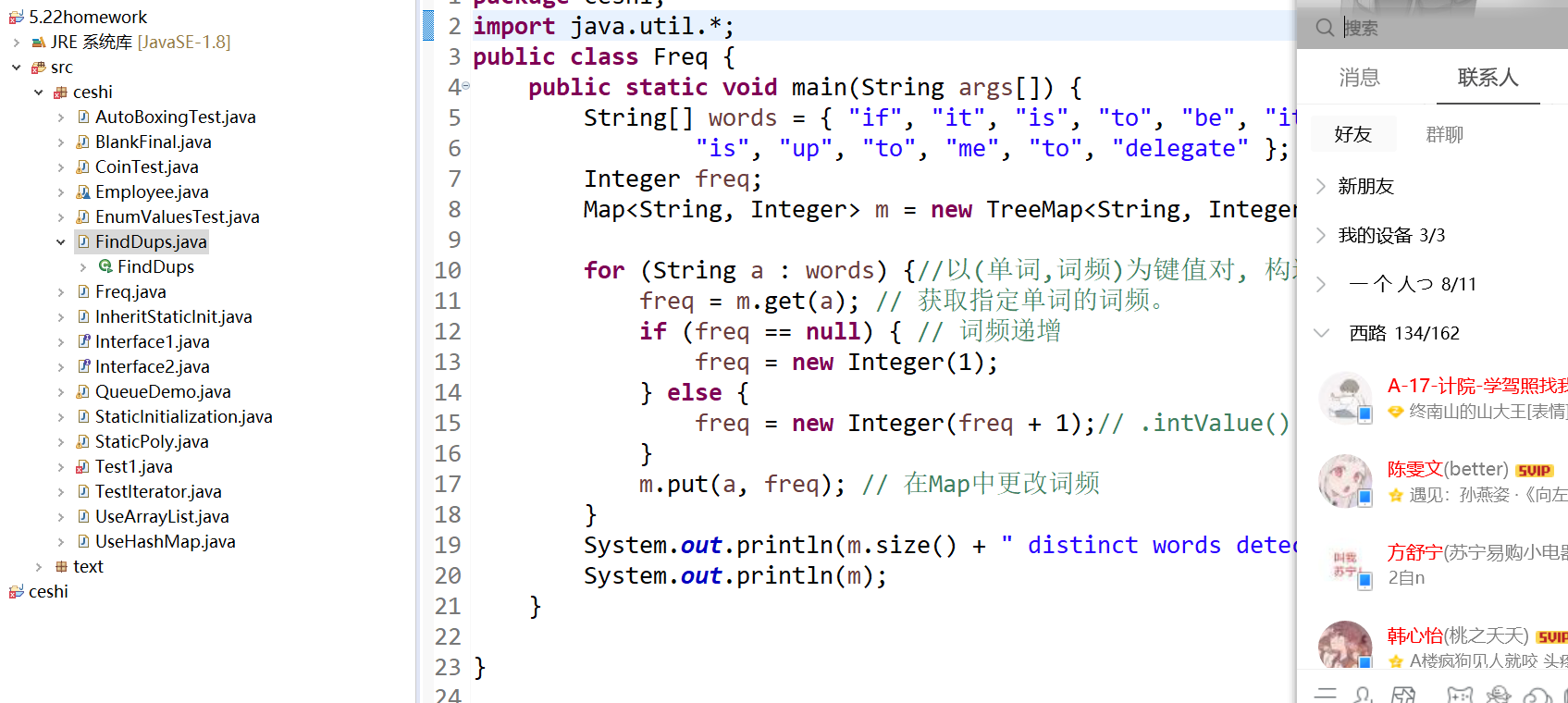
5月22号 上机报告 张帅豪

18030100101



没有贴图，因为代码多，一张图覆盖不了。就贴一张，作为凭证。

Ppt7

源程序

**package** ceshi;

**public** **class** StaticInitialization {

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

System.***out***.println("new Cupboard() in main");

**new** Cupboard();

System.***out***.println("new Cupboard() in main");

**new** Cupboard();

*table*.f2(1);

*cupboard*.f3(1);

}

**static** Table *table* = **new** Table();

**static** Cupboard *cupboard* = **new** Cupboard();

}

**class** Bowl {

Bowl(**int** i) {

System.***out***.println("Bowl(" + i + ")");

}

**void** f1(**int** i) {

System.***out***.println("f1(" + i + ")");

}

}

**class** Table {

**static** Bowl *bowl1* = **new** Bowl(1);

Table() {

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

*bowl2*.f1(1);

}

**void** f2(**int** i) {

System.***out***.println("f2(" + i + ")");

}

**static** Bowl *bowl2* = **new** Bowl(2);

}

**class** Cupboard {

Bowl bowl3 = **new** Bowl(3);

**static** Bowl *bowl4* = **new** Bowl(4);

Cupboard() {

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

*bowl4*.f1(2);

}

**void** f3(**int** i) {

System.***out***.println("f3(" + i + ")");

}

**static** Bowl *bowl5* = **new** Bowl(5);

}

运行结果

Bowl(1)

Bowl(2)

Table()

f1(1)

Bowl(4)

Bowl(5)

Bowl(3)

Cupboard()

f1(2)

new Cupboard() in main

Bowl(3)

Cupboard()

f1(2)

new Cupboard() in main

Bowl(3)

Cupboard()

f1(2)

f2(1)

f3(1)

分析结果

主函数里

先运行static即静态函数**static** Table *table* = **new** Table();，转到类Table，先运行静态，即按顺序bowl(1),bowl(2),因为构造方法也可以看作静态，故有table（），f1（1）.

再运行**static** Cupboard *cupboard* = **new** Cupboard();转到类cupboard，先运行静态类bowl（4），bowl(5),非静态类bowl(3),之后构造方法，Cupboard()，f1(2)。

再运行System.***out***.println("new Cupboard() in main");

**new** Cupboard();

System.***out***.println("new Cupboard() in main");

**new** Cupboard();

因为cupboard已经初始化，所以静态对象不再运行，故有结果

new Cupboard() in main

Bowl(3)

Cupboard()

f1(2)

new Cupboard() in main

Bowl(3)

Cupboard()

f1(2)

再运行 *table*.f2(1);

*cupboard*.f3(1);

很明显有结果f2(1)，f3(1)

Ppt21

源程序

**package ceshi;**

**public class InheritStaticInit {**

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

**new T2();**

**}**

**}**

**class T1 {**

**static int s1 = 1;**

**static {**

**System.out.println("static block of T1: " + T2.s2);**

**}**

**T1() {**

**System.out.println("T1(): " + s1);**

**}**

**}**

**class T2 extends T1 {**

**static int s2 = 2;**

**static {**

**System.out.println("static block of T2: " + T2.s2);**

**}**

**T2() {**

**System.out.println("T2(): " + s2);**

**}**

**}**

运行结果

static block of T1: 0

static block of T2: 2

分析过程：初始化**new** T2();，先运行静态，因为要寻至父代

static block of T1: 0

static block of T2: 2

之后运行构造方法

T1(): 1

T2(): 2

，

Ppt31

源程序

**package** ceshi;

**public** **class** BlankFinal {

**private** **final** **int** i = 0; // 被初始化的final

**private** **final** **int** j; // 空白final

**private** **final** Poppet p; // 空白final引用

**public** BlankFinal() {

j = 1; // 初始化空白final

p = **new** Poppet(1); // 初始化空白final引用

}

**public** BlankFinal(**int** x) {

j = x; // 初始化空白final

p = **new** Poppet(x); // 初始化空白final引用

}

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

**new** BlankFinal();

**new** BlankFinal(47);

}

}

**class** Poppet {

**private** **int** i;

Poppet(**int** ii) { i = ii; }

}

运行结果

空

Ppt53

源程序

**(1) interface Printable{**

**void print() {};**

**}**

**不可以**

**(2) abstract interface Printable{**

**void print();**

**}**

**可以**

**(3) abstract interface Printable extends Interface1, Interface2{**

**void print() {};**

**}**

**不可以**

**(4) interface Printable{**

**void print();**

**}**

**可以**

Ppt65

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** FindDups {

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

// 创建一个HashSet对象，缺省的初始容量是16

Set<String> s = **new** HashSet<String>();

// 将命令行中的每个字符串加入到集合s中，

//其中重复的字符串将不能加入，并被打印输出

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

**if** (!s.add(args[i]))

System.***out***.println("Duplicate detected: " + args[i]);

}

// 输出集合s的元素个数以及集合中的所有元素：

System.***out***.println(s.size() + " distinct words detected: " + s);

}

}

运行结果

0 distinct words detected: []

分析，ppt中没有使用**import** java.util.\*;导致一直有bug，后来搜了论坛,才好。Set定义一个string类的集合，

Ppt71

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** UseArrayList {

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

List<String> scores = **new** ArrayList<String>();

scores.add("86"); // 添加元素

scores.add("98"); // 添加元素

scores.add(1, "99"); // 插入元素

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

System.***out***.print(scores.get(i) + " ");// 输出结果

}

scores.set(1, "77"); // 修改第二个元素

scores.remove(0); // 删除第一个元素

System.***out***.println("\n修改并删除之后");

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

System.***out***.print(scores.get(i) + " ");

}

System.***out***.println(" \n按字符串输出\n" + scores.toString());

}

}

运行结果

86 99 98

修改并删除之后

77 98

按字符串输出

[77, 98]

开始，加入86、98、中间在1处插入99，因为数组从零开始，所以有86，99，98.之后修改位标1，即99换成77，删除0位标即86，所以结果为77，98

Ppt75

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** QueueDemo {

**public** **static** **void** printQ(Queue queue) {

**while** (queue.peek() != **null**)

System.***out***.print(queue.remove() + " ");

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

}

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

Queue<Integer> queue = **new** LinkedList<Integer>();

Random rand = **new** Random(47);

**for** (**int** i = 0; i < 10; i++)

queue.offer(rand.nextInt(i + 10));

*printQ*(queue);

Queue<Character> qc = **new** LinkedList<Character>();

**for** (**char** c : "Brontosaurus".toCharArray())

qc.offer(c);

*printQ*(qc);

}

}

运行结果

8 1 1 1 5 14 3 1 0 1

B r o n t o s a u r u s

分析，queue.offer()，不断向queue中加入随机数，最后用remove()获取首元素并清除。同理qc一样的原理，不过是字符。

Ppt80

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** Freq {

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

String[] words = { "if", "it", "is", "to", "be", "it",

"is", "up", "to", "me", "to", "delegate" };

Integer freq;

Map<String, Integer> m = **new** TreeMap<String, Integer>();

**for** (String a : words) {//以(单词,词频)为键值对, 构造频率表

freq = m.get(a); // 获取指定单词的词频。

**if** (freq == **null**) { // 词频递增

freq = **new** Integer(1);

} **else** {

freq = **new** Integer(freq + 1);// .intValue()

}

m.put(a, freq); // 在Map中更改词频

}

System.***out***.println(m.size() + " distinct words detected:");

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

}

}

运行结果

8 distinct words detected:

{be=1, delegate=1, if=1, is=2, it=2, me=1, to=3, up=1}

程序开头写入{ "if", "it", "is", "to", "be", "it", "is", "up", "to", "me", "to", "delegate" };，用for循环读入表中的单词作为键，1为对应值。如果有重复的，修改对应的值+1。所以运行结果为8 distinct words detected:{be=1, delegate=1, if=1, is=2, it=2, me=1, to=3, up=1}

Ppt82

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** UseHashMap {

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

HashMap<String, String> hScore = **new** HashMap<String, String>();

hScore.put("张一", "86");

hScore.put("李二", "98");

hScore.put("海飞", "99");

System.***out***.println("按字符串输出：" + hScore.toString());

hScore.put("李二", "77");

hScore.remove("张一");

System.***out***.println("修改并删除之后");

System.***out***.println("按字符串输出：" + hScore.toString());

}

}

运行结果

按字符串输出：{李二=98, 海飞=99, 张一=86}

修改并删除之后

按字符串输出：{李二=77, 海飞=99}

过程很简单，不做解释

Ppt84

源程序

**package** ceshi;

**import** java.util.\*;

**public** **class** TestIterator {

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

String sentence="I believe I can fly, I believe I can touch the sky.";

String[] strs=sentence.split(" ");

List<String> list=**new** ArrayList<String>( Arrays.*asList*(strs) );

Iterator<String> it=list.iterator();

**while**(it.hasNext())

System.***out***.print(it.next()+"\_");

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

it=list.iterator();

**while**(it.hasNext()){

**if**(it.next().equals("I"))

it.remove();

}

it=list.iterator();

**while**(it.hasNext())

System.***out***.print(it.next()+" ");

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

}

}

运行结果

I\_believe\_I\_can\_fly,\_I\_believe\_I\_can\_touch\_the\_sky.\_

believe can fly, believe can touch the sky.

修改程序输出list为[I, believe, I, can, fly,, I, believe, I, can, touch, the, sky.]，遍及输出有I\_believe\_I\_can\_fly,\_I\_believe\_I\_can\_touch\_the\_sky.\_

之后先把I除去，之后输出it.next()+" "，故有believe can fly, believe can touch the sky.

Ppt95

源程序

**package** ceshi;

**import** java.util.\*;

**enum** Coin { ***PENNY***(1), ***NICKEL***(5), ***DIME***(10), ***QUARTER***(25);

**private** **final** **int** value;

Coin(**int** value) { **this**.value = value; }

**public** **int** value() { **return** value; }

}

**enum** CoinColor { ***COPPER***, ***NICKEL***, ***SILVER*** }

**public** **class** CoinTest {

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

**for** (Coin c : Coin.*values*()) {

System.***out***.print(c + ": " + c.value() + ", ");

**switch** (c) {

**case** ***PENNY***:

System.***out***.println(CoinColor.***COPPER***); **break**;

**case** ***NICKEL***:

System.***out***.println(CoinColor.***NICKEL***); **break**;

**case** ***DIME***:

**case** ***QUARTER***:

System.***out***.println(CoinColor.***SILVER***); **break**;

}

}

}

}

运行结果

PENNY: 1, COPPER

NICKEL: 5, NICKEL

DIME: 10, SILVER

QUARTER: 25, SILVER

Ppt100

源程序

**package** ceshi;

**public** **class** AutoBoxingTest {

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

Integer x, y;

**int** c;

x = 22;// autoboxing

y = 15;// autoboxing

**if** ((c = x.compareTo(y)) == 0)

System.***out***.println("x is equal to y");

**else** **if** (c < 0)

System.***out***.println("x is less than y");

**else**

System.***out***.println("x is greater than y");

System.***out***.println("x + y = " + (x + y));// autounboxing

}

}

运行结果

x is greater than y

x + y = 37

问答题、

1、 字符串变量名（args）属于引用变量，名字代号而已，可以自己取的。

2、总的来说就是个存放字符串数组用的， 去掉就不知道 ＂args＂ 声明的变量是什么类型了。

3、如果有 public static void main 则代表当你RUN这程序曷会RUN这部分,；

4、你在其他位置写的都是为这MAIN而设的。args是arguments的缩写，只是一个默认名，一般都习惯性照写。

5、就是一个字符串数组的变量名，不是关键字args是arguments的缩写默认名，一般都习惯性照写你可以改成其他名。

6、String[] args是[main函数](https://www.baidu.com/s?wd=main%E5%87%BD%E6%95%B0&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)的形式参数,可以用来获取命令行用户输入进去的参数。

7、java 本身不存在不带String args[]的[main函数](https://www.baidu.com/s?wd=main%E5%87%BD%E6%95%B0&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)，java程序中去掉String args[]会出现错误。

记事本

