2021 年 Java 练习题

一、选择题

1、关于 Java 语言叙述错误的是：（ C ）

A．Java 语言具有跨平台性　　B．Java 是一种面向对象的语言

C．Java 类可以多继承　　D．Java 程序具有可移植性

2、Java 应用程序中 main 方法的返回类型是什么？（ D ）

A、int　　B、float　　C、double　　D、void

3、关于 main()方法的说法中，正确的是(C)

A、main()方法只能放在公共类中　　B、main()的头定义可以根据情况任意更改

C、一个类中可以没有 main()方法　　D、所有对象的创建都必须放在 main()方法

4、Java 源程序扩展名是（ A ）。

A. .java　　B. .class　　C. .obj　　D. .exe

5、Java 的编译命令是(B)

A、javap　　B、javac　　C、jdb　　D、java

6、Java 编译器将 Java 程序转换为字节码并保存在（ B ）文件中。

A. .java　　B. .class　　C. .obj　　D. .exe

7、运行 Java 程序的命令是(D)

A、jvm　　B、javac　　C、jre　　D、java

8、为了在命令行方式下编译和运行 Java 程序，需要建立 Windows 环境变量(C)，其值为 JDK 安装目录下的 bin 路径。

A、Java_Home　　B、Marven_Home　　C、Path　　D、ClassPath

9、下列表示 Java 运行时环境的是(C)

A、JDK　　B、SDK　　C、JRE　　D、JVM

10、(D)可以理解成一个以字节码为机器指令的 CPU。

A、JDK　　B、SDK　　C、JRE　　D、JVM

11、下面哪个不是 java 语言中的关键字？（ B ）

A．long　　B．sizeof　　C．instanceof　　D．const

12、以下哪项标识符是不合法的。(D)

A．super　　B．$1　　C．fieldname　　D．3_a

13、0.99 的数据类型是(B)

A、float　　B、double　　C、float　　D、double

14、关于 Java 中数据类型叙述正确的是（ B ）

A、整型数据在不同平台下长度不同　　B．boolean 类型数据只有 2 个值，true 和 false

C．数组属于简单数据类型　　D．String 属于 Java 基本数据类型

15、在 Java 语言中，以下哪个数组声明是不正确的(D)

A、int[ ] s1, s2;　　B、int[ ] a = new int[5];　　C、int n=5; int[ ] a = new int[n] ;　　D、int a[5];

16、以下二维数组声明合法的是（ C ）

A．int[2][3] arr = new int[ ][ ];　　B．in t arr[ ][ ] = new int[2, 3];

C．int[ ][ ] arr = new int[2][3];　　D．int arr[2][3];

17、下列数据类型中，占用字节数最小的是(C)

A、short　　B、int　　C、byte　　D、long

18、long 类型占用字节数是(D)

A、1　　B、2　　C、4　　D、8

19、已知 int i= Integer.MIN_VALUE;则(i-1)+(i-1L)= （ C ）

A. 0　　B. 2　　C. -2　　D. -1

20、下面程序段执行后 b 的值是(B)。

```
Integer a =new Integer(9);
boolean b = a  instanceof  Object;
```

A. 9　　B.true　　C.1　　D.false

21、在命令行键入：java Test　aaa　bb　c　　回车后输出的结果是（A　）。

```
public class Test {
    public static void main(String args[]) {
        int　k1=args.length;
        int　k2=args[1].length();
        System.out.print(k1+"　"+k2);
    }
}
```

A. 3　2　　B. 1　2　　C. 1　3　　D. 3　3

22、A 类中定义如下方法：

```
public static void change(String str){　　str+="xyz";　　}
```

则以下代码块输出结果是：（　A　）

```
String str="abc";　　A.change(str);　　System.out.println(str);
```

A. abc　　B. xyz　　C. abcxyz　　D. xyzabc

23、定义 String s="hello world";　以下操作不合法的是（　　B　　）

A、int i=s.length();　　B、 s>>>=2;　　C、String ts=s.trim();　　D、char ch=s.charAt(0);

24、String 类的 indexOf()方法如未能找到所指定的子字符串，那么其返回值为（　　B　　）

A、false　　B、-1　　C、0　　D、true

25、String 类提供的下列方法中，返回值不是 String 类型的是(　　D　　)

A、toLowerCase()　　B、substring()　　C、concat()　　D、charAt()

26、当用 private 修饰的成员变量时，以下选项正确的是（　　C　　）

A．可以被三种类访问：该类自身、与它在同一包中的其他类、在其他包中的该类的子类

B．可以被两种类访问：该类本身、该类的所有子类

C．只能被该类自身所访问和修改

D．只能被在同一个包中的类访问

27、关于继承的说法，正确的是(　　B　　)

A、子类将继承父类所有的属性和方法　　B、子类将继承父类的非私有属性和方法

C、子类只继承父类 public 的方法和属性　　D、子类只继承父类的方法，而不继承属性

28、在 Java 中，一个类可同时定义许多同名的方法，这些方法的形式参数个数、类型或顺序各不相同。这种面向对象程序的特性称为（　　C　）。

A、隐藏　　B、覆盖　　C、重载　　D、继承

29、覆盖与重载的关系是(　　A　　)

A、覆盖发生在父类与子类之间，而重载发生在同一个类中

B、覆盖方法可以不同名，而重载方法必须同名

C、final 修饰的方法可以被覆盖，但不能被重载

D、覆盖与重载是相同的

30、方法与覆盖方法的(　　A　　)必须相同

A、名称、参数列表、返回类型　　B、名称、参数列表、抛出异常

C、名称、参数列表、访问权限　　D、名称、返回类型、抛出异常

31、以下 Java 描述正确的是（　　D　　）

A、一个包中不能包含多个类　　B、一个源文件中，可以有多个 public 类

C、同一个包的类不可以相互访问　　D、系统会为源文件创建默认的包

32、下列软件包中，不需要使用 import 指令就可直接使用的是(　　A　　)

A、java.lang　　B、java.text　　C、java.sql　　D、java.util

33、Java 集合类来自（　D　　）包。

A、java.lang　　B、java.io　　C、java.net　　D、java.util

34、下列(　　C　　)不属于 java.net 包。

A、Socket　　B、ServerSocket　　C、InputStream　　D、DatagramPacket

35、Java 的访问修饰符具有不同的访问权限，按从小到大排序正确是（ C ）
A. private<protected<default<public B. private<protected<public<default
C. private<default<protected<public D. default<private<protected<public
36、以下关于构造函数的描述正确的是（ D ）。
A．构造函数是特殊的方法，返回类型只能为 void
B．可以定义多个构造函数，但每个方法名需各不相同
C．在类定义中必须定义构造函数，而且只能定义一个构造函数
D．构造方法与类同名，在创建新对象时系统会自动调用构造函数
37、如下所示的 test 类中，共有（ C ）个构造方法。

```
public class test{
    private int x,y;
    test(int x){ this.x=x;    y=100; }
    public void test(double y){    this.x=(int)y;  }
    public test(int x,int y){ this(x);    this.y=y;  }
}
```

A、 0 B、 1 C、 2 D、 3
38、关于 Java 的修饰符，以下说法错误的是（ B ）
A．抽象类中的方法不一定都是抽象方法
B．final 类中的属性和方法都必须被 final 修饰符修饰
C．类及其属性、方法可以同时有一个以上的修饰符来修饰
D．要使类中某个成员变量只能被它自身访问到，该变量只能用 private 修饰
39、关于 final 的说法，错误的(D)
A、如果变量被标记为 final，其结果是使它成为常数
B、改变 final 变量的值会导致一个编译错误
C、当函数参数为 final 类型时，无法改变该参数的值
D、final 变量定义的时候，必须给出初始值
40、在一个 Java 类中，下列描述正确的选项是（ B ）
A．可以有 2 个以上 package 语句 B．可以有 2 个以上 import 语句
C．可以有 2 个以上 public 类 D．只能有 1 个类定义
41、关键字 this 和 super 可分别用来引用(B)
A、父类的实例对象和当前对象 B、当前对象和父类的实例对象
C、子类的实例对象和父类的实例对象 D、父类的实例对象和子类的实例对象
42、程序输出结果（ C ）

```
public class test {
    static int a=0;
    public static void main(String[] args) {
        test t1=new test();
        test t2=new test();
        t1.a++;
        t2.a++;
        test.a--;
        System.out.print(a);
    }
}
```

A、 0 B、 -1 C、 1 D、编译错
43、下列有关接口的说法，正确的是(D)
A、接口与抽象类是相同的概念 B、一个类不可实现多个接口
C、接口之间不能有继承关系 D、实现一个接口必须实现接口的所有方法
44、关于继承与接口的说法中，不正确的是(D)

A、Java 只支持单继承　　B、一个类可以同时实现多个接口
C、一个类在实现接口的同时还能继承某个基类　　D、接口不能继承
45、abstract 可以与(　　D　　)关键字修饰同一方法
A、private　　B、static　　C、final　　D、public
46、以下哪个接口的定义是正确的（　　D　　）　　注：A1、A2 为已定义的接口
A、 interface　B　{　 void print()　{　} ;}
B、 abstract　interface　B　{ void print() ;}
C、 interface　B　extends　A1　{　int x; }
D、 interface　B　extends　A1,A2{　 }
47、设 B 是类，C 和 D 是接口，以下声明正确的是(　　C　　)
A、public class A implements C extends B {　 }
B、public interface A extends B {　 }
C、public class A extends B implements C, D {　 }
D、public interface A implements C extends B {　 }
48、B 是一个抽象类，C 是 B 的非抽象子类，下列创建对象 x 的语句中正确的是(　 B　)
A. B x= new B( );　　B. B x= new C( );　　C. C x= new B( );　　D. C x= C( );
49、关键字 super 的作用是（　　D　　）
A．用来访问父类被隐藏的非私有成员变量　　B．用来调用父类中被重写的方法
C．用来调用父类的构造函数　　D．以上都是
50、程序输出结果(　 A　)

```
public class test {
    public static int f( final int i ) {
        return ++i;
    }
    public static void main(String[] args) {
        System.out.print(f(2));
    }
}
```

A、编译出错　　B、2　　C、0　　D、3
51、有关 JFrame 描述不正确的是（　D　）
A．JFrame 是一个顶层容器　　B．JFrame 支持多线程
C．JFrame 默认布局是 BorderLayout　　D．JFrame 中不能包含其他容器
52、JPanel 默认的布局是（　B　）
A、BorderLayout　　B、FlowLayout　　C、GridLayout　　D、CardLayout
53、能将容器划分为"East"、 "South"、"West"、"North"、"Center"五个区域的布局管理器是（　 A　）
A、BorderLayout　　B、FlowLayout　　C、GridLayout　　D、CardLayout
54、Swing 提供滚动条功能的容器是（　C　）。
A．JTabbedPane　　B．JLayeredPane　　C．JScrollPane　　D．JSplitPane
55、下列容器中，不是顶层容器的是(　　C　　)
A、Applet　　B、JDialog　　C、JPanel　　D、JFrame
56、点击窗口关闭按钮时仅回收窗口，则 defaultCloseOperation 的取值应为（　 B　　）
A、EXIT_ON_CLOSE　　B、DISPOSE_ON_CLOSE
C、DO_NOTHING_ON_CLOSE　　D、HIDE_ON_CLOSE
57、关于异常处理，以下说法错误的是（　 D ）
A．可以使用 throw 语句抛出异常
B．程序可以使用 try、catch、finally 语句捕获异常
C．无论 try 块中是否发生异常，finally 标识的代码块都会被执行
D．try 语句后只能有一个 catch 语句
58、Java 异常处理用法中，错误的是（　 C　）

A、try...catch　　B、try...finally　　C、catch...finally　　D、try...catch...finally

59、Java 异常的超类是（　B　）

A．Exception　　B．Throwable　　C．Throws　　D．Error

60、在异常处理时，如释放资源、关闭文件等最好由（　C　）来完成

A、try 代码块　　B、catch 代码块　　C、finally 代码块　　D、throw 代码块

61、int x= Integer.parseInt("12.34");会导致的异常是（　D　）

A. NullPointerException　　B. ClassCastException

C. ArithmeticException　　D. NumberFormatException

62、执行 serversocket = new ServerSocket(5432) 时，可能会抛出的(　D　)类型的异常

A、NullPointerException　　B、ClassCastException

C、ArithmeticException　　D、IOException

63、按照功能的不同，Java 流可以分为 (　C　)

A、输入流和输出流　B、字符流和字节流　　C、节点流和处理流　　D、文件流和对象流

64、按照处理数据的单位，Java 流可以分为 (　B　)

A、输入流和输出流　B、字符流和字节流　　C、节点流和处理流　　D、文件流和对象流

65、按照输入的方向，Java 流可以分为 (　A　)

A、输入流和输出流　B、字符流和字节流　　C、节点流和处理流　　D、文件流和对象流

66、下列选项中，(　A　)流使用了缓冲区技术

A、BuffereOutputStream　　B、FileInputStream

C、DataOutputStream　　D、FileReader

67、要串行化某类的对象，则该类就必须实现(　A　)接口

A、Serializable　　B、Runnable　　C、Comparable　　D、Cloneable

68、下列（　D　）是 Java 语言中所定义的字节输入流。

A、OutputStream　　B、Reader　　C、Writer　　D、InputStream

69、Java 中的抽象类 Reader 和 Writer 所处理的流是（　A　）。

A．字符流　　B．对象流　　C．字节流　　D．图像流

68、(　A　) 是 Collection 接口的实现类

A、LinkedList　　B、List　　C、Set　　D、HashMap

70、在以下集合框架中，插入性能最高的是（　B　）

A．ArrayList　　B．LinkedList　　C．List　　D．Collection

71、以下哪个选项不是继承自 Collection 接口（　C　）

A．List　　B．Set　　C．Map　　D．Array

72、List 接口的实现类不包括（　A　）

A、HashSet　　B、LinkedList　　C、ArrayList　　D、Vector

73、(　C　)的功能是遍历并选择集合序列中的对象

A、List　　B、Comparable　　C、Iterator　　D、Comparator

74、下面的 Inner 类编译后生成的 class 文件名为（　B　）

```
public  class  Outer {
    public  class  Inner {
        //
    }
}
```

A. Inner.class　　B. Outer$Inner.class

C. Outer.Inner.class　　D. Outer.class

75、下列关于 Java 线程的说法正确的是（　C　）

A、一个正在运行的程序的称为线程

B、创建线程必须继承 Thread 类

C、Thread 类在 java.lang 包

D、使用线程对象的 run()方法启动一个线程

76、解决多线程并发问题，使用（　C　）修饰代码块。
A、final　　B、supper　　C、synchronied　　D、static
77、关键字 synchronized 可以用于修饰(　　D　　)
A、代码块　　B、方法　　C、类　　D、以上都是
78、线程通过(　　D　　)方法可以休眠一段时间，然后恢复运行。
A、run　　B、setPrority　　C、yield　　D、sleep
79、如下代码创建一个新线程并启动线程，（　C　）可以保证正确创建 target 对象。

```
public static void main(String[] args) {
      Runnable target=new MyRunnable( );
      Thread t=new Thread(target);
}
```

A、

```
public class MyRunnable extends Runnable {
      public void run( ) {          }
  }
```

B、

```
public class MyRunnable extends Runnable {
      void run( ) {   }
  }
```

C、

```
public class MyRunnable   implements Runnable   {
      public void run( ) {          }
  }
```

D、

```
public class MyRunnable   implements Runnable   {
      void run( ) { }
  }
```

80、当线程调用 start( )后，其所处状态为（ C　）
A、阻塞状态　　B、运行状态　　C、就绪状态　　D、新建状态
81、如果采用继承 Thread 类的方式创建线程，则需要重写 Thread 类的（　D　）方法
A．start　　B．stop　　C．interrupt　　D．run
82、创建守护线程的方法是（）
A．start　　B．setDaemon　　C．interrupt　　D．run
83、以下说法中关于线程通信的说法错误的是（　D　）
A、可以调用 wait()、notify()、notifyAll()三个方法实现线程通信
B、wait()、notify()、notifyAll()必须在 synchronized 方法或者代码块中使用
C、wait()有多个重载的方法，可以指定等待的时间
D、wait()、notify()、notifyAll()是 Object 类提供的方法，子类可以重写
84、关于 sleep()和 wait()，以下哪个选项描述错误的是（　　D　　）
A．sleep 是线程类的方法，wait 是 Object 类的方法
B．sleep 不释放对象锁，wait 放弃对象锁
C．sleep 暂停线程、但监控状态仍然保持，结束后会自动恢复
D．wait 后进入等待锁定池，只有针对此对象发出 notify 方法后获得对象锁进入运行状态
85、关于 Socket 通信编程，以下选项错误的是（　D　）
A．服务器端通过 new ServerSocket()创建 TCP 连接对象
B．服务器端通过 TCP 连接对象调用 accept()方法创建通信的 Socket 对象
C．客户端通过 new Socket()方法创建通信的 Socket 对象
D．客户端通过 new ServerSocket()创建 TCP 连接对象
86、在 Java 网络编程中，使用客户端套接字 Socket 创建对象时，需要指定（　A　）
A. 服务器主机名称和端口　　B. 服务器端口和文件
C. 服务器名称和文件　　D. 服务器地址和文件
87、在 Java 语言中，与 Internet 地址有关的类是(　D　　)
A、Socket　　B、UDPSocket　　C、DatagramSocket　　D、InetAddress

88、ServerSocket 的监听方法 accept( )方法的返回值类型是（ A ）
A. Socket　　B. Void　　C. Object　　D. DatagramSocket
89、下面代码错误定义的是（　A　）
```
class Test {    // A
        private int i=1;   // B
        Test(){   };    // C
        abstract   void   bb();    // D
        public void cc() {      i++;   }
 }
```
90、下列程序段执行后的结果是(　B　)
```
String s = new String("abcdefg");
for (int i=s.length()-1; i>=0; i-=2){
    System.out.print(s.charAt(i));
}
```
A.aceg　　B. geca　　C.defg　　D.abcd
91、下面程序输出的结果是（　A　）
```
    String   s1=new String("abc");
    String   s2=new String("abc");
    boolean b1=s1.equals(s2);
    boolean b2=(s1==s2);
    System.out.print(b1+"    "+b2);
```
A.true　false　　B.false　true　　C.true　true　　D.false　false
92、有以下程序片段，下列哪个选项不能插入到行 1。（　A　）
```
1.
2.public   class   A{
3.    //do sth
4.}
```
A、public class B{ }　　B、package　mine;　　C、class B{　}　　D、import　java.util.*;
93、类 ABC 定义如下：
```
1．public   class   ABC{
2．        public   int   max( int   a, int   b) {    }
3．
4．}
```
将以下哪个方法插入行 3 是不合法的。（　B　）
A、public　float　max(float　a, float　b, float　c){　}
B、public　int　max (int　c,　int　d){　}
C、public　float　max(float　a,　float　b){　}
D、private　int　max(int a, int b, int c){　}
94、将以下哪个方法的定义插入第 6 行后，不会引发编译错误？(　B　)
```
1．  class Super{
2．        public void   a(){    }
3．  }
4．
5．  public class Sub extends Super{
6．
7．}
```
A．public int　a(){return　1;}　　B．public int a(int　x){ return x; }
C．void a(){　}　　D．public int a(float　x){return x;}
95、下面描述不能使程序通过编译的是（　D　）

```
1. class Super{
2.          private int a;
3.          protected Super(int a){this.a = a;}
4. }

11. class Sub extends Super{
12.   public Sub(int a){ super(a);}
13.   public Sub(){this.a = 5;}
14. }
```

A. 删除第 13 行

B. 将第 13 行改为：public Sub(){ this(5);}

C. 将第 13 行改为：public Sub(){ super(5);}

D. 将第 13 行改为：public Sub(){ }

二、程序填空

1、两种方法实现回文判断，请完善。

```
public class Ex_2 {
    //方法 1
    public static boolean isPalindrome(String s) {
        char ch1, ch2;
        for( int i = 0; i < s.length()/2; i++ ) {
            ch1 = ______s.charAt(i) ______;
            ch2 = ______s.charAt( s.length()-1-i ) ______;
            if ( ch1 != ch2 )  {
                return false;
            }
        }
        return true;
    }
    //方法 2
    public static boolean isPalindrome(String s) {
        StringBuffer sb = new StringBuffer(s);
        return sb. ______reverse()______.______toString()______.equals(s);
    }
}
```

2、所谓 "水仙花数 "是指一个三位数，其各位数字立方和等于该数本身。输出 3 位数的水仙花数，请完善。

```
public class demo {
    public static boolean isFlower(int n) {
        int ge, shi, bai, sum = 0;
        ge = ______n % 10; ______              //个位
        shi = ______n % 100 / 10; ______       //十位
        bai = n / 100;              //百位
        sum = ge*ge*ge + shi*shi*shi + bai*bai*bai;
        if (______sum == n______)
            return true;
        else
            return false;
    }
```

```
    public static void main(String[] args) {
        for (int n = 100; n < 1000; n++) {
            if ( isFlower(n) )
                System.out.println(n + "是一个水仙花数");
        }
    }
}
```

3、汉诺塔-递归实现

思路：要知道如何将 n 块盘子从 A 通过 B 移动到 C（要求大盘子总在下面），可以先将上面的 n-1 块盘子从 A 通过 C 移动到 B，然后将最大的盘子从 A 移动到 C，最后再将上面的 n-1 块盘子从 B 通过 A 移动到 C。

```
public class Test {
    /*
    @param n     //汉诺塔上盘子的个数
    @param A     //开始时有盘子的汉诺塔
    @param B     //中介汉诺塔
    @param C     //要将盘子移动到上面的目标汉诺塔
    */
    public static void Hanoi(int n,char A,char B,char C){
        if(n<=0){
            throw new IllegalArgumentException("n must be >=1");
        }
        if(n==1){
            System.out.println(A+"->"+C);
        }
        else{
            ______Hanoi(n-1,A,C,B) ______;        // 将除去最大的盘子的 n-1 个盘子从 A 通过 C 移动到 B
            System.out.println(A+"->"+C);   //将最大的盘子从 A 移动到 C
            ______Hanoi(n-1,B,A,C) ______;        //将除去最大的盘子的 n-1 个盘子从 B 通过 A 移动到 C
        }
    }

    public static void main(String[] args) {
        Hanoi(3,'A','B','C');
    }
}
```

3、以下代码的功能包括：

a. 定义了 Student 类；

b. 创建类 Student 类的实例 student；

c. 调用 set_id 方法将学号设置为 1234；

d. 调用 show 方法输出了学号。

请完善以下代码。

```
class Student{
    int id;
    void setId(int id)   {   this.id=id;   }
    void show()    {   System.out.println("ID is :"+id); }
}
```

```
public class Demo{
    public static void main(String args[]){
    ______ Student student = new Student()______ ;        //构造一个学生
    ______ student.setId(1234)_______;        //设置学号为 1234
    _______ student.show()______;     //调用 show()显示学号
    }
}
```

4、定义一个三角形 Triangle 类：
数据成员：a, b, c（double 类型）；
主要的操作：
(1) 判断是否构成三角形 isTriangle( )
(2) 计算三角形面积 getArea( )
(3) 显示三角形信息 toString()

```
public class Triangle{
        private double a,b,c;
        public Triangle(){
        }
        public Triangle(double a,double b,double c){
            this.a=a;  this.b=b;  this.c=c;
        }
        public boolean isTriangle(){
            return _________a + b > c && a + c > b && b + c > a________;
        }
        public double getArea(){
            double p=(a+b+c)/2;
            if(____isTriangle()____)
                return _____Math.sqrt_____ (p*(p-a)*(p-b)*(p-c));
            else
                return 0;
        }
        public void toString()    {
            System.out.println("三边为："+a+","+b+","+c);
        }

        public static void main(String []s) {
            Triangle t=new Triangle(3,4,5);
            t.showSize();
            System.out.println ("面积为："+ t.getArea() );
        }
    }
```

5、以下程序运用泛型技术实现数组元素交换，请完善。

```
class ArrayUtils_____<T>_____ {
    public void changePosition(____T[ ] arr ____, int index1, int index2) {
        _____ T temp ____ = arr[index1];
        arr[index1] = arr[index2];
        arr[index2] = temp;
    }
```

```
}
public class Demo {
    public static void main(String[] args) {
        ArrayUtils<String> au = new ArrayUtils<String>();
        String[] arr = new String[] { "aa", "bb", "cc", "dd", "ee" };
        au.changePosition(arr, 1, 3);
    }
}
```

6、以下程序运用泛型技术实现数组倒排，请完善。

```
class ArrayUtils____<T>____ {
    public void reverse( ____T[] arr____ ) {
        for (int i = 0; i < ___ arr.length / 2___; i++) {
            ____(6) T temp ___ = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = temp;
        }
    }
}
public class Demo {
    public static void main(String[] args) {
        ArrayUtils<Integer> au = new ArrayUtils<Integer>();
        Integer[] arr = new Integer[] { 1, 2, 3, 4, 5 };
        au.reverse(arr);
    }
}
```

7、以下代码是集合框架 Map 和 Set 的综合应用，请完善。

```
public class Demo {
    public static void main(String[] args) {
        Map<String, Double> course = new HashMap<String, Double>();
        course.put("Java", 3.0);
        course.put("Java EE", 2.5);
        System.out.println("Map 集合元素个数大小：" + course. ___size()___);
        System.out.println("Java EE 课程的学分：" + course. ___get___ ("Java EE"));
        System.out.println("使用迭代器遍历结果：");
        Set<String> set = course. ___keySet()___;
        Iterator<String> it = set.iterator();
        while ( it. ___hasNext()___ ) {
            String key = ___it.next()___;
            System.out.println(key + "---" + course.get(key));
        }
    }
}
```

8、从键盘获取一行输入字符串，去除重复字符后输出。

```
pulbic class demo {
    public static void main(String[] args) {
        System.out.println("请输入一行字符串：");
```

```
        Scanner sc = new Scanner(System.in);
        String line = sc. ______nextLine()______;
        char[] c = line. ______toCharArray()______;
        HashSet<Character> set = new HashSet<Character>();
        for (int i = 0; i < c.length; i++) {
            _____________set.add(c[i]) _____________;
        }
        for (Character ch : set) {
            System.out.print(ch + " ");
        }
    }
}
```

9、集合排序程序，输出结果为：

排序前：

[(x = 3, y = 2), (x = 1, y = 3), (x = 1, y = 2)]

排序后：

[(x = 1, y = 2), (x = 1, y = 3), (x = 3, y = 2)]

排序规则要求：x 值小的对象排在前面，如果 x 值相同，则 y 值小的对象排在前面。

```
class Point {
    int x;
    int y;
    public Point(int x, int y) {
        this.x = x;
        this.y = y;
    }
    public String toString() {
        return "(x = " + x + ", y = " + y + ")";
    }
}
class myComparator _______implements Comparator_______{
    public int compare(Object o1, Object o2) {
        Point p1 = (Point) o1;
        Point p2= (Point)o2;
        if ( p1.x != p2.x )
            return _______p1.x - p2.x_______;
        else
            return _______p1.y - p2.y_______;
    }
}
public class Test{
    public static void main(String[] args) {
        List list = new ArrayList();
        list.add(new Point(3,2));
        list.add(new Point(1,3));
        list.add(new Point(1,2));
        System.out.println("排序前：");
        System.out.println(list);
        Collections.sort(list, ___new myComparator()__);
```

```
        System.out.println("排序后：");
        System.out.println(list);
    }
}
```

10、学生信息使用 Map<String,String> 来表示，分别包含学号和姓名。使用 List<Map<String,String>>集合来存放多个学生信息。检查学生的学号是否有重复，请完善。

```
class Demo {
    public static void main(String[] args) {
        List<Map<String, String>> studentList = new ArrayList<>();
        Map<String, String> stu1 = new HashMap<>();
        stu1.put("id", "101");
        stu1.put("name", "小明");
        studentList.add(stu1);
        Map<String, String> stu2 = new HashMap<>();
        stu2.put("id", "102");
        stu2.put("name", "小丽");
        studentList.add(stu2);
        Map<String, String> stu3 = new HashMap<>();
        stu3.put("id", "101");
        stu3.put("name", "小王");
        studentList.add(stu3);
        for (int i = 0; i < ___studentList.size()___; i++)
            for (int j = ___i + 1___; j < studentList.size(); j++) {
                Map<String, String> tmp1 = studentList.get(i);
                Map<String, String> tmp2 = studentList.get(j);
                String id1 = tmp1.get("id");
                String id2 = tmp2.get("id");
                if (___id1.equals(id2) ___) {
                    System.out.println("编号" + i + "和编号" + j + "重复，请检查");
                    ___break___;
                }
            }
    }
}
```

11、如果 Student 的 id 和 name 相同则认为是同一个学生，利用 Set 集合不可重复特性，使用 Set<Student>存放 Student 信息完成去重任务。

```
class Student {
    String id;
    String name;
    public Student(String id, String name) {
        this.id = id;
        this.name = name;
    }
    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;
    }
    @Override
    public boolean ___equals(Object obj) ___ {
        Student stu = (Student) obj;
        return this.id.equals(stu.getId())&&this.name.equals(stu.getName());
    }
    @Override
    public int ___hashCode()___ {
        return Integer.parseInt(id);
    }
    @Override
    public String ___toString()___ {
        return "id=" + id + ", name=" + name ;
    }
}
class Demo {
    public static void main(String[] args) {
        Set<Student> studentList=new HashSet<>();
        Student stu1=new Student("100","小明");
        Student stu2=new Student("101","小丽");
        Student stu3=new Student("100","小明");   //重复对象
        studentList.add(stu1);
        studentList.add(stu2);
        studentList.add(stu3);
        for (Student s:studentList){
            System.out.println(s);
            //输出结果：
            //id=100, name=小明
            //id=101, name=小丽
        }
    }
}
```

12、定义 PrintThread 线程用于输出 1～100，每输出一个数休眠 1 秒，在 main 方法中启动该线程。

```
public class PrintThread extends Thread {
    public void ____run()____ {
        for(int i=1;i<=100;i++) {
            System.out.println(Thread.____currentThread()____.getName()+": "+i);
            try {
                    ____Thread.sleep(1000) ____;     // 休眠 1 秒
                }catch (Exception e) {
                    System.out.println(e.toString());
```

```
                }
            }
        }
}
public class Test{
    public static void main(String[] args){
        PrintThread t= new PrintThread("a");
        t. __start()__; //启动线程
    }
}
```

13、以下线程程序模拟乌龟和兔子赛跑。

```
class Animal extends Thread {
    private String name;
    private int speed; //  速度
    private int distance; //  距离
    private int sumdistance = 0; //  已跑距离
    public Animal(String name, int speed, int distance) {
        this.name = name;
        this.speed = speed;
        this.distance = distance;
    }
    public void run() {
        int time=(int) (Math.random()* 1000) + 500;    //随机一段时间
        while____ ( sumdistance < distance ) ____ {
            try {
                Thread.sleep(time);
            } catch (Exception e) {
                System.out.println(e);
            }
            _________sumdistance += speed*time_________;     //  获得当前跑了多长距离
            System.out.println(name + " :  已经跑了  " + sumdistance);
        }
    System.out.println(name + "  终于冲过终点了  ! ");
    }
}
public class AnimalRace {
    public static void main(String[] args) {
        Animal rabbit = new Animal("兔子", 20, 50);
        Animal turtle = new Animal("乌龟", 15, 50);
        new Thread(rabbit).start();;
        new Thread(turtle).start();;
    }
}
```

14、两个线程均打印 10 次线程名，两次打印之间间隔 1 秒。

```
class ThreadDemo _________extends Thread_________ {
    public ThreadDemo(String str) {
        _________super(str); _________
```

```
    }
    @override
    public void run() {
        for(int i=0;i<10;i++){
            System.out.print(" "+ this. ____getName()____ );
            try {
                Thread.sleep(1000);
            }catch(InterruptedException e){
                System.out.println(e.getMessage());
            }
        }
    }
}
public class Test {
    public static void main(String[] args) {
        ThreadDemo thread1=new ThreadDemo("T1");
        ThreadDemo thread2=new ThreadDemo("T2");
        thread1.start();
        thread2.start();
    }
}
```

15、使用 Java 线程同步，完成三个售票窗口同时出售 10 张票。

```
class Station    ________ implements Runnable ________ {
    private int tick = 10;   //票数
    public void ________run()________ {
        while (tick > 0) {
            ________synchronized (this) ________ {
                if (tick > 0) {
                    System.out.println(Thread.currentThread().getName() + "卖出了第" + tick + "张票");
                    tick--;
                } else {
                    System.out.println("票卖完了");
                }
            }
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        } //end while
    }   //end run
}
public class test {
    public static void main(String[] args) {
        Station station=new Station();
        Thread t1=new Thread(station,"窗口 1");
        Thread t2=new Thread(station,"窗口 2");
        Thread t3=new Thread(station,"窗口 3");
```

```
        t1.start();        t2.start();        t3.start();
    }
}
```

16、使用 Java 线程同步，模拟汽车加油操作：当前有 2 台加油设备，20 辆车需要加油，每辆车加满油需 30 秒。

```
public class Demo {
    public static void main(String[] args) {
        OilEquipment oe=new OilEquipment();
        Thread t1=new Thread(oe);
        Thread t2=new Thread(oe);
        t1.start();
        t2.start();
    }
}
class OilEquipment implements Runnable{
    private int nums=20;
    public void run() {
        while(true){
            _______ synchronized(this) _______ {
              if( nums>0 ){
                System.out.println(" 正 在 由 设 备 "+_______ Thread.currentThread().getName()_______ +" 给 第 "+nums+"辆车加油");
                nums--;
              }
              else
                break;
            }
           try {
                Thread.sleep(30000);
            }catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```

17、给下面程序会造成死锁。

```
public class Test{
    public static void main(String[] args){
        new Thread(new LockThread1()).start();
        new Thread(new LockThread2()).start();
    }
}
class Lock{
    public static Object lock1 = new Object();
    public static Object lock2 = new Object();
}
class LockThread1 implements Runnable {
    public void run(){
```

```
                for(int i=0;i<2;i++){
                    synchronized(Lock.lock1){
                        try {
                            Thread.sleep(3000);
                        } catch (InterruptedException e) { e.printStackTrace();       }
                        synchronized(_____Lock.lock2_____){
                        }
                    }
                }
        }
}
class LockThread2 implements Runnable{
                public void run(){
                for(int i=0;i<2;i++){
                    synchronized(____Lock.lock2_____){
                            try {
                            Thread.sleep(3000);
                        } catch (InterruptedException e) { e.printStackTrace();}
                            synchronized(_____Lock.lock1_____){
                            }
                        }
                    }
                }
}
```

18、下面程序不使用缓冲流完成文件复制功能，请完善。

```
public class Test {
        public static void main(String[] args) throws Exception{
            FileInputStream fis = new FileInputStream("c:\\aaa\\a.jpg");
            FileOutputStream fos = new FileOutputStream("c:\\bbb\\b.jpg");
            System.out.println("正在拷贝...");
            byte[] b = new byte[ ____fis.available()____ ];     //一次性获取文件的字节大小
            ___fis.read(b); ___
            ___fos.write(b) ____;
            fis.close();
            fos.close();
            System.out.println("复制完成");
        }
}
```

19、下面程序使用缓冲流完成文件复制功能，请完善。

```
public void copyFile(String srcfile, String targetfile) throws IOException {
        BufferedInputStream br = new BufferedInputStream( ____new FileInputStream(srcfile) ____ ) ;
        BufferedOutputStream bw = new BufferedOutputStream( ____new FileOutputStream(targetfile) ____);
        int i;
        while ( ( i = ____br.read()____ ) != -1 ) {
            ____bw.write(i) ____;
        }
        br.close();
```

```
    bw.close();
}
```

20、以下程序使用 BufferedWriter 类在 D:\Hello.txt 文件中写入 10 万个数，每个数字占一行。

```
public class Test {
    public static void main(String[] args) throws IOException{
        BufferedWriter bw =new BufferedWriter(_____new FileWriter("D:\\Hello.txt")_____);
        for (int i=1;i<=100000;i++){
            bw.write( i );
            _____bw.newLine()_____;
        }
        bw.close();
    }
}
```

21、Socket 编程，当服务器端（127.0.0.1:520）收到客户端发送的"你好"后，将发送"欢迎"给客户端。

```
class Client {
    public static void main(String[] args) throws Exception {
        Socket socket = new _____Socket("127.0.0.1", 520) _____;
        //向服务器端发送消息
        BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(_____socket.getOutputStream()_____));
        bw.write("你好\n");
        bw.flush();
        //读取服务器返回的消息
        BufferedReader br = new BufferedReader(new InputStreamReader(_____socket.getInputStream()_____));
        String msg = br.readLine();
        System.out.println("接收服务器信息：" + msg);
        br.close();
        bw.close();
        socket.close();
    }
}
class Server {
    public static void main(String[] args) throws Exception {
        ServerSocket ss = new _____ServerSocket(520) _____;
        Socket socket = ss. _____accept()_____;
        //读取客户端消息
        BufferedReader br = new BufferedReader(new InputStreamReader(socket.getInputStream()));
        String msg = br.readLine();
        if (_____msg.equals("你好")_____) {
            System.out.println("接收客户端的消息：" + msg);
            //向客户端发送消息
            BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            bw.write("欢迎\n");
            bw.flush();
            bw.close();
        }
        br.close();
        socket.close();
```

```
        ss.close();
    }
}
```

22、爬取某个网页，将其内容存储在 data.html 文件中。

```
public class Test {
    String url="某个网页地址";
    public static void main(String[] args) throws IOException {
        URL url = _____new URL(url) _____;
        URLConnection conn = url. _____openConnection()_____;
        BufferedReader br = new BufferedReader(new InputStreamReader( _____conn.getInputStream() _____));
        //注：上面两行可用 openStream()方法简写：
        //BufferedReader br = new BufferedReader(new InputStreamReader( _____url.openStream()_____ ))
        BufferedWriter bw = new BufferedWriter(new _____FileWriter_____ ("data.html"));
        String line;
        while ((line = br.readLine()) != null) {
            System.out.println(line);
            _____bw.write(line) _____;
            bw.newLine();
        }
        br.close();
        bw.close();
    }
}
```

23、Swing 编程，给按钮添加点击事件，请完善。

```
class Demo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("demo");
        frame.setSize(400,300);
        frame.setLocation(200,200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JButton btn = new JButton("测试");
        Listener listener=new Listener();
        btn.addActionListener(_____listener_____);
        frame. _____add_____ (btn, BorderLayout.CENTER);
        frame.setVisible(true);
    }
}
class Listener _____implements_____ActionListener{
    @Override
    public void _____actionPerformed_____ (ActionEvent e) {
        System.out.println("button clicked");
    }
}
```

24、Swing 编程，下拉列表实现省市的级联操作，请完善。

```
class Demo {
    public static void main(String[] args) {
```

```
        JFrame frame=new JFrame("测试");
        frame.setSize(400,300);
        frame.setLocation(200,200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        Panel panel=new Panel();
        JComboBox comboBox1 = new JComboBox();
        comboBox1.setModel(new DefaultComboBoxModel(_____new String[]_____{"请选择", "湖北", "湖南"}));
        panel.add(comboBox1);
        JComboBox comboBox2 = new JComboBox();
        panel.add(comboBox2);
        comboBox1.addItemListener(new ItemListener() {
            public void _____itemStateChanged_____ (ItemEvent e) {
                int index =comboBox1. _____getSelectedIndex()_____;
                switch (index) {
                    case 0:
                        comboBox2.removeAllItems();
                        break;
                    case 1:
                        comboBox2.removeAllItems();
                        comboBox2.addItem("武汉");
                        comboBox2.addItem("宜昌");
                        break;
                    case 2:
                        comboBox2.removeAllItems();
                        comboBox2.addItem("长沙");
                        comboBox2.addItem("湘潭");
                }
            }
        });
        frame.add(panel);
        frame.setVisible(true);
    }
}
```

三、程序阅读题

1、写出下面程序运行结果。

```
class Demo {
    public static void f1(int i){
        i++;
        System.out.println(i);
    }
    public static void f2(StringBuffer s){
        System.out.println(s);
        s.append("b");
        System.out.println(s);
    }
    public static void main(String[] args) {
        int i=1;
```

答案：

2

1

a

ab

ab

```
        f1(i);
        System.out.println(i);
        StringBuffer a=new StringBuffer("a");
        f2(a);
        System.out.println(a);
    }
}
```

2、写出下面程序运行结果。

```
class A {
    public A() {
        System.out.print("1");
    }
    static { System.out.print("2"); }
}
class B extends A {
    public B() {
        System.out.print("3");
    }
    static { System.out.print("4"); }
}
public class test {
    public static void main(String[] args) {
        new B();
    }
}
```

答案：
2413
说明：先执行 A 的静态代码块，再 B 的静态代码块，然后 A 构造函数，最后 B 构造函数

3、写出下面程序运行结果。

```
class Value {
    static int c = 0;
    Value() {
        c = 100;
    }
    Value(int i) {
        c = i;
    }
    static void inc() {
        c++;
    }
}
class Demo {
    public static void prt(String s) {
        System.out.println(s);
    }
    Value v = new Value(1);
    static Value v1, v2;
    static {
        prt("v1.c=" + v1.c + "    v2.c=" + v2.c);
        v1 = new Value(2);
```

答案：
v1.c=0 v2.c=0
v1.c=2 v2.c=2
ct.c=1
v1.c=1 v2.c=1
v1.c=2 v2.c=2

```
        prt("v1.c=" + v1.c + "    v2.c=" + v2.c);
    }
    public static void main(String[] args) {
        Demo demo = new Demo();
        prt("ct.c=" + demo.v.c);
        prt("v1.c=" + v1.c + "    v2.c=" + v2.c);
        v1.inc();
        prt("v1.c=" + v1.c + "    v2.c=" + v2.c);
    }
}
```

4、写出下面程序运行结果。

```
public class Test {
    public static void main(String[] args) throws IOException {
        String s1 = "abc";
        String s2 = "abc";
        System.out.println( s1==s2 );
        System.out.println( s1.equals(s2) );
        String s3 = new String("abc");
        String s4 = new String("abc");
        System.out.println( s3==s4 );
        System.out.println( s3.equals(s4) );
        String  s5=s1;
        System.out.println(s1==s5);
        System.out.println(s1.equals(s5));
    }
}
```

答案：
true
true
false
true
true
true

5、下面程序运行结果是（　　）

```
class Alpha{
    public void foo(){
        System.out.print("A");
    }
}
class Beta extends Alpha{
    public void foo(){
        System.out.print("B");
    }
}
public class test{
     public static void main(String[] args){
            Alpha a = new Beta();
            Beta b = (Beta)a;
            a.foo();
            b.foo();
     }
}
```

答案：
BB

6、下面程序运行结果是（　　）

```
public class Test {
    public static void main(String[] args) throws IOException {
        int i = 0;
        String greetings[] = {"12", "abc"};
        while (i < 3) {
            try {
                System.out.println(greetings[i]);
                int k = Integer.parseInt("a");
            } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println("下标越界");
            } catch (NumberFormatException e) {
                System.out.println("格式不正确");
            }
            i++;
        }
    }
}
```

答案：
12
格式不正确
abc
格式不正确
下标越界

7、写出下面程序运行结果。

```
class A{
    public String s = "A";
    public void setS(String s){          this.s = s;          }
    public String getS(){          return this.s;          }
}
public class B extends A{
    public String s = "B";
    public void setS(String s){          this.s = s;          }
    public String getS(){          return this.s;          }
    public static void main(String[] args){
        A a = new A();
        B b = new B();
        System.out.println(a.s);
        System.out.println(b.s);
        a.setS("a");
        b.setS("b");
        a = b;
        System.out.println(a.s);
        System.out.println(b.s);
        System.out.println(a.getS());
        System.out.println(b.getS());
    }
}
```

答案：
A
B
A
b
b
b

8、写出下面程序运行结果。

```
class A {
    public static String staticGet() {       return "1";    }
```

```
    public String dynamicGet() {       return "2";       }
}
class B extends A {
    public static String staticGet() {       return "3";    }
    public String dynamicGet() {       return "4";       }
}
public class test {
    public static void main(String[] args) {
        A a = new B();
        System.out.print( a.staticGet() );
        System.out.print( a.dynamicGet() );
    }
}
```

答案：
14
说明：staticGet()静态方法不会覆盖，a.staticGet()是父类方法，a.dynamicGet()是多态体现是儿子方法

9、写出下面程序运行结果。

```
class Atom{
    Atom(){
        System.out.print("A");
    }
}
class Rock extends Atom{
    Rock(String type){
        System.out.print(type);
    }
}
public class Mountain extends Rock{
    Mountain(){
        super("B");
        new Rock("C");
    }
    public static void main(String[] a){
        new Mountain();
    }
}
```

答案：
ABAC
注意：Rock(String type)中会隐式调用父类默认构造函数 Atom()

10、写出下面程序运行结果（　　）

```
public class Test {
    private static void changeValue(int value) {
        value*=2;
    }
    public static void sort( int[] a ) {
        Arrays.sort(a);
    }
    private static void changeValue(int[] arr) {
        for(int i=0;i<arr.length;i++)
            arr[i]*=2;
    }
    public static void main(String[] args) {
        int[] arr={5,4,3,2,1};
```

答案：
[5, 4, 3, 2, 1]
[1, 2, 3, 4, 5]
[2, 4, 6, 8, 10]

```
        changeValue(arr[0]);
        System.out.println( Arrays.toString(arr) );
        sort(arr);
        System.out.println( Arrays.toString(arr) );
        changeValue(arr);
        System.out.println( Arrays.toString(arr) );
    }
}
```

11、写出下面程序运行结果。

```
class A {
    public void f1() {
        System.out.println("a1");
    }
    public final void f2() {
        System.out.println("a2");
    }
    public static void f3() {
        System.out.println("a3");
    }
}
class B extends A {
    public void f1() {
        System.out.println("b1");
    }
    public static void f3() {
        System.out.println("b3");
    }
}
class Demo {
    public static void main(String[] args) {
        A t = new B();
        t.f1();
        t.f2();
        t.f3();
        A.f3();
        B.f3();
    }
}
```

答案：

b1
a2
a3
a3
b3

12、写出下面程序运行结果。

```
class A {
    public A() {
        System.out.println("I am A");
    }

    public A(String s) {
        this();
        System.out.println("I am " + s);
```

答案：

I am A
I am B
I am C

```
    }
}
class B extends A {
    public B() {
        this("I am D");
    }

    public B(String s) {
        super(s);
        System.out.println("I am C");
    }
}
class Demo {
    public static void main(String[] args) {
        A b = new B("B");
    }
}
```