

```
1. public class Threads5 {
2. public static void main (String[] args) {
3. new Thread(new Runnable() {
4. public void run() {
5. System.out.print("bar");
6. }).start();
7. }
8. }
```

What is the result?

- A.** Compilation fails.
- B.** An exception is thrown at runtime.
- C.** The code executes normally and prints "bar".
- D.** The code executes normally, but nothing prints.

Answer: C

```
public class TestOne implements Runnable {
2. public static void main (String[] args) throws Exception {
3. Thread t = new Thread(new TestOne());
4. t.start();
5. System.out.print("Started");
6. t.join();
7. System.out.print("Complete");
8. }
9. public void run() {
10. for (int i = 0; i < 4; i++) {
11. System.out.print(i);
12. }
13. }
14. }
```

What can be a result?

- A.** Compilation fails.

- B. An exception is thrown at runtime.
- C. The code executes and prints "StartedComplete".
- D. The code executes and prints "StartedComplete0123".
- E. The code executes and prints "Started0123Complete".

Answer: E

```
10. public class Starter extends Thread {
11.     private int x = 2;
12.     public static void main(String[] args) throws
Exception {
13.         new Starter().makeItSo();
14.     }
15.     public Starter() {
16.         x = 5;
17.         start();
18.     }
19.     public void makeItSo() throws Exception {
20.         join();
21.         x = x - 1;
22.         System.out.println(x);
23.     }
24.     public void run() { x *= 2; }
25. }
```

- A. 4
- B. 5
- C. 8
- D. 9
- E. Compilation fails.
- F. An exception is thrown at runtime.
- G. It is impossible to determine for certain.

Answer: D

```
public class TestFive {
2. private int x;
3. public void foo() {
4.     int current = x;
5.     x = current + 1;
6. }
```

```

7. public void go() {
8. for(int i = 0; i < 5; i++) {
9. new Thread() {
10. public void run() {
11. foo();
12. System.out.print(x + ", ");
13. } }.start();
14. }}

```

Which two changes, taken together, would guarantee the output: 1, 2, 3, 4, 5, ? (Choose two.)

- A.** move the line 12 print statement into the foo() method
- B.** change line 7 to public synchronized void go() {
- C.** change the variable declaration on line 2 to private volatile int x;

Oracle 1z0-851: Practice Exam

"Pass Any Exam. Any Time." - www.actualtests.com 41

- D.** wrap the code inside the foo() method with a synchronized(this) block
- E.** wrap the for loop code inside the go() method with a synchronized block synchronized(this) { // for loop code here }

Answer: A,D

```

public class Threads2 implements Runnable {
2.
3. public void run() {
4. System.out.println("run.");
5. throw new RuntimeException("Problem");
6. }
7. public static void main(String[] args) {
8. Thread t = new Thread(new Threads2());
9. t.start();

```

```
10. System.out.println("End of method.");  
11. }  
12. }
```

Which two can be results? (Choose two.)

A. java.lang.RuntimeException: Problem

B. run.

java.lang.RuntimeException: Problem

C. End of method.

java.lang.RuntimeException: Problem

D. End of method.

run.

java.lang.RuntimeException: Problem

E. run.

java.lang.RuntimeException: Problem

End of method.

Answer: D,E

Given that t1 is a reference to a live thread, which is true?

A. The Thread.sleep() method can take t1 as an argument.

B. The Object.notify() method can take t1 as an argument.

C. The Thread.yield() method can take t1 as an argument.

D. The Thread.setPriority() method can take t1 as an argument.

E. The Object.notify() method arbitrarily chooses which thread to notify.

Answer: E

Given that **Triangle** implements **Runnable**, and:

```
31. void go() throws Exception {
```

```
32. Thread t = new Thread(new Triangle());
```

```
33. t.start();
34. for(int x = 1; x < 100000; x++) {
35. //insert code here
36. if(x%100 == 0) System.out.print("g");
37. } }
38. public void run() {
39. try {
40. for(int x = 1; x < 100000; x++) {
41. // insert the same code here
42. if(x%100 == 0) System.out.print("t");
43. }
44. } catch (Exception e) { }
45. }
```

Which two statements, inserted independently at both lines 35 and 41, tend to allow both threads to temporarily pause and allow the other thread to execute? (Choose two.)

- A. Thread.wait();
- B. Thread.join();
- C. Thread.yield();
- D. Thread.sleep(1);
- E. Thread.notify();

Answer: C,D

```
public class Threads3 implements Runnable {
2. public void run() {
3. System.out.print("running");
4. }
5. public static void main(String[] args) {
6. Thread t = new Thread(new Threads3());
7. t.run();
8. t.run();
```

```

9. t.start();
10. }
11. }

```

What is the result?

- A.** Compilation fails.
- B.** An exception is thrown at runtime.
- C.** The code executes and prints "running".
- D.** The code executes and prints "runningrunning".
- E.** The code executes and prints "runningrunningrunning".

Answer: E

```

public class PingPong implements Runnable {
12. synchronized void hit(long n) {
13. for(int i = 1; i < 3; i++)
14. System.out.print(n + "-" + i + " ");
15. }
16. public static void main(String[] args) {
17. new Thread(new PingPong()).start();
18. new Thread(new PingPong()).start();
19. }
20. public void run() {
21. hit(Thread.currentThread().getId());
22. }
23. }

```

Which two statements are true? (Choose two.)

- A.** The output could be 8-1 7-2 8-2 7-1
- B.** The output could be 7-1 7-2 8-1 6-1
- C.** The output could be 8-1 7-1 7-2 8-2
- D.** The output could be 8-1 8-2 7-1 7-2

Answer: C,D

```

interface A { void x(); }

```

```
11. class B implements A { public void x() {} public void y()
{} }
12. class C extends B { public void x() {} } And:
20. java.util.List<A> list = new java.util.ArrayList<A>();
21. list.add(new B());
22. list.add(new C());
23. for (A a : list) {
24. a.x();
25. a.y();
26. }
```

What is the result?

- A. The code runs with no output.
- B. An exception is thrown at runtime.
- C. Compilation fails because of an error in line 20.
- D. Compilation fails because of an error in line 21.
- E. Compilation fails because of an error in line 23.
- F. Compilation fails because of an error in line 25.

Answer: F

```
1. public class TestOne {
2. public static void main (String[] args) throws Exception {
3. Thread.sleep(3000);
4. System.out.println("sleep");
5. }
6. }
```

What is the result?

- A. Compilation fails.
- B. An exception is thrown at runtime.
- C. The code executes normally and prints "sleep".
- D. The code executes normally, but nothing is printed.

Answer: C

```
1. public class Threads4 {
```

```
2. public static void main (String[] args) {
3.     new Threads4().go();
4. }
5. public void go() {
6.     Runnable r = new Runnable() {
7.     public void run() {
8.         System.out.print("foo");
9.     }
10. };
11. Thread t = new Thread(r);
12. t.start();
13. t.start();
14. }
15. }
```

What is the result?

- A.** Compilation fails.
- B.** An exception is thrown at runtime.
- C.** The code executes normally and prints "foo".
- D.** The code executes normally, but nothing is printed.

Answer: B

Which two statements are true? (Choose two.)

- A.** It is possible for more than two threads to deadlock at once.
- B.** The JVM implementation guarantees that multiple threads cannot enter into a deadlocked state.
- C.** Deadlocked threads release once their sleep() method's sleep duration has expired.
- D.** Deadlocking can occur only when the wait(), notify(), and notifyAll() methods are used incorrectly.

E. It is possible for a single-threaded application to deadlock if synchronized blocks are used incorrectly.

F. If a piece of code is capable of deadlocking, you cannot eliminate the possibility of deadlocking by inserting invocations of `Thread.yield()`.

Answer: A,F

Which two scenarios are NOT safe to replace a `StringBuffer` object with a `StringBuilder` object? (Choose two.)

A. When using versions of Java technology earlier than 5.0.

B. When sharing a `StringBuffer` among multiple threads.

C. When using the `java.io` class `StringBufferInputStream`.

D. When you plan to reuse the `StringBuffer` to build more than one string.

Answer: A,B

```
public class Threads4 {
2. public static void main (String[] args) {
3.     new Threads4().go();
4. }
5. public void go() {
6.     Runnable r = new Runnable() {
7.     public void run() {
8.         System.out.print("foo");
9.     }
10. };
11. Thread t = new Thread(r);
12. t.start();
13. t.start();
14. }
```

15. }

What is the result?

- A. Compilation fails.
- B. An exception is thrown at runtime.
- C. The code executes normally and prints "foo".
- D. The code executes normally, but nothing is printed.

Answer: B

```
public class TestOne {  
2. public static void main (String[] args) throws Exception {  
3. Thread.sleep(3000);  
4. System.out.println("sleep");  
5. }  
6. }
```

What is the result?

- A. Compilation fails.
- B. An exception is thrown at runtime.
- C. The code executes normally and prints "sleep".
- D. The code executes normally, but nothing is printed.

Answer: C

```
1. public class TestSeven extends Thread {  
2. private static int x;  
3. public synchronized void doThings() {  
4. int current = x;  
5. current++;  
6. x = current;  
7. }  
8. public void run() {  
9. doThings();  
10. }  
11. }
```

Which statement is true?

- A.** Compilation fails.
- B.** An exception is thrown at runtime.
- C.** Synchronizing the run() method would make the class thread-safe.
- D.** The data in variable "x" are protected from concurrent access problems.
- E.** Declaring the doThings() method as static would make the class thread-safe.
- F.** Wrapping the statements within doThings() in a synchronized(new Object()) { } block would make the class thread-safe.

Answer: E

Which two code fragments will execute the method doStuff() in a separate thread? (Choose two.)

- A.**

```
new Thread() {  
    public void run() { doStuff(); }  
};
```
- B.**

```
new Thread() {  
    public void start() { doStuff(); }  
};
```
- C.**

```
new Thread() {  
    public void start() { doStuff(); }  
}.run();
```
- D.**

```
new Thread() {  
    public void run() { doStuff(); }  
}.start();
```
- E.**

```
new Thread(new Runnable() {  
    public void run() { doStuff(); }  
}).run();
```
- F.**

```
new Thread(new Runnable() {  
    public void run() { doStuff(); }  
}
```

```
}).start();
```

Answer: D,F

Given that Triangle implements Runnable, and:

```
31. void go() throws Exception {  
32. Thread t = new Thread(new Triangle());  
33. t.start();  
34. for(int x = 1; x < 100000; x++) {  
35. //insert code here  
36. if(x%100 == 0) System.out.print("g");  
37. } }  
38. public void run() {  
39. try {  
40. for(int x = 1; x < 100000; x++) {  
41. // insert the same code here  
42. if(x%100 == 0) System.out.print("t");  
43. }  
44. } catch (Exception e) { }  
45. }
```

Which two statements, inserted independently at both lines 35 and 41, tend to allow both threads to temporarily pause and allow the other thread to execute? (Choose two.)

- A. Thread.wait();
- B. Thread.join();
- C. Thread.yield();
- D. Thread.sleep(1);
- E. Thread.notify();

Answer: C,D

Which two code fragments will execute the method doStuff() in a separate thread? (Choose two.)

- A. new Thread() {

```
public void run() { doStuff(); }  
};
```

```
B. new Thread() {  
public void start() { doStuff(); }  
};
```

```
C. new Thread() {  
public void start() { doStuff(); }  
}.run();
```

```
D. new Thread() {  
public void run() { doStuff(); }  
}.start();
```

```
E. new Thread(new Runnable() {  
public void run() { doStuff(); }  
}).run();
```

```
F. new Thread(new Runnable() {  
public void run() { doStuff(); }  
}).start();
```

Answer: D,F

```
Given: public class NamedCounter {  
private final String name;  
private int count;  
public NamedCounter(String name) { this.name = name; }  
public String getName() { return name; }  
public void increment() { count++; }  
public int getCount() { return count; }  
public void reset() { count = 0; }  
}
```

Which three changes should be made to adapt this class to be used safely by multiple threads?

(Choose three.)

A. declare reset() using the synchronized keyword

- B. declare getName() using the synchronized keyword
- C. declare getCount() using the synchronized keyword
- D. declare the constructor using the synchronized keyword
- E. declare increment() using the synchronized keyword

Answer: A,C,E

Given that t1 is a reference to a live thread, which is true?

- A. The Thread.sleep() method can take t1 as an argument.
- B. The Object.notify() method can take t1 as an argument.
- C. The Thread.yield() method can take t1 as an argument.
- D. The Thread.setPriority() method can take t1 as an argument.
- E. The Object.notify() method arbitrarily chooses which thread to notify.

Answer: E

```
10. public class Starter extends Thread {
11.     private int x = 2;
12.     public static void main(String[] args) throws
Exception {
13.         new Starter().makeItSo();
14.     }
15.     public Starter() {
16.         x = 5;
17.         start();
18.     }
19.     public void makeItSo() throws Exception {
20.         join();
21.         x = x - 1;
22.         System.out.println(x);
23.     }
24.     public void run() { x *= 2; }
25. }
```

A. 4

- B.** 5
- C.** 8
- D.** 9
- E.** Compilation fails.
- F.** An exception is thrown at runtime.
- G.** It is impossible to determine for certain.

Answer: D

Which two scenarios are NOT safe to replace a StringBuffer object with a StringBuilder object?
(Choose two.)

- A.** When using versions of Java technology earlier than 5.0.
- B.** When sharing a StringBuffer among multiple threads.
- C.** When using the java.io class StringBufferInputStream.
- D.** When you plan to reuse the StringBuffer to build more than one string.

Answer: A,B
