赛题库

赛项编号：GZ-2018106

赛项名称：软件测试

（一） 功能测试

1、概述

本部分要求完成资产管理系统的Web端功能测试、Web端界面测试和Web端兼容性测试、手机APP功能测试。完成要求的相关测试文档。Web端要求使用Chrome浏览器作为测试工具。

2、赛题

1) 登录功能测试

登录功能描述：资产管理员、超级管理员需要通过登录进入 Web 端资产管理系统，登录是进入该系统的唯一入口。首先选择角色，再输入有效的用户名、密码、任务ID和验证码，才能登录该系统。

2) 首页功能测试

资产管理员/超级管理员登录后，默认进入首页欢迎页面。

3) 个人信息功能测试

登录系统后，资产管理员/超级管理员可以查看个人信息，姓名、手机号、工号等。可修改手机号、修改登录密码和退出系统。

4) 资产类别功能测试

“资产类别”作为资产信息的属性而存在，超级管理员可以对资产类别进行管理：包括资产类别的新增、修改、启用和禁用；资产管理员没有操作权限，只能进行资产类别的查看。

5) 品牌功能测试

“品牌”作为资产信息的属性而存在，超级管理员可以对品牌进行管理：包括品牌的新增、修改、启用和禁用；资产管理员没有操作权限，只能进行品牌的查看。

6) 取得方式功能测试

“取得方式”作为资产信息的属性而存在，超级管理员可以对取得方式进行管理：包括取得方式的新增、修改、启用和禁用；资产管理员没有操作权限，只能进行取得方式的查看。

7) 供应商功能测试

“供应商”作为资产信息的属性而存在，超级管理员可以新增、修改、启用、禁用、查询、查看供应商信息。资产管理员可以查询、查看供应商信息。

8) 存放地点功能测试

“存放地点”作为资产信息的属性而存在，超级管理员可以新增、修改、启用、禁用、查询、查看存放地点详情。资产管理员可以查询、查看存放地点详情。

9) 部门管理功能测试

该模块用于资产管理员对组织机构信息进行管理，资产管理员可以新增、修改部门信息。该模块超级管理员不可见。

10) 人员管理功能测试

该模块用于资产管理员对人员信息进行管理，资产管理员可以新增、修改、删除员工信息。该模块超级管理员不可见。

11) 资产入库功能测试

该模块用于资产管理员对资产的入库过程进行管理，资产管理员可以进行资

产入库登记、修改、查询、导出资产信息。该模块超级管理员不可见。

12) 资产借还功能测试

该模块用于资产管理员对资产的借还过程进行管理，资产管理员可以进行资产借用登记、归还、查询以及查看资产借还信息。该模块超级管理员不可见。

13) 资产转移功能测试

资产管理员可进行资产转移，转移给新的使用人。该模块用于资产管理员对资产的转移过程进行管理，由资产管理员记录资产转移的流水。该模块超级管理员不可见。

14) 资产维修功能测试

该模块用于资产管理员对资产的维修过程进行管理，资产管理员可以进行资产维修登记、维修统计、查询以及查看资产维修信息。该模块超级管理员不可见。

15) 资产报废功能测试

该模块用于资产管理员对资产的报废过程进行管理，资产管理员可以进行资产报废登记、查询以及查看资产报废信息。该模块超级管理员不可见。

16) 资产盘点功能测试

该模块用于资产管理员对资产的盘点过程进行管理，资产管理员可以进行新增盘点单、启动盘点、录入/修改盘点结果、结束盘点以及查询、查看盘点结果等操作。该模块超级管理员不可见。

17) 资产申购功能测试

该模块用于资产管理员对资产的申购过程进行管理，资产管理员可以进行资产申购登记、查询以及查看资产申购信息。该模块超级管理员不可见。

18) 统计报表功能测试

由资产管理员对现有资产进行各维度的统计，资产管理员可以根据资产状态、资产类别、供应商、品牌、取得方式、存放地点各指标统计现有资产，并生成相应的图表。

19) App 登录功能测试

资产 APP 在手机安装完毕后，点击图标，启动该程序；进入登录页面，输入有效的任务 ID、用户名、密码进行登录。

20) APP 我的功能测试

用于展示用户的相关信息，并完成退出系统操作。

21) APP 资产功能测试

用于资产管理员对资产的入库过程进行管理。

22) APP 盘点功能测试

用于资产管理员对资产的盘点过程进行管理。

23) APP 报表功能测试

资产管理员可以根据资产状态、资产类别、供应商、品牌、取得方式、存放地点各指标统计现有资产，并生成相应的图表。

(二) 性能测试

1. 概述

使用 LoadRunner12.55_Community_Edition 英文版执行性能测试，录制脚本、配置参数、调试脚本、脚本回放；设置场景，执行性能测试并且截图完成性能测试总结报告。

性能测试包括以下步骤：

（1）性能测试压力点选取。

（2）脚本录制、调试和回放。

（3）场景参数配置和执行测试。

（4）测试结果数据分析，并截图。

（5）性能测试报告编写。

2. 性能测试赛题

（1）步骤一：性能测试压力点选取

题 1：用户登录

题 2：用户退出

题 3：个人信息

题 4：资产类别

题 5：品牌

题 6：取得方式

题 7：供应商

题 8：存放地点

题 9：部门管理

题 10：人员管理

题 11：资产入库

题 12：资产借还

题 13：资产转移

题 14：资产维修

题 15：资产报废

题 16：资产盘点

题 17：资产申购

题 18：统计报表

（2）步骤二：脚本录制、调试和回放。迭代次数设置、回放参数值配置，输出日志使用、参数化配置、集合点、事务、检查点设置等。

参数类型可选择：

题 1：Date/Time

题 2：File

题 3：Group Name

题 4：Iteration Number

题 5：Load Generator Name

题 6：Random Number

题 7：Table、Unique Number

题 8：User Defined Function

题 9：Vuser ID

题 10：XML

数据分配方法可选择：

题 1：Sequential

题 2：Random

题 3：Unique

数据更新方式可选择：

题 1：Each occurrence

题 2：Each iteration

题 3：Once

（3）步骤三：场景配置并且执行场景。

场景配置虚拟用户数可设置：

题 1：5

题 2：10

题 3：15

题 4：20

题 5：25

题 6：30

题 7：35

题 8：40

题 9：45

题 10：50

持续时长可设置：

题 1：5min

题 2：10min

题 3：15min

题 4：20min

题 5：所有 vuser 运行完成

递增虚拟用户数可设置：

题 1：5

题 2：10

题 3：15

题 4：20

递增时长设置：

题 1：10s

题 2：15s

题 3：20s

题 4：25s

题 5：30s

递减虚拟用户数可设置：

题 1：5

题 2：10

题 3：15

题 4：20

题 5：立刻停止

递减时长可设置：

题 1：10s

题 2：15s

题 3：20s

题 4：25s

题 5：30s。

（4）步骤四：测试结果数据截图。

题 1：Summary Report

题 2：Running Vusers

题 3：Hit per Second

题 4：Throughput

题 5：Transaction Summary

题 6：Average Transaction Response Time

题 7：Vuser Summary

题 8：Rendezvous

题 9：Error Statistics (by Description)

题 10：Errors per Second (by Description)

题 11：Transactions per Second

题 12：Total Transactions per Second

题 13：Transaction Performance Summary

题 14：Transaction Response Time Under Load

题 15：Transaction Response Time (Percentile)

题 16：Transaction Response Time (Distribution)

题 17：Transaction Response Time By Location

题 18：Web Page Diagnostics、Page Component Breakdown

题 19：Page Download Time Breakdown

题 20：Time to First Buffer Breakdown

题 21：Downloaded Component Size (KB)

题 22：HTTP Status Code Summary

题 23：HTTP Responses per Second。

（5）步骤五：性能测试报告编写。

根据性能测试情况，参考性能测试总结报告模版，按要求截取性能测试过程和结果截图并粘贴到性能测试报告，完成性能测试总结报告的编写。

（三） 白盒测试

1.概述

按照白盒测试的需求和软件代码，进行 JAVA 代码走查。

2.白盒测试赛题

请阅读分析 JAVA 代码并根据代码逻辑写出执行本段代码的输出结果。说明：代码编译执行的 JDK 版本为 JDK 1.6。如果题目代码执行结果是编译不能通过，请注明编译失败的行数；如果题目代码错误是在运行中抛出异常，请注明抛出异常的行数及抛出异常的类型。题目中行数仅为标示，不是代码的组成部分。

题目 1. 以下这段代码：

```
1. enum Animals {
2. DOG("woof"), CAT("meow"), FISH("burble");
3. String sound;
4. Animals(String s) { sound = s; }
5. }
6. class TestEnum {
7. static Animals a;
8. public static void main(String [] args) {
9. System.out.println(a.DOG.sound + " " + a.FISH.sound);
10. }
11. }
```

执行结果是？

题目 2. 以下这段代码：

```
1. package pkgA;
2. public class Foo {
```

```
3. int a = 5;
4. protected int b = 6;
5. }
6. package pkgB;
7. import pkgA.*;
8. public class Fiz extends Foo {
9. public static void main(String[] args) {
10. Foo f = new Foo();
11. System.out.print(" " + f.a);
12. System.out.print(" " + f.b);
13. System.out.print(" " + new Fiz().a);
14. System.out.println(" " + new Fiz().b);
15. }
16. }
```

执行结果是?

题目 3. 以下这段代码:

```
4. class Announce {
5. public static void main(String[] args) {
6. for(int __x = 0; __x < 3; __x++) ;
7. int #lb = 7;
8. long [] x [5];
9. Boolean []ba[];
10. enum Traffic { RED, YELLOW, GREEN };
11. }
```

```
12. }
```

执行结果是?

题目 4. 以下这段代码:

```
4. public class Frodo extends Hobbit {
5. public static void main(String[] args) {
6. Short myGold = 7;
7. System.out.println(countGold(myGold, 6));
8. }
9. }
10. class Hobbit {
11. int countGold(int x, int y) { return x + y; }
12. }
```

执行结果是?

题目 5. 以下这段代码:

```
1. class Top {
2. public Top(String s) { System.out.print("B"); }
3. }
4. public class Bottom2 extends Top {
5. public Bottom2(String s) { System.out.print("D"); }
6. public static void main(String [] args) {
```

7. new Bottom2("C");

8. System.out.println(" ");

9. } }

执行结果是?

题目 6. 以下这段代码:

3. class Clidder {

4. private final void flipper() { System.out.println("Clidder"); }

5. }

6. public class Clidlet extends Clidder {

7. public final void flipper() { System.out.println("Clidlet"); }

8. public static void main(String [] args) {

9. new Clidlet().flipper();

10. } }

执行结果是?

题目 7. 以下这段代码:

3. class Dog {

4. public void bark() { System.out.print("woof "); }

5. }

6. class Hound extends Dog {

7. public void sniff() { System.out.print("sniff "); }

```
8. public void bark() { System.out.print("howl "); }
9. }
10. public class DogShow {
11. public static void main(String[] args) { new DogShow().go(); }
12. void go() {
13. new Hound().bark();
14. ((Dog) new Hound()).bark();
15. ((Dog) new Hound()).sniff();
16. }
17. }
```

执行结果是?

题目 8. 以下这段代码:

```
3. public class Redwood extends Tree {
4. public static void main(String[] args) {
5. new Redwood().go();
6. }
7. void go() {
8. go2(new Tree(), new Redwood());
9. go2((Redwood) new Tree(), new Redwood());
10. }
11. void go2(Tree t1, Redwood r1) {
12. Redwood r2 = (Redwood)t1;
13. Tree t2 = (Tree)r1;
14. }
```

15. }

16. class Tree { }

执行结果是?

题目 9. 以下这段代码：

3. public class Tenor extends Singer {

4. public static String sing() { return "fa"; }

5. public static void main(String[] args) {

6. Tenor t = new Tenor();

7. Singer s = new Tenor();

8. System.out.println(t.sing() + " " + s.sing());

9. }

10. }

11. class Singer { public static String sing() { return "la"; } }

执行结果是?

题目 10. 以下这段代码：

3. class Alpha {

4. static String s = " ";

5. protected Alpha() { s += "alpha "; }

6. }

7. class SubAlpha extends Alpha {

```
8. private SubAlpha() { s += "sub "; }
9. }
10. public class SubSubAlpha extends Alpha {
11. private SubSubAlpha() { s += "subsub "; }
12. public static void main(String[] args) {
13. new SubSubAlpha();
14. System.out.println(s);
15. }
16. }
```

执行结果是?

题目 11. 以下这段代码:

```
3. class Building {
4. Building() { System.out.print("b "); }
5. Building(String name) {
6. this(); System.out.print("bn " + name);
7. }
8. }
9. public class House extends Building {
10. House() { System.out.print("h "); }
11. House(String name) {
12. this(); System.out.print("hn " + name);
13. }
14. public static void main(String[] args) { new House("x "); }
15. }
```

执行结果是？

题目 12. 以下这段代码：

```
3. class Mammal {
4. String name = "furry ";
5. String makeNoise() { return "generic noise"; }
6. }
7. class Zebra extends Mammal {
8. String name = "stripes ";
9. String makeNoise() { return "bray"; }
10. }
11. public class ZooKeeper {
12. public static void main(String[] args) { new ZooKeeper().go(); }
13. void go() {
14. Mammal m = new Zebra();
15. System.out.println(m.name + m.makeNoise());
16. }
17. }
```

执行结果是？

题目 13. 以下这段代码：

```
3. class A { }
```

```
4. class B extends A { }
5. public class ComingThru {
6. static String s = "-";
7. public static void main(String[] args) {
8. A[] aa = new A[2];
9. B[] ba = new B[2];
10. sifter(aa);
11. sifter(ba);
12. sifter(7);
13. System.out.println(s);
14. }
15. static void sifter(A[]... a2) { s += "1"; }
16. static void sifter(B[]... b1) { s += "2"; }
17. static void sifter(B[] b1) { s += "3"; }
18. static void sifter(Object o) { s += "4"; }
19. }
```

执行结果是？

题目 14. 以下这段代码：

```
1. class Alien {
2. String invade(short ships) { return "a few"; }
3. String invade(short... ships) { return "many"; }
4. }
5. class Defender {
6. public static void main(String [] args) {
```

```
7. System.out.println(new Alien().invade(7));
8. } }
```

执行结果是?

题目 15. 以下这段代码:

```
1. class Dims {
2. public static void main(String[] args) {
3. int[][] a = {{1,2,}, {3,4}};
4. int[] b = (int[]) a[1];
5. Object o1 = a;
6. int[][] a2 = (int[][]) o1;
7. int[] b2 = (int[]) o1;
8. System.out.println(b[1]);
9. } }
```

执行结果是?

题目 16. 以下这段代码:

```
1. class Mixer {
2. Mixer() { }
3. Mixer(Mixer m) { m1 = m; }
4. Mixer m1;
5. public static void main(String[] args) {
```

```
6.  Mixer m2 = new Mixer();
7.  Mixer m3 = new Mixer(m2); m3.go();
8.  Mixer m4 = m3.m1; m4.go();
9.  Mixer m5 = m2.m1; m5.go();
10. }
11. void go() { System.out.print("hi "); }
12. }
```

执行结果是？

题目 17. 以下这段代码：

```
1.  class Fizz {
2.  int x = 5;
3.  public static void main(String[] args) {
4.  final Fizz f1 = new Fizz();
5.  Fizz f2 = new Fizz();
6.  Fizz f3 = FizzSwitch(f1,f2);
7.  System.out.println((f1 == f3) + " " + (f1.x == f3.x));
8.  }
9.  static Fizz FizzSwitch(Fizz x, Fizz y) {
10. final Fizz z = x;
11. z.x = 6;
12. return z;
13. } }
```

执行结果是？

题目 18. 以下这段代码：

```
1.  class Bird {
2.  { System.out.print("b1 "); }
3.  public Bird() { System.out.print("b2 "); }
4.  }
5.  class Raptor extends Bird {
6.  static { System.out.print("r1 "); }
7.  public Raptor() { System.out.print("r2 "); }
8.  { System.out.print("r3 "); }
9.  static { System.out.print("r4 "); }
10. }
11. class Hawk extends Raptor {
12. public static void main(String[] args) {
13. System.out.print("pre ");
14. new Hawk();
15. System.out.println("hawk ");
16. }
17. }
```

执行结果是？

题目 19. 以下这段代码：

```
3. public class Ouch {
4. static int ouch = 7;
```

```
5. public static void main(String[] args) {
6. new Ouch().go(ouch);
7. System.out.print(" " + ouch);
8. }
9. void go(int ouch) {
10. ouch++;
11. for(int ouch = 3; ouch < 6; ouch++)
12. ;
13. System.out.print(" " + ouch);
14. }
15. }
```

执行结果是?

题目 20. 以下这段代码:

```
3. public class Bertha {
4. static String s = "";
5. public static void main(String[] args) {
6. int x = 4; Boolean y = true; short[] sa = {1,2,3};
7. doStuff(x, y);
8. doStuff(x);
9. doStuff(sa, sa);
10. System.out.println(s);
11. }
12. static void doStuff(Object o) { s += "1"; }
13. static void doStuff(Object... o) { s += "2"; }
```

```
14. static void doStuff(Integer... i) { s += "3"; }
15. static void doStuff(Long L) { s += "4"; }
16. }
```

执行结果是?

题目 21. 以下这段代码:

```
3. class Box {
4. int size;
5. Box(int s) { size = s; }
6. }
7. public class Laser {
8. public static void main(String[] args) {
9. Box b1 = new Box(5);
10. Box[] ba = go(b1, new Box(6));
11. ba[0] = b1;
12. for(Box b : ba) System.out.print(b.size + " ");
13. }
14. static Box[] go(Box b1, Box b2) {
15. b1.size = 4;
16. Box[] ma = {b2, b1};
17. return ma;
18. }
19. }
```

执行结果是?

题目 22. 以下这段代码：

```
3. public class Dark {
4. int x = 3;
5. public static void main(String[] args) {
6. new Dark().go1();
7. }
8. void go1() {
9. int x;
10. go2(++x);
11. }
12. void go2(int y) {
13. int x = ++y;
14. System.out.println(x);
15. }
16. }
```

执行结果是？

题目 23. 以下这段代码：

```
1. class Hexy {
2. public static void main(String[] args) {
3. Integer i = 42;
4. String s = (i<40)?"life":(i>50)?"universe":"everything";
5. System.out.println(s);
```

```
6. }
7. }
```

执行结果是?

题目 24. 以下这段代码：

```
1.  class Feline {
2.  public static void main(String[] args) {
3.  Long x = 42L;
4.  Long y = 44L;
5.  System.out.print(" " + 7 + 2 + " ");
6.  System.out.print(foo() + x + 5 + " ");
7.  System.out.println(x + y + foo());
8.  }
9.  static String foo() { return "foo"; }
10. }
```

执行结果是?

题目 25. 以下这段代码：

```
3. public class Twisty {
4. { index = 1; }
5. int index;
6. public static void main(String[] args) {
```

```
7. new Twisty().go();
8. }
9. void go() {
10. int [][] dd = {{9,8,7}, {6,5,4}, {3,2,1,0}};
11. System.out.println(dd[index++][index++]);
12. }
13. }
```

执行结果是?

题目 26. 以下这段代码:

```
3. public class McGee {
4. public static void main(String[] args) {
5. Days d1 = Days.TH;
6. Days d2 = Days.M;
7. for(Days d: Days.values()) {
8. if(d.equals(Days.F)) break;
9. d2 = d;
10. }
11. System.out.println((d1 == d2)?"same old" : "newly new");
12. }
13. enum Days {M, T, W, TH, F, SA, SU};
14. }
```

执行结果是?

题目 27. 以下这段代码：

```
3. interface Vessel { }
4. interface Toy { }
5. class Boat implements Vessel { }
6. class Speedboat extends Boat implements Toy { }
7. public class Tree {
8. public static void main(String[] args) {
9. String s = "0";
10. Boat b = new Boat();
11. Boat b2 = new Speedboat();
12. Speedboat s2 = new Speedboat();
13. if((b instanceof Vessel) && (b2 instanceof Toy)) s += "1";
14. if((s2 instanceof Vessel) && (s2 instanceof Toy)) s += "2";
15. System.out.println(s);
16. }
17. }
```

执行结果是？

题目 28. 以下这段代码：

```
1.  class Plane {
2.  static String s = "-";
3.  public static void main(String[] args) {
4.  new Plane().s1();
5.  System.out.println(s);
```

```
6.  }
7.  void s1() {
8.  try { s2(); }
9.  catch (Exception e) { s += "c"; }
10. }
11. void s2() throws Exception {
12. s3(); s += "2";
13. s3(); s += "2b";
14. }
15. void s3() throws Exception {
16. throw new Exception();
17. } }
```

执行结果是?

题目 29. 以下这段代码:

```
1.  class Emu {
2.  static String s = "-";
3.  public static void main(String[] args) {
4.  try {
5.  throw new Exception();
6.  } catch (Exception e) {
7.  try {
8.  try { throw new Exception();
9.  } catch (Exception ex) { s += "ic "; }
10. throw new Exception(); }
```

```
11. catch (Exception x) { s += "mc "; }
12. finally { s += "mf "; }
13. } finally { s += "of "; }
14. System.out.println(s);
15. } }
```

执行结果是?

题目 30. 以下这段代码:

```
3. class SubException extends Exception { }
4. class SubSubException extends SubException { }
5.
6. public class CC { void doStuff() throws SubException { } }
7.
8. class CC2 extends CC { void doStuff() throws SubSubException { } }
9.
10. class CC3 extends CC { void doStuff() throws Exception { } }
11.
12. class CC4 extends CC { void doStuff(int x) throws Exception { } }
13.
14. class CC5 extends CC { void doStuff() { } }
```

执行结果是?

题目 31. 以下这段代码：

```
3. public class Ebb {
4. static int x = 7;
5. public static void main(String[] args) {
6. String s = "";
7. for(int y = 0; y < 3; y++) {
8. x++;
9. switch(x) {
10. case 8: s += "8 ";
11. case 9: s += "9 ";
12. case 10: { s+= "10 "; break; }
13. default: s += "d ";
14. case 13: s+= "13 ";
15. }
16. }
17. System.out.println(s);
18. }
19. static { x++; }
20. }
```

执行结果是？

题目 32. 以下这段代码：

```
3. public class Circles {
4. public static void main(String[] args) {
5. int[] ia = {1,3,5,7,9};
```

```
6. for(int x : ia) {
7. for(int j = 0; j < 3; j++) {
8. if(x > 4 && x < 8) continue;
9. System.out.print(" " + x);
10. if(j == 1) break;
11. continue;
12. }
13. continue;
14. }
15. }
16. }
```

执行结果是?

题目 33. 以下这段代码:

```
3. public class OverAndOver {
4. static String s = "";
5. public static void main(String[] args) {
6. try {
7. s += "1";
8. throw new Exception();
9. } catch (Exception e) { s += "2";
10. } finally { s += "3"; doStuff(); s += "4";
11. }
12. System.out.println(s);
13. }
```

```
14. static void doStuff() { int x = 0; int y = 7/x; }
15. }
```

执行结果是?

题目 34. 以下这段代码:

```
3. public class Wind {
4. public static void main(String[] args) {
5. foreach:
6. for(int j=0; j<5; j++) {
7. for(int k=0; k< 3; k++) {
8. System.out.print(" " + j);
9. if(j==3 && k==1) break foreach;
10. if(j==0 || j==2) break;
11. }
12. }
13. }
14. }
```

执行结果是?

题目 35. 以下这段代码:

```
1. import java.io.*;
2. class Player {
```

```
3.  Player() { System.out.print("p"); }
4.  }
5.  class CardPlayer extends Player implements Serializable {
6.  CardPlayer() { System.out.print("c"); }
7.  public static void main(String[] args) {
8.  CardPlayer c1 = new CardPlayer();
9.  try {
10. FileOutputStream fos = new FileOutputStream("play.txt");
11. ObjectOutputStream os = new ObjectOutputStream(fos);
12. os.writeObject(c1);
13. os.close();
14. FileInputStream fis = new FileInputStream("play.txt");
15. ObjectInputStream is = new ObjectInputStream(fis);
16. CardPlayer c2 = (CardPlayer) is.readObject();
17. is.close();
18. } catch (Exception x ) { }
19. }
20. }
```

执行结果是?

题目 36. 以下这段代码:

```
1.  import java.util.*;
2.  class Sun {
3.  public static void before() {
4.  Set set = new TreeSet();
```

```
5.  set.add("2");
6.  set.add(3);
7.  set.add("1");
8.  Iterator it = set.iterator();
9.  while (it.hasNext())
10. System.out.print(it.next() + " ");
11. }
12. public static void main(String[] args) { before(); }
13. }
```

执行结果是?

题目 37. 以下这段代码:

```
10. import java.util.*;
11. class Cat {
12. public static void main(String[] args) {
13. TreeSet<String> s = new TreeSet<String>();
14. TreeSet<String> subs = new TreeSet<String>();
15. s.add("a"); s.add("b"); s.add("c"); s.add("d"); s.add("e");
16.
17. subs = (TreeSet)s.subSet("b", true, "d", true);
18. s.add("g");
19. s.pollFirst();
20. s.pollFirst();
21. s.add("c2");
22. System.out.println(s.size() +" "+ subs.size());
```

23. } }

执行结果是?

题目 38. 以下这段代码:

3. import java.util.*;

4. class Dog { int size; Dog(int s) { size = s; } }

5. public class FirstGrade {

6. public static void main(String[] args) {

7. TreeSet<Integer> i = new TreeSet<Integer>();

8. TreeSet<Dog> d = new TreeSet<Dog>();

9.

10. d.add(new Dog(1)); d.add(new Dog(2)); d.add(new Dog(1));

11. i.add(1); i.add(2); i.add(1);

12. System.out.println(d.size() + " " + i.size());

13. }

14. }

执行结果是?

题目 39. 以下这段代码:

1. public class TestObj {

2. public static void main(String[] args) {

3. Object o = new Object() {

```
4. public boolean equals(Object obj) {
5. return true;
6. }
7. }
8. System.out.println(o.equals("Fred"));
9. }
10. }
```

执行结果是?

题目 40. 以下这段代码:

```
1. public class HorseTest {
2. public static void main(String[] args) {
3. class Horse {
4. public String name;
5. public Horse(String s) {
6. name = s;
7. }
8. }
9. Object obj = new Horse("Zippo");
10. System.out.println(obj.name);
11. }
12. }
```

执行结果是?

题目 41. 以下这段代码：

```
1.  public abstract class AbstractTest {
2.  public int getNum() {
3.  return 45;
4.  }
5.  public abstract class Bar {
6.  public int getNum() {
7.  return 38;
8.  }
9.  }
10. public static void main(String[] args) {
11. AbstractTest t = new AbstractTest() {
12. public int getNum() {
13. return 22;
14. }
15. };
16. AbstractTest.Bar f = t.new Bar() {
17. public int getNum() {
18. return 57;
19. }
20. };
21. System.out.println(f.getNum() + " " + t.getNum());
22. } }
```

执行结果是？

题目 42. 以下这段代码：

```
5. class A { void m() { System.out.println("outer"); } }
6.
7. public class TestInners {
8. public static void main(String[] args) {
9. new TestInners().go();
10. }
11. void go() {
12. new A().m();
13. class A { void m() { System.out.println("inner"); } }
14. }
15. class A { void m() { System.out.println("middle"); } }
16. }
```

执行结果是？

题目 43. 以下这段代码：

```
3. public class City {
4. class Manhattan {
5. void doStuff() throws Exception { System.out.print("x "); }
6. }
7. class TimesSquare extends Manhattan {
8. void doStuff() throws Exception { }
9. }
10. public static void main(String[] args) throws Exception {
```

```
11. new City().go();
12. }
13. void go() throws Exception { new TimesSquare().doStuff(); }
14. }
```

执行结果是？

题目 44. 以下这段代码：

```
3. public class Navel {
4. private int size = 7;
5. private static int length = 3;
6. public static void main(String[] args) {
7. new Navel().go();
8. }
9. void go() {
10. int size = 5;
11. System.out.println(new Gazer().adder());
12. }
13. class Gazer {
14. int adder() { return size * length; }
15. }
16. }
```

执行结果是？

题目 45. 以下这段代码：

```
3. import java.util.*;
4. public class Pockets {
5. public static void main(String[] args) {
6. String[] sa = {"nickel", "button", "key", "lint"};
7. Sorter s = new Sorter();
8. for(String s2: sa) System.out.print(s2 + " ");
9. Arrays.sort(sa,s);
10. System.out.println();
11. for(String s2: sa) System.out.print(s2 + " ");
12. }
13. class Sorter implements Comparator<String> {
14. public int compare(String a, String b) {
15. return b.compareTo(a);
16. }
17. }
18. }
```

执行结果是？

题目 46. 以下这段代码：

```
3. class MyThread extends Thread {
4. public static void main(String [] args) {
5. MyThread t = new MyThread();
6. Thread x = new Thread(t);
7. x.start();
```

```
8. }
9. public void run() {
10. for(int i=0;i<3;++i) {
11. System.out.print(i + "..");
12. } } }
```

执行结果是?

题目 47. 以下这段代码:

```
1. public class WaitTest {
2. public static void main(String [] args) {
3. System.out.print("1 ");
4. synchronized(args){
5. System.out.print("2 ");
6. try {
7. args.wait();
8. }
9. catch(InterruptedException e){}
10. }
11. System.out.print("3 ");
12. } }
```

执行结果是?

题目 48. 以下这段代码：

```
1. public class WaitTest {
2. public static synchronized void main(String[] args) throws
3. InterruptedException {
4. Thread t = new Thread();
5. t.start();
6. System.out.print("X");
7. t.wait(10000);
8. System.out.print("Y");
9. } }
```

执行结果是？

题目 49. 以下这段代码：

```
1.  class MyThread extends Thread {
2.  MyThread() {
3.  System.out.print(" MyThread");
4.  }
5.  public void run() { System.out.print(" bar"); }
6.  public void run(String s) { System.out.print(" baz"); }
7.  }
8.  public class TestThreads {
9.  public static void main (String [] args) {
10. Thread t = new MyThread() {
11. public void run() { System.out.print(" foo"); }
12. };
```

13. t.start();

14. } }

执行结果是?

题目 50. 以下这段代码:

3. public class Leader implements Runnable {

4. public static void main(String[] args) {

5. Thread t = new Thread(new Leader());

6. t.start();

7. System.out.print("m1 ");

8. t.join();

9. System.out.print("m2 ");

10. }

11. public void run() {

12. System.out.print("r1 ");

13. System.out.print("r2 ");

14. }

15. }

执行结果是?

题目 51. 以下这段代码:

3. class Chicks {

4. synchronized void yack(long id) {

```
5. for(int x = 1; x < 3; x++) {
6. System.out.print(id + " ");
7. Thread.yield();
8. }
9. }
10. }
11. public class ChicksYack implements Runnable {
12. Chicks c;
13. public static void main(String[] args) {
14. new ChicksYack().go();
15. }
16. void go() {
17. c = new Chicks();
18. new Thread(new ChicksYack()).start();
19. new Thread(new ChicksYack()).start();
20. }
21. public void run() {
22. c.yack(Thread.currentThread().getId());
23. }
24. }
```

执行结果是?
