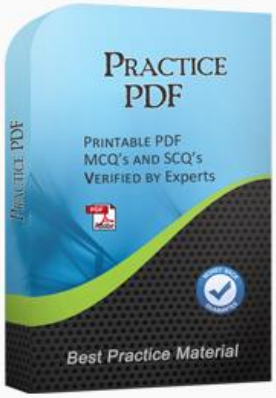
SurePassExams

SurePassExams

- Online Tool, Convenient, easy to study.
- Instant Online Access
- Supports All Web Browsers
- Practice Online Anytime
- Test History and Performance Review
- Supports Windows / Mac / Android / iOS, etc.

- Installable Software Application
- Simulates Real Exam Environment
- Builds Exam Confidence
- Supports MS Operating System
- Two Modes For Practice
- Practice Offline Anytime



- Printable PDF Format
- Prepared by IT Experts
- Instant Access to Download
- Study Anywhere, Anytime
- 365 Days Free Updates
- Free PDF Demo Available

Security & Privacy

We respect customer privacy. We use McAfee's security service to provide you with utmost security for your personal information & peace of mind.

365 Days Free Updates

Free update is available within 365 days after your purchase. After 365 days, you will get 50% discounts for updating.

Money Back Guarantee

Full refund if you fail the corresponding exam in 90 days after purchasing. And Free get any another product.

Instant Download

After Payment, our system will send you the products you purchase in mailbox in a minute after payment. If not received within 2 hours, please contact us.

http://www.surepassexams.com/

Legal & authoritative company offering the highest pass-rate Exam Torrent materials and helping use pass for sure.

Exam : **1z1-803**

Title : Java SE 7 Programmer I

Vendor : Oracle

Version : DEMO

NO.1 Given the code fragment: What is the result?

```
String color = "Red";

switch (color) {
    case "Red":
        System.out.println("Found Red");
    case "Blue":
        System.out.println("Found Blue");
        break;
    case "White":
        System.out.println("Found White");
        break;
    default:
        System.out.println("Found Default");
}
```

A. Found Red

B. Found Red Found Blue

C. Found Red Found Blue Found White

D. Found Red Found Blue Found White Found Default

Answer: B

Explanation:

As there is no break statement after the case "Red" statement the case Blue statement will run as well.

Note: The body of a switch statement is known as a switch block. A statement in the switch block can be labeled with one or more case or default labels. The switch statement evaluates its expression, then executes all statements that follow the matching case label.

Each break statement terminates the enclosing switch statement. Control flow continues with the first statement following the switch block. The break statements are necessary because without them, statements in switch blocks fall through: All statements after the matching case label are executed in sequence, regardless of the expression of subsequent case labels, until a break statement is encountered.

NO.2 Given the code fragment

```
class Test2 {
int fvar;
static int cvar;
    public static void main(String[] args) {
    Test2 t = new Test2();
    // insert code here to write field variables
     }
    }
```

Which code fragments, inserted independently, enable the code compile?

A. t.fvar = 200;

B. cvar = 400;

C. fvar = 200; cvar = 400;

D. this.fvar = 200; this.cvar = 400;

E. t.fvar = 200; Test2.cvar = 400;
F. this.fvar = 200; Test2.cvar = 400;
Answer: ABE

NO.3 Given:
public class X {
static int i;
int j;
public static void main(String[] args) {
X x1 = new X();
X x2 = new X();
x1.i = 3;
x1.j = 4;
x2.i = 5;
x2.j = 6;
System.out.println(
x1.i + " "+
x1.j + " "+
x2.i + " "+
x2.j);
} } What is the result?
A. 3 4 5 6
B. 3 4 3 6
C. 5 4 5 6
D. 3 6 4 6
Answer: C

NO.4 Given:

```
public class DoWhile1 {
    public static void main(String[] args) {
        int ii = 2;
        do {
            System.out.println(ii);
        } while (--ii);
    }
}
```

What is the result?
A. 2 1
B. 2 1 0
C. null
D. an infinite loop
E. compilation fails
Answer: E
Explanation:

The line while (--ii); will cause the compilation to fail. ii is not a boolean value.
A correct line would be while (--ii>0);

NO.5 Given the code fragment:

```
int[][] array2D = { {0,1,2}, {3,4,5,6} };

System.out.print(array2D[0].length + " ");
System.out.print(array2D[1].getClass().isArray() + " ")
System.out.println(array2D[0][1]);
```

What is the result?

A. 3 false 1

B. 2 true 3

C. 2 false 3

D. 3 true 1

E. 3 false 3

F. 2 true 1

G. 2 false 1

Answer: D

Explanation:
The length of the element with index 0, {0, 1, 2}, is 3. Output: 3 The element with index 1, {3, 4, 5, 6}, is of type array. Output: true The element with index 0, {0, 1, 2} has the element with index 1: 1. Output: 1

NO.6 Given:

```
public class Test {

    static void dispResult(int[] num) {
        try {
            System.out.println(num[1] / (num[1] - num[2]));
        } catch(ArithmeticException e) {
            System.err.println("first exception");
        }
        System.out.println("Done");
    }

    public static void main(String[] args) {
        try {
            int[] arr = {100, 100};
            dispResult(arr);
        } catch(IllegalArgumentException e) {
            System.err.println("second exception");
        } catch(Exception e) {
            System.err.println("third exception");
        }
    }
}
```

What is the result?

A. 0 Done

B. First Exception Done

C. Second Exception
D. Done Third Exception
E. Third Exception
Answer: E

NO.7 Given the code fragment:

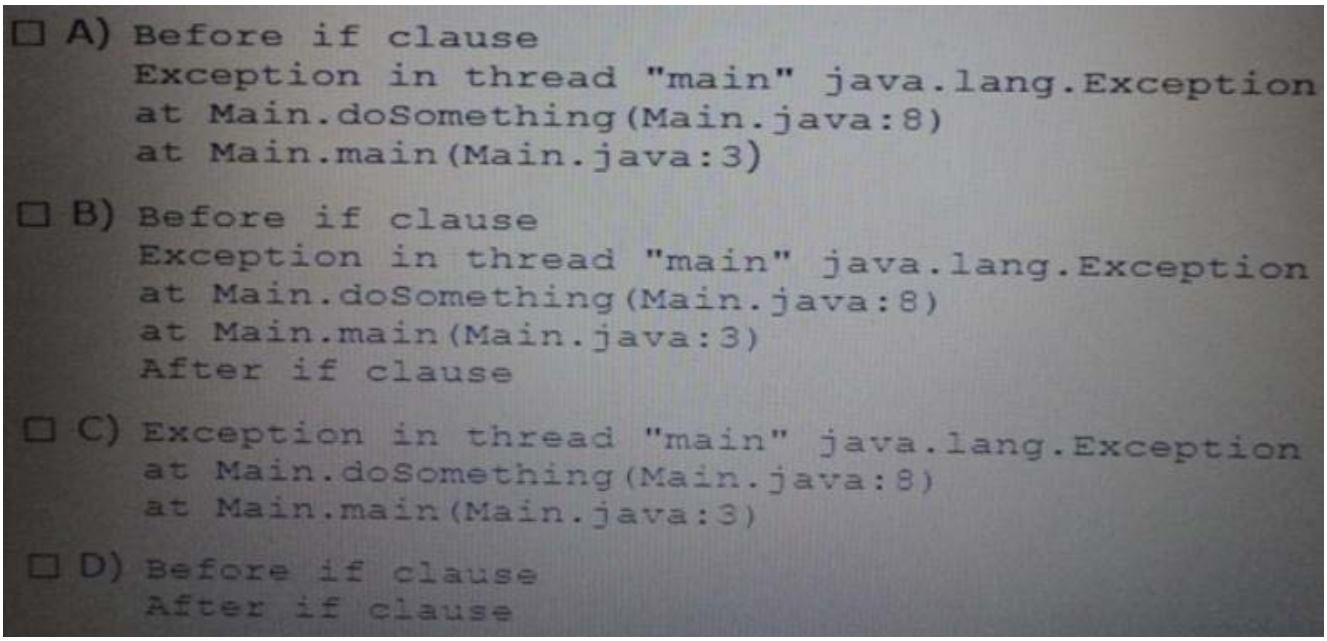
```
interface Contract{ }
class Super implements Contract{ }
class Sub extends Super {}

public class Ref {
    public static void main(String[] args) {
        List objs = new ArrayList();

        Contract c1 = new Super();
        Contract c2 = new Sub();                    // line n1
        Super s1 = new Sub();

        objs.add(c1);
        objs.add(c2);                               // line n2
        objs.add(s1);

        for(Object itm: objs) {
            System.out.println(itm.getClass().getName());
        }
    }
}
```

A. Super Sub Sub
B. Contract Contract Super
C. Compilation fails at line n1
D. Compilation fails at line n2
Answer: A

NO.8 Given:

```
public class Main {
    public static void main(String[] args) throws Exception {
        doSomething();
    }
    private static void doSomething() throws Exception {
        System.out.println("Before if clause");
        if (Math.random() > 0.5) {
            throw new Exception();
        }
        System.out.println("After if clause");
    }
}
```

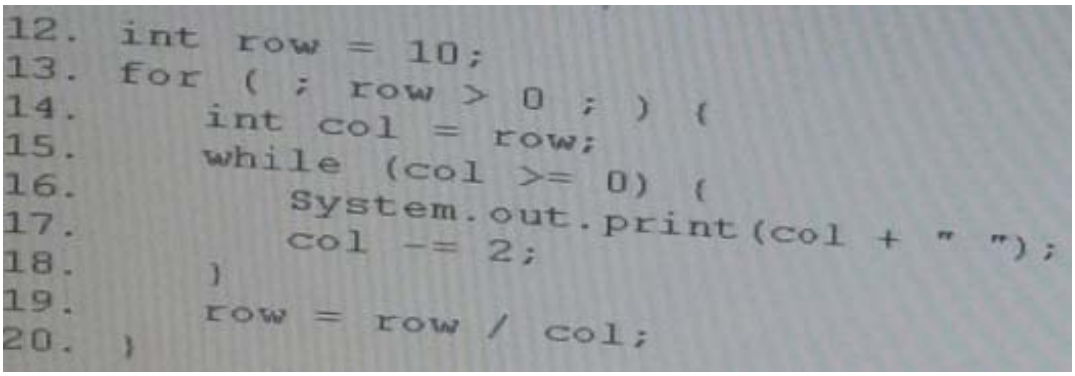
Which two are possible outputs?



A. Option A
B. Option B
C. Option C
D. Option D
Answer: A,D
Explanation:
The first println statement, System.out.println("Before if clause");, will always run.
If Math.Random() > 0.5 then there is an exception. The exception message is displayed and the program terminates.
If Math.Random() > 0.5 is false, then the second println statement runs as well.

NO.9 Which three are advantages of the Java exception mechanism?
A. Improves the program structure because the error handling code is separated from the normal program function
B. Provides a set of standard exceptions that covers all the possible errors
C. Improves the program structure because the programmer can choose where to handle exceptions
D. Improves the program structure because exceptions must be handled in the method in which they occurred
E. allows the creation of new exceptions that are tailored to the particular program being
Answer: A,C,E
Explanation:
A: The error handling is separated from the normal program logic.
C: You have some choice where to handle the exceptions.
E: You can create your own exceptions.

NO.10 Given the code fragment:



```
12. int row = 10;
13. for ( ; row > 0 ; ) {
14.     int col = row;
15.     while (col >= 0) {
16.         System.out.print(col + " ");
17.         col -= 2;
18.     }
19.     row = row / col;
20. }
```

What is the result?

A. 10 8 6 4 2 0

B. 10 8 6 4 2

C. AnArithmeticException is thrown at runtime

D. The program goes into an infinite loop outputting: 10 8 6 4 2 0. . .

E. Compilation fails

Answer: A

NO.11 Given:

```
class Cake { int model; String flavor; Cake() { model = 0; flavor = "Unknown"; }
} public class Test {
public static void main(String[] args) {
Cake c = new Cake();
bake1(c);
System.out.println(c.model + " " + c.flavor);
bake2(c);
System.out.println(c.model + " " + c.flavor);
}
public static Cake bake1(Cake c) {
c.flavor = "Strawberry";
c.model = 1200;
return c;
}
public static void bake2(Cake c) {
c.flavor = "Chocolate";
c.model = 1230;
return;
} }
```

What is the result?

A. 0 unknown 0 unknown

B. 1200 Strawberry 1200 Strawberry

C. 1200 Strawberry 1230 Chocolate

D. Compilation fails

Answer: C

Explanation:

1200 Strawberry

1230 Chocolate

NO.12 Given:

```
1. import java.io.Error;
2.      public class TestApp {
3.      public static void main(String[] args) {
4.           TestApp t = new TestApp();
5.           try {
6.                t.doPrint();
7.                t.doList();
8.
9.           } catch (Exception e2) {
10.               System.out.println("Caught " + e2);
11.          }
12.     }
13.     public void doList() throws Exception {
14.          throw new Error("Error");
15.     }
16.     public void doPrint() throws Exception {
17.          throw new RuntimeException("Exception");
18.     }
19. }
```

What is the result?

```
A) Caught java.lang.RuntimeException: Exception
   Exception in thread "main" java.lang.Error: Error
        at TestApp.doList(TestApp.java: 14)
        at TestApp.main(TestApp.java: 6)

B) Exception in thread "main" java.lang.Error: Error
        at TestApp.doList(TestApp.java: 14)
        at TestApp.main(TestApp.java: 6)

C) Caught java.lang.RuntimeException: Exception
   Caught java.lang.Error: Error

D) Caught java.lang.RuntimeException: Exception
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer: D

NO.13 Given the code fragment:

```
int j=0, k=0;

for(int i=0; i < x; i++) {
    do {
        k = 0;
        while (k < z){
            k++;
            System.out.print(k + " ");
        }
        System.out.println(" ");
        j++;
    } while (j < y);
    System.out.println("---");
}
```

What values of x, y, z will produce the following result?

1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4

A. X = 4, Y = 3, Z = 2
B. X = 3, Y = 2, Z = 3
C. X = 2, Y = 3, Z = 3
D. X = 4, Y = 2, Z = 3
E. X = 2, Y = 3, Z = 4

Answer: E

Explanation:

Z is for the innermost loop. Should print 1 2 3 4. So Z must be 4.
Y is for the middle loop. Should print three lines of 1 2 3 4. So Y must be set 3.
X is for the outmost loop. Should print 2 lines of. So X should be 2.

NO.14 Given: What is the result?

```
public class Test {
  public static void main(String[] args) {
    Test ts = new Test();
    System.out.print(isAvailable + " ");
    isAvailable= ts.doStuff();
    System.out.println(isAvailable);
  }
  public static boolean doStuff() {
    return !isAvailable;
  }
  static boolean isAvailable = false;
}
```

A. true true
B. true false
C. false true

D. false false
E. Compilation fails
Answer: C

NO.15 View the exhibit:
```
public class Student {
public String name = "";
public int age = 0;
public String major = "Undeclared";
public boolean fulltime = true;
public void display() {
System.out.println("Name: " + name + " Major: " + major); }
public boolean isFullTime() {
return fulltime;
}
}
```
Given:
```
Public class TestStudent {
public static void main(String[] args) {
Student bob = new Student ();
bob.name = "Bob";
bob.age = 18;
bob.year = 1982;
}
}
```
What is the result?
A. year is set to 1982.
B. bob.year is set to 1982
C. A runtime error is generated.
D. A compile time error is generated.
Answer: D

NO.16 Given:
```
public class MyClass { public static void main(String[] args) { while (int ii = 0; ii < 2) { ii++;
System.out.println("ii = " + ii); } }
}
```
What is the result?
A. ii = 1 ii = 2
B. Compilation fails
C. The program prints nothing
D. The program goes into an infinite loop with no output
E. The program goes to an infinite loop outputting: ii = 1 ii = 1
Answer: B
Explanation:
The while statement is incorrect. It has the syntax of a for statement.

The while statement continually executes a block of statements while a particular condition is true. Its syntax can be expressed as:
while (expression) {
statement(s)
}
The while statement evaluates expression, which must return a boolean value. If the expression evaluates to true, the while statement executes the statement(s) in the while block. The while statement continues testing the expression and executing its block until the expression evaluates to false.
Reference: The while and do-while Statements

NO.17 Given the code fragment:

```
String[] colors = {"red", "blue", "green", "yellow","maroon","cyan"};
```

Which code fragment prints blue, cyan, ?

```
A) for (String c:colors){
       if (c.length() != 4) {
           continue;
       }
   System.out.print(c+", ");
   }

B) for (String c:colors[]) {
       if (c.length() <= 4) {
           continue;
       }
   System.out.print(c+", ");
   }

C) for (String c:String[] colors) {
       if (c.length() >= 3) {
           continue;
       }
   System.out.print(c+", ");
   }

D) for (String c:colors){
       if (c.length() != 4) {
           System.out.print(c+", ");
           continue;
       }
   }
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

NO.18 Given:

```
abstract class X {
    public abstract void methodX();
}
interface Y{
    public void methodY();
}
```

Which two code fragments are valid?

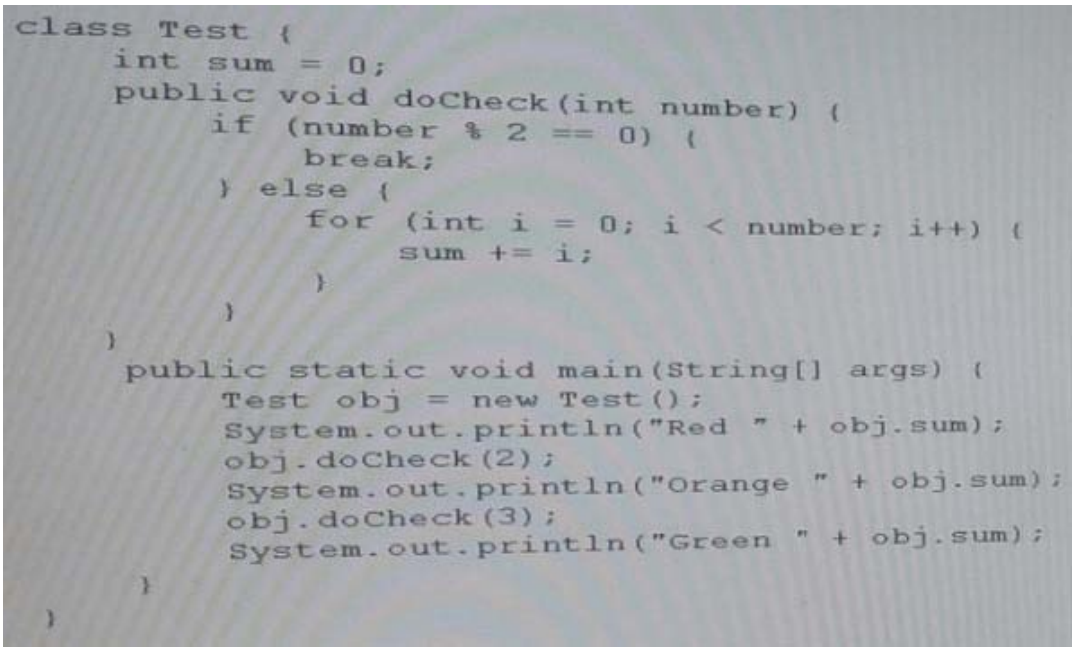
```
☐ A) class Z extends X implements Y{
        public void methodZ(){}
     }

☐ B) abstract class Z extends X implements Y{
        public void methodZ(){}
     }

☐ C) class Z extends X implements Y{
        public void methodX(){}
     }

☐ D) abstract class Z extends X implements Y{
     }

☐ E) class Z extends X implements Y{
        public void methodY(){}
     }
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E

Answer: B,C

Explanation:
When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class (C). However, if it does not, then the subclass must also be declared abstract (B). Note: An abstract class is a class that is declared abstract-it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

NO.19 Given: What is the result?



```
class Test {
    int sum = 0;
    public void doCheck(int number) {
        if (number % 2 == 0) {
            break;
        } else {
            for (int i = 0; i < number; i++) {
                sum += i;
            }
        }
    }
    public static void main(String[] args) {
        Test obj = new Test();
        System.out.println("Red " + obj.sum);
        obj.doCheck(2);
        System.out.println("Orange " + obj.sum);
        obj.doCheck(3);
        System.out.println("Green " + obj.sum);
    }
}
```

A. Red 0 Orange 0 Green 3
B. Red 0 Orange 0 Green 6
C. Red 0 Orange 1
D. Green 4
E. Compilation fails
Answer: E

NO.20 Given:
public class String1 {
public static void main(String[] args) {
String s = "123";
if (s.length() >2)
s.concat("456");
for(int x = 0; x <3; x++)
s += "x";
System.out.println(s);
} }
What is the result?
A. 123
B. 123xxx
C. 123456
D. 123456xxx
E. Compilation fails
Answer: B
Explanation:
123xxx
The if clause is not applied.
Note: Syntax of if-statement:
if (Statement) {

```
}
```