|  |
| --- |
| Question 1 |
|  | Given: |
|  | 1. 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”. //correct |
|  |  |
|  |  |
|  | Question 2 |
|  | 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(); //compilation error |
|  | D. new Thread() { |
|  | public void run() { doStuff(); } //correct |
|  | } .start(); |
|  | E. new Thread(new Runnable() { |
|  | public void run() { doStuff(); } |
|  | } ).run(); |
|  | F. new Thread(new Runnable() { |
|  | public void run() { doStuff(); } |
|  | }).start(); //correct |
|  |  |
|  | Question 3 |
|  | Given: |
|  | 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. //correct |
|  | C. The code executes normally and prints ‘foo”. |
|  | D. The code executes normally, but nothing is printed. |
|  |  |
|  | Question 4 |
|  | Given: |
|  | 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”. //correct |
|  | D. The code executes normally, but nothing prints. |
|  |  |
|  | Question 5 |
|  | Given: |
|  | 11. Runnable r = new Runnable() { |
|  | 12. public void run() { |
|  | 13. System.out.print(”Cat”); |
|  | 14. } |
|  | 15. }; |
|  | 16. Threadt=new Thread(r) { |
|  | 17. public void run() { |
|  | 18. System.out.print(”Dog”); |
|  | 19. } |
|  | 20. }; |
|  | 21. t.start(); |
|  | What is the result? |
|  | A. Cat |
|  | B. Dog //correct |
|  | C. Compilation fails. |
|  | D. The code runs with no output. |
|  | E. An exception is thrown at runtime. |
|  |  |
|  | Question 6 |
|  | Click the Exhibit button. |
|  | Given: |
|  | 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. } |
|  | What is the output if the main() method is rum? |
|  | A. 4 |
|  | B. 5 |
|  | C. 8 |
|  | D. 9 //correct |
|  | E. Compilation fails. |
|  | F. An exception is thrown at runtime. |
|  | G. It is impossible to determine for certain. |
|  |  |
|  | Question 7 |
|  | Given: |
|  | 1. 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 //correct |
|  | E. run. |
|  | java.lang.RuntimeException: Problem |
|  | End of method. //correct |
|  |  |
|  |  |
|  | Question 8 |
|  | Given: |
|  | 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”. //correct |
|  | D. The code executes normally, but nothing is printed. |
|  |  |
|  | Question 9 |
|  | Given: |
|  | 1. 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 “Started0l23Complete”. //correct |
|  |  |
|  | Question 10 |
|  |  |
|  | Given: |
|  | 1. public class TwoThreads { |
|  | 2 |
|  | 3. private static Object resource = new Object(); |
|  | 4. |
|  | 5. private static void delay(long n) { |
|  | 6. try { Thread.sleep(n); } |
|  | 7. catch (Exception e) { System.out.print(”Error “); } |
|  | 8. } |
|  | 9 |
|  | 10. public static void main(String[] args) { |
|  | 11. System.out.print(”StartMain “); |
|  | 12. new Thread1().start(); |
|  | 13. delay(1000); |
|  | 14. Thread t2 = new Thread2(); |
|  | 15. t2.start(); |
|  | 16. delay(1000); |
|  | 17. t2.interrupt |
|  | 18. delay(1000); |
|  | 19. System.out.print(”EndMain “); |
|  | 20. } |
|  | 21. |
|  | 22. static class Thread 1 extends Thread { |
|  | 23. public void run() { |
|  | 24. synchronized (resource) { |
|  | 25. System.out.print(”Startl “); |
|  | 26. delay(6000) |
|  | 27. System.out.print(”End1 “); |
|  | 28. } |
|  | 29. } |
|  | 30. } |
|  | 31. |
|  | 32. static class Thread2 extends Thread { |
|  | 33. public void run() { |
|  | 34. synchronized (resource) { |
|  | 35. System.out.print(”Start2 “); |
|  | 36. delay(2000); |
|  | 37. System.out.print(”End2 “); |
|  | 38. } |
|  | 39. } |
|  | 40. } |
|  | 41. } |
|  | Assume that sleep(m) executes in exactly m milliseconds, and all other |
|  | code executes in an insignificant amount of time. What is the output if |
|  | the main() method is run? |
|  | A. Compilation fails. |
|  | B. Deadlock occurs. |
|  | C. StartMain Start1 Error EndMain End1 |
|  | D. StartMain Start1 EndMain End1 Start2 End2 |
|  | E. StartMain Start1 Error Start2 EndMain End2 End1 |
|  | F. StartMain Start1 Start2 Error End2 EndMain End1 |
|  | G. StartMain Start1 EndMain End1 Start2 Error End2 //correct |
|  |  |
|  | Question 11 |
|  | 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() { coount++; } |
|  | 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 //correct |
|  | B. declare getName() using the synchronized keyword |
|  | C. declare getCount() using the synchronized keyword //correct |
|  | D. declare the constructor using the synchronized keyword |
|  | E. declare increment() using the synchronized keyword //correct |
|  |  |
|  | Question 12 |
|  | Click the Exhibit button: |
|  | 1. public class Threads 1 { |
|  | 2. intx=0; |
|  | 3. public class Runner implements Runnable { |
|  | 4. public void run() { |
|  | 5. int current = 0; |
|  | 6. for(int=i=0;i<4;i++){ |
|  | 7. current = x; |
|  | 8. System.out.print(current + “, “); |
|  | 9. x = current + 2; |
|  | 10. } |
|  | 11. } |
|  | 12. } |
|  | 13. |
|  | 14. public static void main(String[] args) { |
|  | 15. new Threads1().go(); |
|  | 16. } |
|  | 17. |
|  | 18. public void go() { |
|  | 19. Runnable r1 = new Runner(); |
|  | 20. new Thread(r1).start(); |
|  | 21. new Thread(r1 ).start(); |
|  | 22. } |
|  | 23. } |
|  | Which two are possible results? (Choose two.) |
|  | A. 0, 2, 4, 4, 6, 8, 10, 6, //correct |
|  | B. 0, 2, 4, 6, 8, 10, 2, 4, |
|  | C. 0, 2, 4, 6, 8, 10, 12, 14, //correct |
|  | D. 0, 0, 2, 2, 4, 4, 6, 6, 8, 8, 10, 10, 12, 12, 14, 14, |
|  | E. 0, 2, 4, 6, 8, 10, 12, 14, 0, 2, 4, 6, 8, 10, 12, 14, |
|  |  |
|  | Question 13 |
|  | Click the Exhibit button. |
|  | 1. import java.util.\*; |
|  | 2. |
|  | 3. public class NameList { |
|  | 4. private List names = new ArrayList(); |
|  | 5. public synchronized void add(String name) { names.add(name); } |
|  | 6. public synchronized void printAll() { |
|  | 7. for (int i = 0; i <names.size(); i++) { |
|  | 8. System.out.print(names.get(i) +“ “); |
|  | 9. } |
|  | 10. } |
|  | 11. public static void main(String[] args) { |
|  | 12. final NameList sl = new NameList(); |
|  | 13.for(int i=0;i<2;i++) { |
|  | 14. new Thread() { |
|  | 15. public void ruin() { |
|  | 16. sl.add(”A”); |
|  | 17. sl.add(”B”); |
|  | 18. sl.add(”C”); |
|  | 19. sl.printAll(); |
|  | 20. } |
|  | 21. }.start(); |
|  | 22. } |
|  | 23. } |
|  | 24. } |
|  | Which two statements are true if this class is compiled and run? |
|  | (Choose two.) |
|  | A. An exception may be thrown at runtime. |
|  | B. The code may run with no output, without exiting. |
|  | C. The code may run with no output, exiting normally. |
|  | D. The code may rum with output “A B A B C C “, then exit. |
|  | E. The code may rum with output “A B C A B C A B C “, then exit. |
|  | F. The code may ruin with output “A A A B C A B C C “, then exit. |
|  | G. The code may ruin with output “A B C A A B C A B C “, then exit. |
|  |  |
|  | Question 14 |
|  | Given: |
|  | 1. 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 3 to private volatile int x;. |
|  | 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 }. |
|  |  |
|  | Question 15 |
|  | Which three will compile and run without exception? (Choose three.) |
|  | A. private synchronized Object o; |
|  | B. void go() { |
|  | synchronized() { /\* code here \*/ } |
|  | } |
|  | C. public synchronized void go() { /\* code here \*/ } //correct |
|  | D. private synchronized(this) void go() { /\* code here \*/ } |
|  | E. void go() { |
|  | synchronized(Object.class) { /\* code here \*/ } //correct |
|  | } |
|  | F. void go() { |
|  | Object o = new Object(); |
|  | synchronized(o) { /\* code here \*/ } //correct |
|  | } |