

2024-2025 年《Java 程序设计》期末试题

注意事项:

1. 答卷前, 考生务必将自己的姓名和准考证号填写在答题卡上。
2. 回答选择题时, 选出每小题答案后, 用铅笔把答题卡对应题目的答案标号涂黑。如需改动, 用橡皮擦干净后, 再选涂其它答案标号。回答非选择题时, 将答案写在答题卡上。写在本试卷上无效。
3. 考试结束后, 将本试卷和答题卡一并交回。请认真核对监考员在答题卡上所粘贴的条形码上的姓名、准考证号与您本人是否相符。

一、选择题

1. 下列使用工具的命令中属于 Java 编译器的是(_____)
A. javac.exe B. javadoc.exe C. java.exe D. javaw.exe
2. 定义一个普通类需要使用到的关键字是(_____)
A. interface B. final C. abstract D. class
3. `StringBuilder str = new StringBuilder()`, 则依次经过 `str.append("Hello")`, `str.append(true)`, `str.append((double)3/2)`, `str.insert(4,"World")` 运算后, `str.toString()` 的值是(_____)
A. HellWorldtrue3/2 B. HelloWorldtrue1.0
C. HelloWorldtrue1 D. HellWorldtrue1.5
4. 通过控制台进行输入时, 需要构造一个与“标准输入流” `System.in` 关联的对象是(_____)
A. String 对象 B. Scanner 对象 C. Writer 对象 D. Reader 对象
5. 在 Java 语言中, 下列针对某一行的注释是(_____)
A. `/** String str = "abc";*/` B. `//double a = 3;`
C. `% int b=0;` D. `/* double i=3.0:*/`
6. 已知存在一个 `Employee` 类, 现需定义一个 `Manager` 类, 二者属于继承关系, 则下列说法正确的是()
A. `Manager` 类是超类 B. `Manager` 类是父类
C. `Employee` 类是子类 D. `Manager` 类是派生类
7. 下列语句, 书写不正确的是()
A. `BigInteger a = BigInteger.valueOf(8)+BigInteger.valueOf("7");`
B. `BigInteger a = BigInteger.valueOf(8);`
C. `BigInteger a = BigInteger.valueOf(Long.parseLong("7"));`
D. `BigInteger a = BigInteger.valueOf(8).add(new BigInteger("7"));`

8. 已知"public class Manager extends Employee {...}", 在 Employee 类中有一个"public double getSalary(){...}"方法, 在 Manager 类中也有一个"public double getSalary(){...}"方法, 则"double s = new Employee("张三", 40000).getSalary();"调用的 getSalary 方法是哪个类的方法()
- A. Manager 类
B. Manager 类或 Employee 类
C. Manager 类和 Employee 类
D. Employee 类
9. 下列关于数组的语句正确的是()
- A. double[] m = new double[-4];
B. String[7] m = new String[];
C. boolean m[] = new boolean[6];
D. int m; m = new double[3];
10. 下列 Java 语言中用于字符串的正确表示形式是
- A. 'abc' B. {'a', 'b', 'c'} C. "abc" D. {"abe", "abc", "abe"}
11. 在 Java 的基本数据类型中有 2 种浮点型, 下列占用内存存储空间最大的浮点型是()
- A. byte B. float C. long D. double
12. 在构造对象的过程中, 程序初始化时执行的过程是()
- A. "静态属性→构造方法→非静态属性"
B. "非静态属性→构造方法→静态属性"
C. "静态属性→非静态属性→构造方法"
D. "非静态属性→静态属性→构造方法"
13. 下列语句执行后, x 的值为()
- ```
int a = 4, b = 5, x = 3;
if (a == --b)
 x = x * b;
```
- A. 12      B. 20      C. 3      D. 15
14. Java 异常处理机制中, finally 块中语句不执行的唯一情况是()
- A. 异常处理代码无异常发生  
B. 异常处理代码中执行了 System.exit(1) 语句  
C. 异常处理代码中执行了 continue 语句  
D. 异常处理代码中执行了 break 语句
15. 下列情况不属于异常的是()
- A. 用户输入错误      B. 设备错误      C. 代码错误      D. 数据类型自动转换

16. 假设要在 "haut.oop.base" 包中定义一个 Employee 类, 该类中的 public 修饰的成员变量和成员方法只能被同一个包中的类(含当前类)所访问和使用, 下列符合需求的定义是()

- A. package haut.oop.base; public class Employee { }
- B. package haut.oop.base; class Employee{ }
- C. public class Employee{ }
- D. class Employee { }

17. 假设 x 的初始值为 3, 则经过  $x = x ++ x -- x *= x$  运算后, 的值为()

- A. 1
- B. 0
- C. -3
- D. 3

18. 下列程序的输出是()

```
public static void main(String[] args){
 int[] arr = new int[]{25, 28, 31, 30, 29};
 Arrays.sort(arr);
 System.out.println(Arrays.toString(arr));
}
```

- A. [25, 28, 29, 30, 31]
- B. [25, 28, 29, 30, 31 ]
- C. [31, 30, 29, 28, 25]
- D. (31, 30, 29, 28, 25)

19. 通过控制台进行输入时, 需要构造一个与“标准输入流” System.in 关联的对象是()

- A. Scanner 对象
- B. Reader 对象
- C. String 对象
- D. Writer 对象

20. 已知 "public class Manager extends Employee {...}", 在 Employee 中有一个 "public double getSalary(){...}" 方法, Manager 也有一个 "public double getSalary(){...}" 方法, 则这种现象叫做()

- A. 方法的定义或实现
- B. 方法的覆盖或重写
- C. 方法的重构或实现
- D. 方法的继承或派生

## 二、判断题

1. 类是相同行为和状态的诸多对象的统称。
2. 在对一个 long 型变量进行赋值时, 数字后面要加 L 或 l。
3. 如果类中没有任何构造方法, 系统会自动创建一个不带参数的构造方法。
4. BigInteger 类可以实现任意精度的整数运算。
5. 运算符 && 和 & 是没有区别的。
6. 无参数构造器创建的对象, 对象的状态被为适当的默认值。
7. 通配符的限定若用 <? extends Manager> 来表示, 其含义这个通配符限制为 Manager 类及该类的所有父类型。

8. `import static` 语句可以导入静态方法和静态字段。
9. 在 Java 语言中是不支持单重继承的, 但是可以使用接口机制来实现多重继承的功能。
10. 在语句块内部可以访问之前在外部声明的变量。
11. 静态成员属于类, 不属于任何单个的对象。
12. 若循环体含有多条语句, 则必须置于一对花括号中, 否则视为语法错误。
13. `Collections` 是一个接口类, 进一步提供一系列的静态方法, 实现对集合的排序、替换、交换、搜索、拷贝等操作。
14. 子类对父类进行方法覆盖时, 需要返回值类型、函数名和参数列表都一模一样。
15. 继承 `java.lang.Thread` 类和实现 `java.lang.Runnable` 接口是 Java 中实现多线程的两种方式。
16. 用链接存储结构存储的线性表称为链表。
17. 可以在变量声明的同时对其初始化。
18. 希望把异常向上交给调用这个方法的方法来处理, 可以通过 `throw` 语句来实现。
19. 大数类中静态的 `valueOf` 可实现普通数值向大数的转化。
20. 包的名字放在类文件的开头, 否则为无名包。

### 三、程序阅读题


1. 写出下列程序的输出

```
int[] arr = {2, 3, 1, 5, 4, 6};
Arrays.sort(arr);
int index = Arrays.binarySearch(arr, 3);
System.out.println(index);
```

2. 阅读下面程序写出执行结果

```
int k;
for(k=1; k<=5; k++){
 if(k>4) break;
 System.out.println("k="+k);
}
```

3. 数组是表示一个具有相同数据类型的数据元素的集合。在 Java 语言中, 数组被定义为一个对象, 每个元素相当于该对象的数据成员变量, 数组中的元素可以任何数据类型。请结合下面的程序回答后面的问题。



```
public static void main(String[] args){
 int[][] a = {{1, 2}, {3, 4, 5, 6}, {7, 8, 9}, {}}; // Jagged
 array
 int len = a.length;
 int col1 = a[0].length;
 int col2 = a[1].length;
 int col3 = a[2].length;
 int col4 = a[3].length;

}
```

- (1) 上述程序中变量 len、col1、col2、col3、col4 的值依次是多少?
- (2) 上述程序中 a[1][1]、a[1][2]、a[1][3] 的值依次是多少?
- (3) 使用 for 循环语句编写一个程序片段实现对数组 a 中的所有元素进行求和。

4. 反转排序是以相反的顺序把原有数组的内容重新排序, 其基本思想是把数组最后一个元素与第一个元素替换, 倒数第二个元素与第二个元素替换, 依此类推, 直到把所有数组元素反转替换。请结合下面的反转排序的程序回答后面的问题。

```
public class Sorter {
 public static void main(String[] args) {
 int[] arr = {10, 20, 30, 40, 50, 60};
 Sorter sorter = new Sorter();
 sorter.sort(arr);
 System.out.println(Arrays.toString(arr));
 }

 public void sort(int[] p) {
 int temp;
 int len = p.length;
 for (int i = 0; i < len / 2; i++) {
 _____; // Line 1
 _____; // Line 2
 _____; // Line 3
 }
 }
}
```

- (1) 上述 for 循环语句中需填入 3 行代码, 该部分是反转排序的关键步骤, 请完善程序  
(2) 上述程序执行后, 最终程序的输出是?

5. 已知一个类的定义如下, 请根据该类的定义回答后面的问题。

```
import java.util.*;

public class Pair<T extends AbstractList<Integer> & List<Integer> &
RandomAccess, U> {
 private T first;
 private U second;

 public Pair(T first, U second) {
 this.first = first;
 this.second = second;
 }

 public T getFirst() {
 return first;
 }

 public U getSecond() {
 return second;
 }

 public void setFirst(T newValue) { // Corrected assignment
 first = newValue;
 }

 public void setSecond(U newValue) {
 second = newValue;
 }
}
```

- (1) 该类用到了 Java 中的什么机制?
- (2) 在类中 "T extends AbstractList<Integer> & List<Integer> & RandomAccess" 和 "U" 表示含义分别是什么?
- (3) 在类的定义中有 AbstractList<Integer>、List<Integer>、RandomAccess 三个类型限定, 这三个限定是 T 必须满足的要求。哪些可以调? 哪些不可以调?
- (4) 按下列方式使用该类创建对象是否合法:
  - (a) `Pair<ArrayList<Integer>, Double> t = new Pair<ArrayList<Integer>, Double>();`
  - (b) `Pair<ArrayList<String>, Integer> t = new Pair<ArrayList<String>, Integer>();`
  - (c) `Pair<ArrayList<Integer>, Float> t = new Pair<ArrayList<Integer>, Float>();`

## 四、程序设计题

### 1. 完成下面两个小题

#### 1. 请设计并编写一个同时满足下列所有需求的抽象类:

- (1) 该类是抽象类, 类名为 `Shapes`, 其所在的包名为 `oop.core.base`, 该类可以通过 `import` 语句被其他包中的类所访问到;
- (2) 该类中有两个 `int` 类型的成员变量, 变量名分别为 `width` 和 `height`, 这两个变量可以被同一个包中的类以及该类的所有子类访问到;
- (3) 该类含有一个构造方法, 该构造方法需传进去两个 `int` 类型的参数, 参数名称分别为 `width`, `height`, 该构造方法使用 `this` 关键字实现了对成员变量 `width` 和 `height` 的初始化;
- (4) 该类中含有一个方法名为 `getArea` 的抽象方法, 该方法无输入参数且其返回值为 `double` 类型;
- (5) 该类中含有一个方法名为 `getPerimeter` 的抽象方法, 该方法无输入参数且其返回值为 `double` 类型。

#### 2. 请设计并编写一个同时满足下列所有需求的类。

- (1) 该类的类名为 `Square`, 其包名为 `oop.core`, 该类可以通过 `import` 语句被其他包中的类所访问到, 该类继承了第 1 题中的抽象类 `Shapes`;
- (2) 该类有一个构造方法, 该构造方法含有两个 `int` 型输入参数, 参数名称分别为 `width`, `height`, 该构造方法通过 `super` 关键字实现了对父类成员变量的初始化;
- (3) 该类对其父类的 `getArea` 方法进行了实现, 要求返回面积, 即计算 `width * height` 的值;
- (4) 该类对其父类的 `getPerimeter` 方法进行了实现, 要求返回周长, 即  $2 * (width + height)$  的值。

### 2. 完成下面两个小题



1. 请设计并编写一个满足下列所有需求的接口。
  - (1) 定义一个 public 接口, 名为 Shapes, 其所在的包名为 oop.core.base;
  - (2) 定义返回值为 double 类型并且无输入参数的 public 抽象方法, 其中方法名为 getArea;
  - (3) 定义返回值为 double 类型并且无输入参数的 public 抽象方法, 其中方法名为 getPerimeter。
  
2. 请设计并编写一个同时满足下列所有需求的类。
  - (1) 定义一个 public 类 Square, 其包名为 oop.core, 且该类实现了第 1 题中的接口 Shapes;
  - (2) 在类 Square 中定义两个 double 类型的 public 成员变量, 其中成员变量名为: width, height;
  - (3) 用带有两个形式参数的 public 构造方法对 Square 类中的成员变量进行初始化, 其中形式参数名为: width, height;
  - (4) 在 Square 类中对 getArea 方法进行实现, 返回面积, 即 width \* height;
  - (5) 在 Square 类中对 getPerimeter 方法进行实现, 返回周长, 即 2 \* (width + height)。
  
3. 在第 1 题和第 2 题的基础之上, 仔细阅读程序, 写出下列程序的输出结果。

```
package oop.core.test;
import oop.core.Square;
public class SquareTest {
 public static void main(String[] args) {
 Square squ = new Square(3, 4);
 double area = squ.getArea();
 double peri = squ.getPerimeter();
 System.out.println("area = " + area + "peri =" + peri);}
}
```

## 5 参考答案

### 5.1 选择题

1. 正确答案：A

1. A. javac.exe

- javac 是 Java 编译器的命令
- 正确。

2. B. javadoc.exe

- javadoc 是生成 API 文档的工具
- 错误。

3. C. java.exe

- java 是 Java 程序运行的命令
- 错误。

4. D. javaw.exe

- javaw 是无控制台窗口运行 Java 程序的命令
- 错误。

2. 正确答案：D

1. A. interface

- interface 是定义接口的关键字
- 错误。

2. B. final

- final 是定义常量的关键字
- 错误。

3. C. abstract

- abstract 是定义抽象类的关键字
- 错误。

4. D. class

- class 是定义类的关键字
- 正确。

3. 正确答案：D

1. A. HellWorldottrue3/2

- str.append("Hello") -> str = "Hello"
- str.append(true) -> str = "Hellottrue"(Java 会把 true 自动转为字符串 "true")
- str.append(((double)3/2)) -> str = "Hellottrue1.5"

- str.insert(4,"World") -> str = "HellWorldottrue1.5"

- 错误。

4. 正确答案：B

1. A. String 对象

- String 对象不能与 System.in 关联
- 错误。

2. B. Scanner 对象

- Scanner 对象可以与 System.in 关联
- 正确。

3. C. Writer 对象

- Writer 对象不能与 System.in 关联
- 错误。

4. D. Reader 对象

- Reader 对象可以与 System.in 关联
- 错误。

5. 正确答案：B

1. A.

- 这是文档注释格式，用于生成 API 文档
- 错误。

2. B.

- 这是单行注释的正确格式
- 正确。

3. C. % int b=0;

- % 不是 Java 中的注释符号
- 错误。

4. D.

- 这是多行注释，但含有语法错误（冒号）
- 错误。

6. 正确答案：D

1. A. Manager 类是超类

- Manager 类是从 Employee 类派生的子类
- 错误。

2. B. Manager 类是父类
  - Manager 类是从 Employee 类派生的子类
  - 错误。
3. C. Employee 类是子类
  - Employee 类是 Manager 类的父类/超类
  - 错误。
4. D. Manager 类是派生类
  - Manager 类是从 Employee 类派生的子类
  - 正确。

7. **正确答案: A**

1. A. BigInteger a = BigInteger.valueOf(8)+BigInteger.valueOf("7"); 正确。
  - BigInteger 不支持直接使用 + 运算符, 因为它是一个对象类型而不是基本数据类型
  - 需要使用 add() 方法进行加法运算, 如 bigInt1.add(bigInt2)
  - 错误。
2. B. BigInteger a = BigInteger.valueOf(8);
  - 正确的语法
  - 正确。
3. C. BigInteger a = BigInteger.valueOf(Long.parseLong("7"));
  - valueOf 方法接受 long 类型参数, 但 Long.parseLong("7") 会将其变成 long 值 7。
  - 正确。
4. D. BigInteger a = BigInteger.valueOf(8).add(new BigInteger("7"));
  - 正确的语法
  - 正确。

8. **正确答案: D**

1. A. Manager 类

- new Employee("张三", 40000) 创建的是 Employee 对象
- 错误。

2. B. Manager 类或 Employee 类

- getSalary() 方法是 Employee 类的方法
- 错误。

3. C. Manager 类和 Employee 类

- getSalary() 方法是 Employee 类的方法
- 错误。

4. D. Employee 类

- getSalary() 方法是 Employee 类的方法

9. **正确答案: A**

1. A. double[] m = new double[-4];
  - 数组大小不能为负数
  - 错误。
2. B. String[7] m = new String[];
  - 数组声明和初始化语法错误
  - 错误。
3. C. boolean m[] = new boolean[6];
  - 正确的语法
  - 正确。
4. D. int m; m = new double[3];
  - 类型不匹配, int 不能赋值为 double 数组
  - 错误。

| 声明方式       | 说明      | 推荐程度 |
|------------|---------|------|
| double[] m | Java 风格 | 更为推荐 |
| double m[] | C/C++风格 | 不推荐  |

表 1 数组声明语法对比

10. **正确答案: C**

1. A. 'abc'
  - 单引号用于字符(char), 不能表示字符串

- 错误。
- 2. B. {'a','b','c'}
  - 花括号用于数组初始化
  - 错误。
- 3. C. "abc"
  - 双引号用于字符串
  - 正确。
- 4. D. {"abe", "abc", "abe"}
  - 花括号用于数组初始化
  - 错误。

11. **正确答案：D**

1. A. byte
  - 占用内存空间最小
  - 错误。
2. B. float
  - 占用内存空间较小
  - 错误。
3. C. long
  - 占用内存空间较大
  - 错误。
4. D. double
  - 占用内存空间最大
  - 正确。

12. **正确答案：C**

Java 的对象初始化过程分为两个层次：

👉 类级别（一次）：

1. 静态属性（static 字段）
2. 静态代码块

👉 这些在类加载阶段执行，仅执行一次，不管创建多少对象。

👉 对象级别（每次 new）：

3. 非静态属性初始化
4. 非静态代码块
5. 构造方法

👉 每次 new 一个对象时都会走一遍。🔧 构造方法 🧠 所以完整流程是：

1. 类被加载（第一次使用类时） • 执行：静态变量初始化 + 静态代码块（一次）
2. 每次创建对象时（new） • 执行：非静态变量初始化 + 非静态代码块（按顺序） → 构造方法体

13. **正确答案：A**

4. D. 15
  - --b 是先减少 1 再比较

14. **正确答案：B**

1. A. 异常处理代码无异常发生
  - finally 块无论是否发生异常都会执行
  - 错误。
2. B. 异常处理代码中执行了 System.exit(1) 语句
  - System.exit() 会立即终止 JVM，不会执行 finally 块
  - 正确。
3. C. 异常处理代码中执行了 continue 语句
  - continue 语句不会阻止 finally 块执行
  - 错误。
4. D. 异常处理代码中执行了 break 语句
  - break 语句不会阻止 finally 块执行
  - 错误。

15. **正确答案：C**

1. A. 用户输入错误
  - 用户输入错误可能导致异常
  - 错误。
2. B. 设备错误
  - 设备错误可能导致异常
  - 错误。
3. C. 代码错误
  - ⚠️（有争议，但本题视为“属于异常”）
  - 错误。
4. D. 数据类型自动转换

- Java 中的自动类型转换（如 `int` → `long`, `float` → `double`）是正常语言机制，不会抛异常。
- 正常。

16. **正确答案：B**

包内可见，不可以用 `public` 修饰符

1. **A. `package haut.oop.base; public class Employee { }`**
  - `public` 修饰符使类可见于所有包
  - 错误。
2. **B. `package haut.oop.base; class Employee{ }`**
  - 默认修饰符使类仅在同一包内可见
  - 正确。
3. **C. `public class Employee{ }`**
  - `public` 修饰符使类可见于所有包
  - 错误。
4. **D. `class Employee { }`**
  - 默认修饰符使类仅在同一包内可见
  - 错误。

17. **正确答案：C**

3. **C. `-3`**
  - `x = 3`
  - `x = x += x -= x *= x`

可以从右到左进行计算

  - 正确。

18. **正确答案：A**

1. **• `sort()` 方法对数组进行升序排序**
  - `Arrays.toString()` 方法将数组转换为字符串
  - 若是直接打印数组，会打印数组的引用地址
4. **D. (31, 30, 29, 28, 25)**
  - 错误的格式
  - 错误。

19. **正确答案：A**

1. **A. Scanner 对象**

- Scanner 对象可以与 `System.in` 关联
- 正确。

2. **B. Reader 对象**

- Reader 对象可以与 `System.in` 关联
- 错误。

3. **C. String 对象**

- String 对象不能与 `System.in` 关联
- 错误。

4. **D. Writer 对象**

- Writer 对象不能与 `System.in` 关联
- 错误。

20. **正确答案：B**

1. **A. 方法的定义或实现**

- 定义和实现是同一概念
- 错误。

2. **B. 方法的覆盖或重写**

- 子类覆盖父类的方法
- 正确。

3. **C. 方法的重构或实现**

- 重构是代码结构调整，不是覆盖
- 错误。

4. **D. 方法的继承或派生**

- 继承是类之间的关系，不是方法覆盖
- 错误。

## 5.2 判断题

1. **正确答案：对**

类是相同行为和状态的诸多对象的统称。

2. **正确答案：错**

在对一个 `long` 型变量进行赋值时，数字后面要加 `L` 或 `l`。

3. **正确答案：对**

如果类中没有任何构造方法，系统会自动创建一个不带参数的构造方法。

4. **正确答案：对**

`BigInteger` 类可以实现任意精度的整数运算。

5. **正确答案：错**

运算符 && 和 & 是没有区别的。

6. **正确答案：对**

无参数构造器创建的对象，对象的状态被为适当的默认值。

7. **正确答案：对**

通配符的限定若用 `<? extends Manager>` 来表示，其含义这个通配符限制为 `Manager` 类及该类的所有父类型。

8. **正确答案：对**

`import static` 语句可以导入静态方法和静态字段。

9. **正确答案：错**

在 Java 语言中是不支持单重继承的，但是可以使用接口机制来实现多重继承的功能。

10. **正确答案：对**

在语句块内部可以访问之前在外部声明的变量。

11. **正确答案：对**

静态成员属于类，不属于任何单个的对象。

12. **正确答案：错**

若循环体含有多条语句，则必须置于一对花括号中，否则视为语法错误。

13. **正确答案：错**

`Collections`（注意末尾的 's'）是一个工具类（utility class），它提供了操作或返回集合的静态方法。

`Collection`（没有 's'）才是一个接口，是集合层次结构的根接口。

14. **正确答案：对**

子类对父类进行方法覆盖时，需要返回值类型、函数名和参数列表都一模一样。

但是，从 Java 5 开始，返回值类型可以是父类方法返回值类型的子类型（这称为协变返回类型，covariant return types）。如果严格要求“一模一样”，那么由于协变返回类型的存在，此说法不完全正确。具体以老师为准。

15. **正确答案：对**

这两种是创建和运行线程的主要方式。通常推荐实现 `Runnable` 接口，因为它更灵活（避免了 Java 单继承的限制，并且能更好地分离任务和执行机制）。

16. **正确答案：对**

用链接存储结构存储的线性表称为链表。

17. **正确答案：对**

可以在变量声明的同时对其初始化。

18. **正确答案：对**

希望把异常向上交给调用这个方法的方法来处理，可以通过 `throw` 语句来实现。

19. **正确答案：对**

例如 `BigInteger.valueOf(long val)` 和 `BigDecimal.valueOf(double val)` 可以将基本类型的数值转换为相应的 `BigInteger` 或 `BigDecimal` 对象。

20. **正确答案：对**

如果一个 Java 源文件属于某个包，那么 `package` 声明语句必须是文件中除去注释和空白行的第一条语句。如果没有 `package` 声明，则该文件中的类属于无名包（unnamed package）。

## 5.3 程序阅读题

1. 我们一步一步分析这段 Java 程序的输出：

1. Arrays.sort(arr) 对数组排序: [2, 3, 1, 5, 4, 6] -> [1, 2, 3, 4, 5, 6]
2. Arrays.binarySearch(arr, 3) 在排序后数组中查找元素 3: 索引: 0 1 2 3 4 5 值: 1  
2 3 4 5 6 找到 3 在索引 2 处
3. 程序输出: 2

2. 详细过程如下:

- |                                                          |       |
|----------------------------------------------------------|-------|
| 1. $k = 1 \rightarrow 1 > 4$ ❌ $\rightarrow$ 打印 $k=1$    | 输出:   |
| 2. $k = 2 \rightarrow 2 > 4$ ❌ $\rightarrow$ 打印 $k=2$    | $k=1$ |
| 3. $k = 3 \rightarrow 3 > 4$ ❌ $\rightarrow$ 打印 $k=3$    | $k=2$ |
| 4. $k = 4 \rightarrow 4 > 4$ ❌ $\rightarrow$ 打印 $k=4$    | $k=3$ |
| 5. $k = 5 \rightarrow 5 > 4$ ✅ $\rightarrow$ break, 循环结束 | $k=4$ |

3. 首先看数组定义:

```
int[][] a = {
 {1, 2}, // a[0], 长度 2
 {3, 4, 5, 6}, // a[1], 长度 4
 {7, 8, 9}, // a[2], 长度 3
 {} // a[3], 长度 0
};
```

那么:

- $len = a.length = 4$  (a 有 4 个一维数组元素)
- $col1 = a[0].length = 2$
- $col2 = a[1].length = 4$
- $col3 = a[2].length = 3$
- $col4 = a[3].length = 0$

✅ 答案:  $len = 4, col1 = 2, col2 = 4, col3 = 3, col4 = 0$

1.  $a[1][1]$ 、 $a[1][2]$ 、 $a[1][3]$  的值依次是多少?

查看  $a[1] = \{3, 4, 5, 6\}$ :

- $a[1][1] = 4$
- $a[1][2] = 5$
- $a[1][3] = 6$

✅ 答案: 依次为 4、5、6

2. 使用 for 循环语句实现对数组 a 中的所有元素进行求和

|                                                                                                                                                                                                   |                                                                                                                                                            |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>//normal for 循环 int sum = 0; for (int i = 0; i &lt; a.length; i++) {     for (int j = 0; j &lt; a[i].length; j++) {         sum += a[i][j];     } } System.out.println("总和是: " + sum);</pre> | <pre>// 也可以用增强型 `for` 循环 int sum = 0; for (int[] row : a) {     for (int val : row) {         sum += val;     } } System.out.println("总和是: " + sum);</pre> |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|

在这个具体数组中，所有元素之和为：

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45$$

4. 填入的代码如下:

```
public void sort(int[] p) {
 int temp;
 int len = p.length;
 for (int i = 0; i < len / 2; i++) {
 temp = p[i];
 p[i] = p[len - i - 1]; //数组 index 从0开始
 p[len - i - 1] = temp;
 }
}
```

输出的结果为:

```
[60, 50, 40, 30, 20, 10]
```



5. 解析:

1. 泛型参数  $T$  的边界:

- 类型参数  $T$  有一个复杂的 **上界 (Upper Bound)**。它使用了 `extends` 关键字和 `&` 连接符来指定多个约束。这些约束可以总结如下:

| 约束                                            | 含义                                                 | 关联类型示例                                |
|-----------------------------------------------|----------------------------------------------------|---------------------------------------|
| <code>extends ArrayList&lt;Integer&gt;</code> | $T$ 必须继承 <code>ArrayList&lt;Integer&gt;</code> 类   | <code>ArrayList</code>                |
| <code>&amp; List&lt;Integer&gt;</code>        | $T$ 必须实现 <code>List&lt;Integer&gt;</code> 接口       | <code>List</code>                     |
| <code>&amp; RandomAccess</code>               | $T$ 必须实现 <code>RandomAccess</code> 标记接口，表示支持高效随机访问 | <code>RandomAccess</code>             |
| 综合要求                                          | $T$ 必须同时满足以上所有条件                                   | <code>ArrayList&lt;Integer&gt;</code> |

表 2 参数解析

  - 综合来看， $T$  被限制为必须是一种支持高效随机访问的、存储 `Integer` 元素的列表实现，例如 `ArrayList<Integer>` 就是一个满足条件的典型例子。而像 `LinkedList<Integer>` 则不满足 `RandomAccess` 条件。

2. 泛型参数  $U$ :

- 类型参数  $U$  没有指定任何边界 (`extends ...`)，这意味着  $U$  可以是任何 Java 类型（包括原始类型的包装类，如 `Integer`, `Double`，或其他自定义类，如 `String`, `MyObject` 等）。

1. 用的是 Java 的泛型系统.

2. 如表 2 所示

3. `ArrayList<Integer>`、`List<Integer>`可以, `RandomAccess` 不可以

4. (ac) 合法, (b)不合法

6. 答案: