Exception Handling

For more details on SUN Certifications, visit JavaScjpDumps

Q: 01 Given:

```
11. public static void parse(String str) {
12. try {
13. float f = Float.parseFloat(str);
14. } catch (NumberFormatException nfe) {
15. f = 0;
16. } finally {
17. System.out.println(f);
18. }
19. }
20. public static void main(String[] args) {
21. parse("invalid");
22. }
```

What is the result?

A. 0.0
B. Compilation fails.
C. A ParseException is thrown by the parse method at runtime.
D. A NumberFormatException is thrown by the parse method at runtime.

Answer: B

Q: 02 Given:

```
84. try {
85. ResourceConnection con = resourceFactory.getConnection();
86. Results r = con.query("GET INFO FROM CUSTOMER");
87. info = r.getData();
88. con.close();
89. } catch (ResourceException re) {
90. errorLog.write(re.getMessage());
91. }
92. return info;
```

Which statement is true if a ResourceException is thrown on line 86?

A. Line 92 will not execute.
B. The connection will not be retrieved in line 85.
C. The resource connection will not be closed on line 88.
D. The enclosing method will throw an exception to its caller.

Answer: C

Q: 03 Given:

```
31. // some code here
32. try {
33. // some code here
34. } catch (SomeException se) {
35. // some code here
36. } finally {
37. // some code here
38. }
```

Under which three circumstances will the code on line 37 be executed? (Choose three.)

A. The instance gets garbage collected.
B. The code on line 33 throws an exception.
C. The code on line 35 throws an exception.
D. The code on line 31 throws an exception.
E. The code on line 33 executes successfully.

Answer: B, C, E

Q: 04 Given:

```
11. class A {
12. public void process() { System.out.print("A,"); }
13. class B extends A {
14. public void process() throws IOException {
15. super.process();
16. System.out.print("B,");
17. throw new IOException();
18. }
19. public static void main(String[] args) {
20. try { new B().process(); }
21. catch (IOException e) { System.out.println("Exception"); }}
```

What is the result?

A. Exception
B. A,B,Exception
C. Compilation fails because of an error in line 20.
D. Compilation fails because of an error in line 14.
E. A NullPointerException is thrown at runtime.

Answer: D

Q: 05 Given:

```
11. static void test() throws Error {
12. if (true) throw new AssertionError();
13. System.out.print("test ");
14. }
15. public static void main(String[] args) {
```

```
try { test(); }
catch (Exception ex) { System.out.print("exception "); }
System.out.print("end ");
}
```

What is the result?

A. end

B. Compilation fails.

C. exception end

D. exception test end

E. A Throwable is thrown by main.

F. An Exception is thrown by main.

Answer: E

Q: 06 Given:

```
Float pi = new Float(3.14f);
if (pi > 3) {
System.out.print("pi is bigger than 3. ");
}
else {
System.out.print("pi is not bigger than 3. ");
}
finally {
System.out.println("Have a nice day.");
}
```

What is the result?

A. Compilation fails.

B. pi is bigger than 3.

C. An exception occurs at runtime.

D. pi is bigger than 3. Have a nice day.

E. pi is not bigger than 3. Have a nice day.

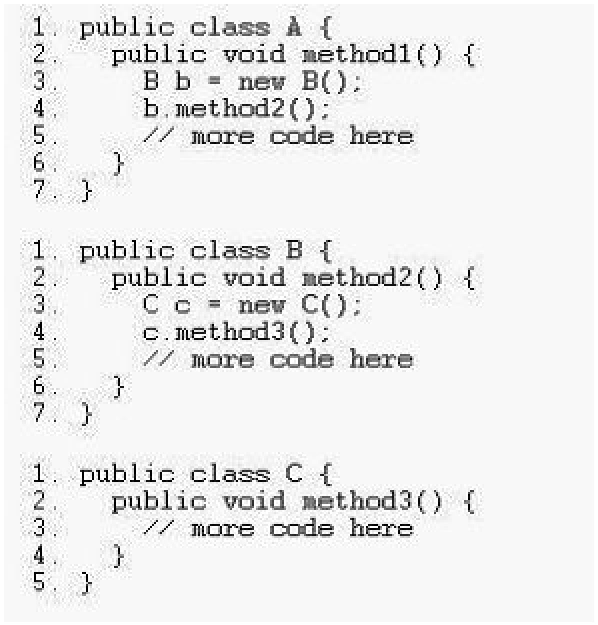
Answer: A

Q: 07 Given:

```
public static void main(String[] args) {
try {
args = null;
args[0] = "test";
System.out.println(args[0]);
} catch (Exception ex) {
System.out.println("Exception");
} catch (NullPointerException npe) {
System.out.println("NullPointerException");
}
```

21. }
What is the result?
A. test
B. Exception
C. Compilation fails.
D. NullPointerException
Answer: C

Q:08 Click the Exhibit button.
Given:
25. try {
26. A a = new A();
27. a.method1();
28. } catch (Exception e) {
29. System.out.print("an error occurred");
30. }
Which two statements are true if a NullPointerException is thrown on line 3 of class C? (Choose two.)



```
1. public class A {
2.   public void method1() {
3.     B b = new B();
4.     b.method2();
5.     // more code here
6.   }
7. }

1. public class B {
2.   public void method2() {
3.     C c = new C();
4.     c.method3();
5.     // more code here
6.   }
7. }

1. public class C {
2.   public void method3() {
3.     // more code here
4.   }
5. }
```

A. The application will crash.
B. The code on line 29 will be executed.
C. The code on line 5 of class A will execute.
D. The code on line 5 of class B will execute.
E. The exception will be propagated back to line 27.
Answer: B, E

Q:09 Given:

```
11. static void test() throws RuntimeException {
12. try {
13. System.out.print("test ");
14. throw new RuntimeException();
15. }
16. catch (Exception ex) { System.out.print("exception "); }
17. }
18. public static void main(String[] args) {
19. try { test(); }
20. catch (RuntimeException ex) { System.out.print("runtime "); }
21. System.out.print("end ");
22. }
```

What is the result?

A. test end
B. Compilation fails.
C. test runtime end
D. test exception end
E. A Throwable is thrown by main at runtime.

Answer: D

Q:10 Given:

```
33. try {
34. // some code here
35. } catch (NullPointerException e1) {
36. System.out.print("a");
37. } catch (RuntimeException e2) {
38. System.out.print("b");
39. } finally {
40. System.out.print("c");
41. }
```

What is the result if a NullPointerException occurs on line 34?

A. c
B. a
C. ab
D. ac
E. bc
F. abc

Answer: D

Q:11 Given:

```
10. public class Foo {
11. static int[] a;
12. static { a[0]=2; }
public static void main( String[] args ) {}
}
```

Which exception or error will be thrown when a programmer attempts to run this code?

A. java.lang.StackOverflowError
B. java.lang.IllegalStateException
C. java.lang.ExceptionInInitializerError
D. java.lang.ArrayIndexOutOfBoundsException

Answer: C

Q: 12 Given:

```
static void test() {
try {
String x = null;
System.out.print(x.toString() + " ");
}
finally { System.out.print("finally "); }
}
public static void main(String[] args) {
try { test(); }
catch (Exception ex) { System.out.print("exception "); }
}
```

What is the result?

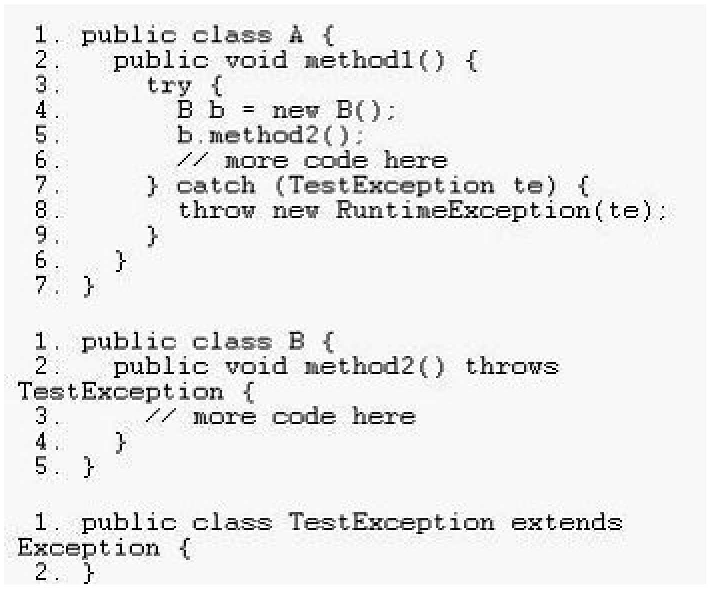
A. null
B. finally
C. null finally
D. Compilation fails.
E. finally exception

Answer: E

Q: 13 Click the Exhibit button.
Given:

```
public void method() {
A a = new A();
a.method1();
}
```

Which statement is true if a TestException is thrown on line 3 of class B?



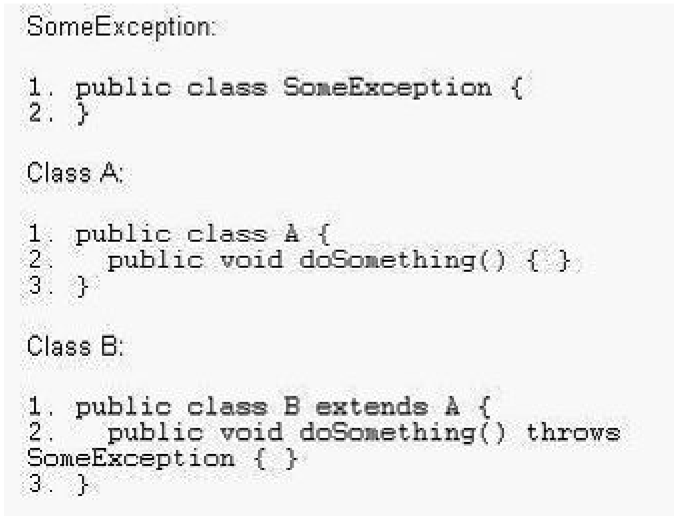
```
 1. public class A {
 2.   public void method1() {
 3.     try {
 4.       B b = new B();
 5.       b.method2();
 6.       // more code here
 7.     } catch (TestException te) {
 8.       throw new RuntimeException(te);
 9.     }
 6.   }
 7. }

 1. public class B {
 2.   public void method2() throws
TestException {
 3.     // more code here
 4.   }
 5. }

 1. public class TestException extends
Exception {
 2. }
```

A. Line 33 must be called within a try block.
B. The exception thrown by method1 in class A is not required to be caught.
C. The method declared on line 31 must be declared to throw a RuntimeException.
D. On line 5 of class A, the call to method2 of class B does not need to be placed in a try/catch block.
Answer: B

Q: 14 Click the Exhibit button.
Which statement is true about the two classes?



```
SomeException:

public class SomeException {
}

Class A:

public class A {
  public void doSomething() { }
}

Class B:

public class B extends A {
  public void doSomething() throws
SomeException { }
}
```

A. Compilation of both classes will fail.
B. Compilation of both classes will succeed.
C. Compilation of class A will fail. Compilation of class B will succeed.
D. Compilation of class B will fail. Compilation of class A will succeed.
Answer: D

Question: 15
Click the Exhibit button.
Class TestException

```
1. public class TestException extends Exception {
2. }
```

Class A:

```
1. public class A {
2.
3. public String sayHello(String name) throws TestException {
4.
5. if(name == null) {
6. throw new TestException();
7. }
8.
9. return “Hello “+ name;
10. }
11.
12. }
```

A programmer wants to use this code in an application:

```
45. A a=new A();
46. System.out.println(a.sayHello(”John”));
```

Which two are true? (Choose two.)
A. Class A will not compile.
B. Line 46 can throw the unchecked exception TestException.
C. Line *45* can throw the unchecked exception TestException.
D. Line 46 will compile if the enclosing method throws a TestException.
E. Line 46 will compile if enclosed in a try block, where TestException is caught**.**
Answer: DE

Question:16
Given:

```
11.classA {
12. public void process() { System.out.print(”A “); } }
13. class B extends A {
14. public void process() throws RuntimeException {
15. super.process();
16. if (true) throw new RuntimeException();
17. System.out.print(“B”); }}
18. public static void main(String[] args) {
19. try { ((A)new B()).process(); }
20. catch (Exception e) { System.out.print(”Exception “); }
21. }
```

What is the result?

A. Exception
B. A Exception
C. A Exception B
D. A B Exception
E. Compilation fails because of an error in line 14.
F. Compilation fails because of an error in line 19.

Answer: B

17. Given:

```
import java.io.*;
class Master {
String doFileStuff() throws FileNotFoundException { return "a"; }
}
class Slave extends Master {
public static void main(String[] args) {
String s = null;
try { s = new Slave().doFileStuff();
} catch ( Exception x) {
s = "b"; }
System.out.println(s);
}
// insert code here
}
```

Which, inserted independently at // insert code here, will compile, and produce the output b? (Choose all that apply.)

A. String doFileStuff() { return "b"; }
B. String doFileStuff() throws IOException { return "b"; }
C. String doFileStuff(int x) throws IOException { return "b"; }
D. String doFileStuff() throws FileNotFoundException { return "b"; }
E. String doFileStuff() throws NumberFormatException { return "b"; }
F. String doFileStuff() throws NumberFormatException,
FileNotFoundException { return "b"; }

Answer:

-> A , D, E, and **F** are correct. It's okay for an overriding method to throw the same exceptions, narrower exceptions, or no exceptions. And it's okay for the overriding method to throw any runtime exceptions.

-> **B** is incorrect, because the overriding method is trying to throw a broader exception.
C is incorrect. This method doesn't override, so the output is a.

18. Given:

```
class Input {
```

```
public static void main(String[] args) {
String s = "-";
try {
doMath(args[0]);
s += "t "; // line 6
}
finally { System.out.println(s += "f "); }
}
public static void doMath(String a) {
int y = 7 / Integer.parseInt(a);
} }
```

And the command-line invocations:

```
java Input
java Input 0
```

Which are true? (Choose all that apply.)

A. Line 6 is executed exactly 0 times.
B. Line 6 is executed exactly 1 time.
C. Line 6 is executed exactly 2 times.
D. The finally block is executed exactly 0 times.
E. The finally block is executed exactly 1 time.
F. The finally block is executed exactly 2 times.
G. Both invocations produce the same exceptions.
H. Each invocation produces a different exception.

Answer:

-> **A , F,** and **H** are correct. Since both invocations throw exceptions, line 6 is never reached. Since both exceptions occurred within a try block, the finally block will always execute. The first invocation throws an ArrayIndexOutOfBoundsException, and the second invocation throws an ArithmeticException for the attempt to divide by zero.

-> **B, C, D, E,** and **G** are incorrect based on the above.

19. Given:

```
class Plane {
static String s = "-";
public static void main(String[] args) {
new Plane().s1();
System.out.println(s);
}
void s1() {
try { s2(); }
catch (Exception e) { s += "c"; }
}
void s2() throws Exception {
s3(); s += "2";
```

s3(); s += "2b";
}
void s3() throws Exception {
throw new Exception();
} }
What is the result?
A. -
B. -c
C. -c2
D. -2c
E. -c22b
F. -2c2b
G. -2c2bc
H. Compilation fails.
Answer:
-> **B** is correct. Once s3() throws the exception to s2(), s2() throws it to s1(), and no more of s2()'s code will be executed.
-> **A, C, D, E, F, G,** and **H** are incorrect based on the above.

20. Given:
try { int x = Integer.parseInt("two"); }
Which could be used to create an appropriate catch block? (Choose all that apply.)
A. ClassCastException
B. IllegalStateException
C. NumberFormatException
D. IllegalArgumentException
E. ExceptionInInitializerError
F. ArrayIndexOutOfBoundsException
Answer:
-> **C** and **D** are correct. Integer.parseInt can throw a NumberFormatException, and IllegalArgumentException is its superclass (i.e., a broader exception).
-> **A, B, E,** and **F** are not in NumberFormatException's class hierarchy.

21. Given:
1. class Ping extends Utils {
2. public static void main(String [] args) {
3. Utils u = new Ping();
4. System.out.print(u.getInt(args[0]));
5. }
6. int getInt(String arg) {
7. return Integer.parseInt(arg);
8. }
9. }

```
10. class Utils {
11. int getInt(String x) throws Exception { return 7; }
12. }
```

And the following three possible changes:
C1. Declare that main() throws an Exception.
C2. Declare that Ping.getInt() throws an Exception.
C3. Wrap the invocation of getInt() in a try / catch block.
Which change(s) allow the code to compile? (Choose all that apply.)

A. Just C1 is sufficient.
B. Just C2 is sufficient.
C. Just C3 is sufficient.
D. Both C1 and C2 are required.
E. Both C1 and C3 are required.
F. Both C2 and C3 are required.
G. All three changes are required.

Answer:
-> **A** and **C** are correct. Remember that line 4 is making a polymorphic call so the compiler knows that an exception might be thrown. If C1 is implemented the exception has been sufficiently declared, and if C3 is implemented the exception has been sufficiently handled. C2 is not necessary in either case.
-> **B, D, E, F,** and **G** are incorrect based on the above.

22. Given:

```
class Emu {
static String s = "-";
public static void main(String[] args) {
try {
throw new Exception();
} catch (Exception e) {
try {
try { throw new Exception();
} catch (Exception ex) { s += "ic "; }
throw new Exception(); }
catch (Exception x) { s += "mc "; }
finally { s += "mf "; }
} finally { s += "of "; }
System.out.println(s);
} }
```

What is the result?

A. -ic of
B. -mf of
C. -mc mf
D. -ic mf of

E. -ic mc mf of
F. -ic mc of mf
G. Compilation fails.
Answer:
-> **E** is correct. There is no problem nesting try / catch blocks. As is normal, when an exception is thrown, the code in the catch block runs, then the code in the finally block runs.
-> **A, B, C, D,** and **F** are incorrect based on the above.

23. Given:
```
class Mineral { }
class Gem extends Mineral { }
class Miner {
static int x = 7;
static String s = null;
public static void getWeight(Mineral m) {
int y = 0 / x;
System.out.print(s + " ");
}
public static void main(String[] args) {
Mineral[] ma = {new Mineral(), new Gem()};
for(Object o : ma)
getWeight((Mineral) o);
}
}
```
And the command-line invocation:
```
java Miner.java
```
What is the result?
A. null
B. null null
C. A ClassCastException is thrown.
D. A NullPointerException is thrown.
E. A NoClassDefFoundError is thrown.
F. An ArithmeticException is thrown.
G. An IllegalArgumentException is thrown.
H. An ArrayIndexOutOfBoundsException is thrown.
Answer:
-> **E** is correct. The invocation should be java Miner, in which case null null would be produced.
-> **A, B, C, D, F, G,** and **H** are incorrect based on the above.

24. Which are most typically thrown by an API developer or an application developer as opposed to being thrown by the JVM? (Choose all that apply.)
A. ClassCastException
B. IllegalStateException

C. NumberFormatException
D. IllegalArgumentException
E. ExceptionInInitializerError
Answer:
-> **B , C,** and **D** are correct. **B** is typically used to report an environment problem such as trying to access a resource that's closed. **C** is often thrown in API methods that attempt
to convert poorly formed String arguments to numeric values. **D** is often thrown in API methods that receive poorly formed arguments.
-> **A** and **E** are thrown by the JVM.