**每日作业卷**

**Java基础第09天**

**ArrayList集合**

# 基础案例

## 训练案例1

### 训练描述：【讲义1.3—ArrayList集合类的构造方法和成员方法】

1. 请按以下要求编写代码：
2. 定义一个只能存储字符串的集合对象；
3. 向集合内添加以下数据：

“孙悟空”

“猪八戒”

“沙和尚”

“铁扇公主”

1. 不用遍历，直接打印集合；
2. 获取第4个元素（注意，是--第4个元素，它的索引是？）
3. 打印一下集合大小；
4. 删除元素“铁扇公主”
5. 删除第3个元素（注意：是--第3个元素）
6. 将元素“猪八戒”改为“猪悟能”
7. 再次打印集合；

### 操作步骤描述

**import** java.util.ArrayList;

**public** **class** Test01 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

ArrayList<String> aString = **new** ArrayList<>();

aString.add("孙悟空");

aString.add("猪八戒");

aString.add("沙和尚");

aString.add("铁扇公主");

System.***out***.println(aString);

aString.remove(3);

aString.remove(2);

aString.set(1, "猪悟能");

System.***out***.println(aString);

}

## }训练案例2

### 训练描述：【讲义1.4—ArrayList案例】

一、 请按以下要求编写代码：

1. 定义Student类，包含以下属性：

学号、姓名、身高

2. 定义MainApp类，包含main()方法；

3. 在main()方法中，创建一个可以存储Student对象的集合对象；

4. 创建以下几个Student对象：

学号 姓名 身高

it001 黄渤 1.72

it002 孙红雷 1.78

it003 章子怡 1.64

it004 杨颖 1.68

5. 将上面几个对象添加到集合中

6. 遍历集合打印每个对象的属性值；

7. 获取并打印集合大小；

8. 获取第3个学员对象，并打印其属性值；

9. 移除掉第4个学员对象；

10. 再次获取并打印集合大小；

11. 将第一个对象替换为以下对象：

it005 撒贝宁 1.67

12. 遍历集合，打印出身高在1.70以上的学员信息；

13. 遍历集合，打印出“孙”姓的学员信息；

### 操作步骤描述

**import** java.awt.print.Printable;

**import** java.util.ArrayList;

**public** **class** Test02 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Test02Student student1 = **new** Test02Student();

student1.Test02Student("it001", "黄渤", "1.72");

Test02Student student2 = **new** Test02Student();

student2.Test02Student("it002", "孙红雷", "1.78");

Test02Student student3 = **new** Test02Student();

student3.Test02Student("it003", "章子怡", "1.64");

Test02Student student4 = **new** Test02Student();

student4.Test02Student("it004", "杨颖", "1.68");

ArrayList<Test02Student> studentgroup = **new** ArrayList<>();

studentgroup.add(student1);

studentgroup.add(student2);

studentgroup.add(student3);

studentgroup.add(student4);

**for**(**int** i = 0; i< studentgroup.size();i++){

System.***out***.println(studentgroup.get(i).toString());

}

System.***out***.println(studentgroup.size());

System.***out***.println();

System.***out***.println(studentgroup.get(2).toString());

System.***out***.println();

studentgroup.remove(3);

**for**(**int** i = 0; i< studentgroup.size();i++){

System.***out***.println(studentgroup.get(i).toString());

}

System.***out***.println(studentgroup.size());

studentgroup.get(0).setName("撒贝宁");

studentgroup.get(0).setNum("it005");

studentgroup.get(0).setHeight("1.67");

System.***out***.println();

**for**(**int** i = 0; i< studentgroup.size();i++){

**if**(Double.*parseDouble*(studentgroup.get(i).getHeight())>1.67){

System.***out***.println(studentgroup.get(i).toString());

}

}

System.***out***.println();

**for**(**int** i = 0; i< studentgroup.size();i++){

**if**(studentgroup.get(i).getName().startsWith("孙")){

System.***out***.println(studentgroup.get(i).toString());

}

}

}

}

**public** **class** Test02Student{

String num;

String name;

String height;

**public** Test02Student() {

**super**();

}

**public** String getNum() {

**return** num;

}

**public** **void** setNum(String num) {

**this**.num = num;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getHeight() {

**return** height;

}

**public** **void** setHeight(String height) {

**this**.height = height;

}

**public** **void** Test02Student(String num, String name, String height) {

**this**.num = num;

**this**.name = name;

**this**.height = height;

}

@Override

**public** String toString() {

**return** num + " "+ name + " " + " " + height;

}

}

it001 黄渤 1.72

it002 孙红雷 1.78

it003 章子怡 1.64

it004 杨颖 1.68

4

it003 章子怡 1.64

it001 黄渤 1.72

it002 孙红雷 1.78

it003 章子怡 1.64

3

it002 孙红雷 1.78

it002 孙红雷 1.78

## 训练案例3

### 训练描述：【综合：Scanner类、方法、ArrayList类综合】

1. 需求说明：程序启动后可以接收三个学员信息，然后可以打印所有学员信息，并找出身高最高的学员，并打印他们的详细信息

### 操作步骤描述

1. 定义MainApp类，包含main()方法；
2. 在main()方法中，创建一个可以存储Student对象的集合对象；
3. 从控制台接收以下几个Student对象：

学号 姓名 性别 年龄 身高

it001 黄渤 男 20 1.72

it002 孙红雷 男 22 1.78

it003 章子怡 女 18 1.64

将每个对象添加到集合中(集合中学员学号唯一)；

1. 在MainApp中创建方法，可以打印所有学员信息：

public static void printAll(ArrayList<Student>stuList){}

1. 在MainApp中创建方法，可以找出身高最高的学员，并打印：

public static void findMaxHeight(ArrayList<Student>stuList){}

1. 在main()方法中，分别调用三个方法：打印所有学员，找出身高最高的学员；

package Test03;

import java.util.ArrayList;

import Test02.Student;

public class MainApp {

public static void printAll(ArrayList<Student>stuList){

for (Student student : stuList) {

System.out.println(student.getName()+" "+student.getId()+" "+student.getHeight());

}

}

public static void findMaxHeight(ArrayList<Student>stuList){

double tallest = 0;

for(int i =0;i<stuList.size()-1;i++){

if(Double.parseDouble(stuList.get(i).getHeight()) > tallest){

tallest = Double.parseDouble(stuList.get(i).getHeight());

}

}

System.out.println("最高身高的的是"+tallest);

}

public static void main(String[] args) {

// TODO Auto-generated method stub

ArrayList<Student> stugroup = new ArrayList<>();

Student student1 = new Student("it001","黄渤","1.72");

Student student2 = new Student("it002","孙红雷","1.78");

Student student3 = new Student("it003","章子怡","1.64");

stugroup.add(student1);

stugroup.add(student2);

stugroup.add(student3);

printAll(stugroup);

findMaxHeight(stugroup);

}

}

**package** Test03;

**public** **class** Student {

**private** String id;

**private** String name;

**private** String height;

**public** Student(String id, String name, String height) {

**super**();

**this**.id = id;

**this**.name = name;

**this**.height = height;

}

**public** Student() {

**super**();

}

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getHeight() {

**return** height;

}

**public** **void** setHeight(String height) {

**this**.height = height;

}

}

# 扩展案例

## 训练案例1

### 训练描述：

分析以下需求，并用代码实现

1.按照以下描述完成类的定义。

学生类

属性：

姓名name

年龄age

成绩score

行为：

吃饭eat()

stuty(String content)(content:表示学习的内容)

2.定义学生工具StudentsTool,有四个方法,描述如下

public void listStudents(Student[] arr):遍历打印学生信息

public int getMaxScore(Student[] arr):获取学生成绩的最高分

public Student getMaxStudent(Student[] arr):获取成绩最高的学员

public int getAverageScore(Student[] arr):获取学生成绩的平均值

public int getCount(Student[] arr):获取不及格的学员数量

3.定义测试类TestStudentTool，在main方法中首先创建长度为5的Student数组并初始化数据,再创建StudentsTool类的对象,并调用以上方法

### 操作步骤描述

**package** Test04;

**public** **class** Student {

**private** String name;

**private** String age;

**private** String score;

**public** Student(String name, String age, String score) {

**super**();

**this**.name = name;

**this**.age = age;

**this**.score = score;

}

**public** Student() {

**super**();

}

**public** **void** eat(){}

**public** **void** study(String content){}

**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 getScore() {

**return** score;

}

**public** **void** setScore(String score) {

**this**.score = score;

}

@Override

**public** String toString() {

**return** "Student [name=" + name + ", age=" + age + ", score=" + score + "]";

}

}

**package** Test04;

**public** **class** StudentTools {

**public** StudentTools(){}

**public** **void** listStudents(Student[] arr){

**for** (Student student : arr) {

String aString = student.toString();

System.***out***.println(aString);

}

}

**public** **int** getMaxScore(Student[] arr){

**int** Topscore = 0;

**for**(**int** i=0;i<arr.length-1;i++){

**if**(Integer.*parseInt*(arr[i].getScore())>0){

Topscore = Integer.*parseInt*(arr[i].getScore());

}

}

**return** Topscore;

}

**public** Student getMaxStudent(Student[] arr){

**int** scroe = **this**.getMaxScore(arr);

Student student = **new** Student();

**for**(**int** i=0;i<arr.length-1;i++){

**if**(Integer.*parseInt*(arr[i].getScore())==scroe){

student = arr[i];

}

}

**return** student;

}

**public** **int** getAverageScore(Student[] arr){

**int** sum = 0;

**for**(**int** i=0;i<arr.length-1;i++){

sum += Integer.*parseInt*(arr[i].getScore());

}

**return** sum/(arr.length);

}

**public** **int** getCount(Student[] arr){

**int** count = 0;

**for**(**int** i=0;i<arr.length-1;i++){

**if**(Integer.*parseInt*(arr[i].getScore())<60){

count++;

}

}

**return** count;

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Student student[] = {**new** Student("1","2","3"),**new** Student("1","2","3"),**new** Student("1","2","3"),**new** Student("1","2","3"),**new** Student("1","2","3")};

StudentTools studentTools =**new** StudentTools();

studentTools.getAverageScore(student);

studentTools.listStudents(student);

studentTools.getMaxScore(student);

studentTools.getCount(student);

studentTools.getMaxStudent(student);

}

}

## 训练案例2

### 训练描述：

分析以下需求，并用代码实现

1.定义Student类

属性:

姓名：String name

年龄：int age

成绩：int score

行为:

空参构造方法

有参构造方法

set和get方法

toString方法

2.定义测试类,进行测试

(1)创建10个学生对象存入ArrayList集合中

(2)打印最高分的学员姓名、年龄、成绩 [要求封装1个方法 参数是集合对象 返回值类型为Student]

(3)打印10个学生的总成绩和平均分 [要求封装两个方法完成]

(4)打印不及格的学员信息及数量 [要求封装一个方法完成]

### 操作步骤描述

**package** Test05;

**import** java.util.ArrayList;

**public** **class** StudentTest {

**public** StudentTest() {}

**public** Student getTopScoreStudentInfo(ArrayList<Student>arr){

Student student = **new** Student();

**int** top=0;

**for**(**int** i=0;i<arr.size();i++){

**if**(Integer.*parseInt*(arr.get(i).getScore())>top){

top = Integer.*parseInt*(arr.get(i).getScore());

student = arr.get(i);

}

}

**return** student;

}

**public** **int** getStudentSum(ArrayList<Student>arr){

**int** sum = 0;

**for**(**int** i=0;i<arr.size();i++){

sum += Integer.*parseInt*(arr.get(i).getScore());

}

**return** sum;

}

**public** **void** getStudentAverage(ArrayList<Student>arr){

**int** sum = **this**.getStudentSum(arr);

**int** average = sum/arr.size();

System.***out***.println();

}

**public** **void** getFlunk(ArrayList<Student>arr){

**for**(**int** i=0;i<arr.size();i++){

**if**(Integer.*parseInt*(arr.get(i).getScore())< 60){

System.***out***.println(arr.get(i).toString());

}

}

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

ArrayList<Student> arr= **new** ArrayList<>();

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

arr.add(**new** Student("1","2","3"));

StudentTest s = **new** StudentTest();

s.getFlunk(arr);

s.getStudentAverage(arr);

s.getTopScoreStudentInfo(arr);

s.getStudentSum(arr);

}

}

**package** Test05;

**public** **class** Student {

**private** String name;

**private** String age;

**private** String score;

**public** Student(String name, String age, String score) {

**super**();

**this**.name = name;

**this**.age = age;

**this**.score = score;

}

**public** Student() {

**super**();

}

**public** **void** eat(){}

**public** **void** study(String content){}

**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 getScore() {

**return** score;

}

**public** **void** setScore(String score) {

**this**.score = score;

}

@Override

**public** String toString() {

**return** "Student [name=" + name + ", age=" + age + ", score=" + score + "]";

}

}

## 训练案例3

### 训练描述：

分析以下需求，并用代码实现

1.定义ArrayList集合，存入多个字符串

如:"ab1" "123ad" "bca" "dadfadf" "dddaaa" "你好啊" "我来啦" "别跑啊"

2.遍历集合,删除长度大于5的字符串,打印删除后的集合对象

3.基于上一步,删除集合中元素包含0-9数字的字符串(只要字符串中包含0-9中的任意一个数字就需要删除此整个字符串)

### 操作步骤描述

略

package Test06;

import java.util.ArrayList;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class MainApp {

public MainApp(){}

public ArrayList<String> clone(ArrayList<String> arr){

ArrayList<String> newarr = new ArrayList<>();

for(int i=0;i<arr.size();i++){

newarr.add(arr.get(i));

}

return newarr;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

ArrayList<String> arr= new ArrayList<>();

ArrayList<String> newarr= new ArrayList<>();

arr.add("ab1");

arr.add("123ad");

arr.add("bca");

arr.add("dadfadf");

arr.add("dddaaa");

arr.add("你好啊");

arr.add("我来啦");

arr.add("你别跑啊");

/\*

\* arr 循环赋值给 newarr

\*

\* 遍历数组 满足条件 删除一个newarr对象

\* 把newarr 赋值 给arr

\* 继续遍历arr 满足条件 删除个newarr对象

\* 循环

\*/

MainApp m = new MainApp();

newarr = m.clone(arr);

System.out.println(newarr);

while(true){

int count = 0;

for(int i=0;i<arr.size();i++){

if(arr.get(i).length()>5){

newarr.remove(i);

break;

}

}

arr = m.clone(newarr);

for(int i=0;i<arr.size();i++){

if(arr.get(i).length()<6){

count++;

}

}

if(count == arr.size()){

break;

}

}

System.out.println(arr);

String reg = "[0-9]+";

Pattern pattern = Pattern.compile(reg);

while(true){

int count = 0;

for(int i=0;i<arr.size();i++){

Matcher matcher= pattern.matcher(arr.get(i));

if(matcher.find()){

newarr.remove(i);

break;

}

}

arr = m.clone(newarr);

for(int i=0;i<arr.size();i++){

Matcher matcher= pattern.matcher(arr.get(i));

if(!(matcher.find())){

count++;

}

}

if(count == arr.size()){

break;

}

}

System.out.println(arr);

}

}

## 训练案例4

### 训练描述：

分析以下需求，并用代码实现

定义MyArrays类，该类中有以下方法，方法描述如下：

1.public static void reverse(ArrayList<Integer> list);

参数ArrayList<Integer> list:要进行操作的集合对象

要求：对list集合对象中的元素进行反转(第一个和最后一个交换，第二个和倒数第二个交换，第三个和倒数第三个交换...)

2.public static Integer max(ArrayList<Integer> list);

参数ArrayList<Integer> list:要进行操作的集合对象

要求：求出list集合对象中的最大值并返回

3.public static Integer min(ArrayList<Integer> list);

参数ArrayList<Integer> list:要进行操作的集合对象

要求：求出list集合对象中的最小值并返回

4.public static int indexOf(ArrayList<Integer> list,Integer i);

参数ArrayList<Integer> list:要进行操作的集合对象

参数Integer i:需要在集合中查找的元素

要求：求出元素i在list集合中第一次出现的索引，如果没有返回-1

5.public static void replaceAll(ArrayList<Integer> list,Integer oldValue,Integer newValue);

参数ArrayList<Integer> list:要进行操作的集合对象

参数Integer oldValue:需要被替换掉的原值

参数Integer newValue：替换后的新值

要求：将list集合中的所有值为oldValue的元素替换为newValue

### 操作步骤描述

**package** Test07;

**import** java.util.ArrayList;

**public** **class** MyArrays {

**public** MyArrays(){}

**public** **static** **void** reverse(ArrayList<Integer> list){

//参数ArrayList<Integer> list:要进行操作的集合对象

//要求：对list集合对象中的元素进行反转(第一个和最后一个交换，第二个和倒数第二个交换，第三个和倒数第三个交换...)

**int** length = list.size();

**int** half = length/2;

**for**(**int** i=0;i<length;i++){

**if**(i<= half){

**int** tem = 0;

tem = list.get(i);

list.set(i,list.get(i+1));

list.set(i+1,tem);

}

}

}

**public** **static** Integer max(ArrayList<Integer> list){

//参数ArrayList<Integer> list:要进行操作的集合对象

//要求：求出list集合对象中的最大值并返回

Integer integer = 0;

**for**(**int** i=0;i<list.size();i++){

**if**(list.get(i)>integer){

integer = list.get(i);

}

}

**return** integer;

}

**public** **static** Integer min(ArrayList<Integer> list){

//参数ArrayList<Integer> list:要进行操作的集合对象

//要求：求出list集合对象中的最小值并返回

Integer min = list.get(0);

**for**(**int** i=0;i<list.size();i++){

**if**(list.get(i) < min){

min = list.get(i);

}

}

**return** min;

}

**public** **static** **int** indexOf(ArrayList<Integer> list,Integer i){

//参数ArrayList<Integer> list:要进行操作的集合对象

//参数Integer i:需要在集合中查找的元素

**int** index = 0;

**for**(**int** j=0;j<list.size();j++){

**if**(list.get(j) == i){

index = j;

**break**;

}

}

**return** index;

}

**public** **static** **void** replaceAll(ArrayList<Integer> list,Integer oldValue,Integer newValue){

//参数ArrayList<Integer> list:要进行操作的集合对象

//参数Integer oldValue:需要被替换掉的原值

//参数Integer newValue：替换后的新值

//要求：将list集合中的所有值为oldValue的元素替换为newValue

**for**(**int** i=0;i<list.size();i++){

**if**(list.get(i)==oldValue){

list.set(i, newValue);

}

}

}

}

**package** Test07;

**public** **class** MainApp {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

}

}