第十章 异常处理

# 选择题

1. 以下不是使用try-catch进行异常处理的机制和意义的是( B )。
2. 将发生异常和处理异常进行分离
3. 使程序能在发生异常时停止执行
4. 被调方法将异常通知给主调方法
5. 主调方法接收到异常后处理异常
6. 以下通常属于异常处理应用范围的是( D )。
7. 处理编译错误
8. 处理逻辑错误
9. 处理程序运行中的致命错误Error
10. 处理程序运行中的异常Exception
11. 关于异常类，以下说法错误的是( B )。
12. 异常类都是Throwable的子类
13. 不能自定义异常类
14. 异常类有参数为String型异常信息字符串的构造方法和无参数构造方法
15. 异常类有getMessage()和toString()方法返回异常信息
16. 下列关于try-catch-finally结构的描述中，错误的是( C )。
17. try块中是可能抛出异常的代码，若某条语句抛出了异常，则其后续语句不再继续执行，而是转到catch进行异常类型匹配
18. 一个try块可以对应多个catch块，用于对多个异常类进行捕获
19. 一个try块只能对应一个catch块，但可对应多个finally块
20. finally语句块一般总会执行
21. 下列关于try和catch的描述中错误的是( A )。
22. 不同的catch代码段是不同的作用域，但可以互相访问其中的局部变量
23. 如果没有异常产生，则所有的catch代码段都被略过不执行
24. try代码段后至少要有一个catch代码段
25. 异常总是由距离产生异常位置最近的匹配的catch代码段处理
26. 下列关于catch的描述中错误的是( D )。
27. 一个catch后的()内可有多个用|分隔的异常类捕获多种异常并做统一处理
28. 如果catch后的()内声明捕获父类异常，则该catch块可以捕获子类型异常
29. catch块中也可以抛出异常，甚至可以将捕获到的异常再次抛出
30. 如果两个catch块要捕获的异常有继承关系，捕获父类异常的catch块应在前面
31. 以下关于异常处理的说法中错误的是( A )。
32. 用try-catch进行异常处理不可以嵌套
33. 在特殊情况下，try-catch-finally结构中即使发生异常，finally块也可能不被执行
34. finally块可有可无
35. 如果需要用一个catch快捕获所有异常，则参数类型可以为Exception
36. 以下程序运行时抛出( C )类型的异常。

String s = null;

System.out.println(s.length());

1. ArithmeticException
2. EOFException
3. NullPointerException
4. IOException
5. 以下程序运行时抛出( D )类型的异常。

int a[] = new int[3];

a[3] = 0;

1. ArithmeticException
2. EOFException
3. NullPointerException
4. ArrayIndexOutOfBoundsException
5. 以下程序运行时抛出( A )类型的异常。

String s = "abc";

System.out.println(s.charAt(3));

1. StringIndexOutOfBoundsException
2. EOFException
3. NullPointerException
4. IOException
5. 以下程序运行时抛出( A )类型的异常。

int a = 1/0;

1. ArithmeticException
2. EOFException
3. NullPointerException
4. ArrayIndexOutOfBoundsException
5. 以下程序运行时输入a，则会抛出( C )类型的异常。

Scanner scan = new Scanner(System.in);

int a = Integer.parseInt(scan.nextLine());

1. ArithmeticException
2. InputMismatchException
3. NumberFormatException
4. ArithmeticException
5. 语句System.out.println(Double.parseDouble(" 3 4 "));的运行结果是( B )。
6. 34
7. 抛出NumberFormatException异常
8. 抛出InputMismatchException异常
9. 3
10. 以下程序的运行结果是( D )。

System.out.println(Math.sqrt(-1));

1. 抛出ArithmeticException类型异常
2. 0
3. i
4. NaN
5. 在try-catch-finally结构中，无论是否发生异常，( C )块中的代码通常都会被执行。
6. try
7. catch
8. finally
9. throw
10. 自定义的异常类通常继承( A )。
11. Exception
12. Throwable
13. Error
14. RuntimeException
15. 以下异常中，方法即使不处理也不必声明抛出的是( D )及其子类。
16. Exception
17. Error
18. RuntimeException
19. B和C
20. 以下程序的运行结果是( D )。

public class Test {

public static void main(String args[]) {

try {

System.out.println(1 / 0);

} catch (Exception e) {

System.out.println("Exception");

} catch (ArithmeticException ae) {

System.out.println("ArithmeticException");

}

}

}

1. NaN
2. Exception
3. ArithmeticException
4. 编译错误
5. 调用异常类对象的( D )方法可以获取异常信息字符串。
6. toString()
7. getMessage()
8. getLocalizedMessage()
9. 以上方法都可以
10. 如果方法体可能抛出IOException类异常，则以下说法正确的是( D )。
11. 方法如果不处理此异常，则必须声明抛出此异常
12. 方法如果不声明抛出此异常，则必须用try-catch处理异常
13. 如果方法声明抛出此异常，则调用这个方法的方法也必须处理或声明此异常
14. 以上说法都正确