数组
        students[0] = s1;
        students[1] = s2;
        students[2] = s3;

        //遍历学生数组
        for(int x=0; x<students.length; x++) {
          Student s = students[x];
          System.out.println(s.getName()+"---"+s.getAge());
        }
    }
}
```

到目前为止，我们想存储对象数据，选择的容器，只有对象数组。而数组的长度是固定的，无法适应数据变化的需求。为了解决这个问题，Java提供了另一个容器`java.util.ArrayList`集合类,让我们可以更便捷的存储和操作对象数据。

1.2 ArrayList类概述

`java.util.ArrayList`是大小**可变的数组**的实现，存储在内的数据称为元素。此类提供一些方法来操作内部存储的元素。 `ArrayList`中可不断添加元素，其大小也自动增长。

- 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

1.3 ArrayList类常用方法

构造方法

方法名	说明
public ArrayList()	创建一个空的集合对象

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

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);
    }
}
```

1.4 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

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);
        }
    }
}
```

1.5 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

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());
        }
    }
}
```

1.6 ArrayList存储学生对象并遍历升级版

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

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);
    }
}
```

1.7 ArrayList存储基本数据类型

ArrayList对象不能存储基本类型，只能存储引用类型的数据。类似`<int>`**不能写**，但是存储基本数据类型对应的包装类型是可以的。所以，想要存储基本类型数据，`<>`中的数据类型，必须转换后才能编写，转换写法如下：

基本类型	基本类型包装类
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

我们发现，只有`Integer`和`Character`需要特殊记忆，其他基本类型只是首字母大写即可。那么存储基本类型数据，代码如下：

```java
public class Demo02ArrayListMethod {
  public static void main(String[] args) {
    ArrayList<Integer> list = new ArrayList<Integer>();
    list.add(1);
    list.add(2);
    list.add(3);
    list.add(4);

    System.out.println(list);
  }
}
```