DURGA ONLINE EXAMS

Test Your Knowledge

HOME

171) **Click the Exhibit button.**
Which two are possible results? (Choose two.)

```
1. public class Threads1 {
2.  int x = 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.}
```

1) **0, 2, 4, 4, 6, 8, 10, 6,**
2) **0, 2, 4, 6, 8, 10, 2, 4,**
3) **0, 2, 4, 6, 8, 10, 12, 14,**
4) **0, 0, 2, 2, 4, 4, 6, 6, 8, 8, 10, 10, 12, 12, 14, 14,**
5) **0, 2, 4, 6, 8, 10, 12, 14, 0, 2, 4, 6, 8, 10, 12, 14,**

Your Selected options :: none
Correct Options :: 1, 3

Click Here for Explanation

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

1) **The Thread.sleep() method can take t1 as an argument.**
2) **The Object.notify() method can take t1 as an argument.**
3) **The Thread.yield() method can take t1 as an argument.**
4) **The Thread.setPriority() method can take t1 as an argument.**
5) **The Object.notify() method arbitrarily chooses which thread to notify.**

Your Selected options :: none
Correct Options :: 5

Click Here for Explanation

173) **Given:**

```
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?

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

Your Selected options :: none

Correct Options :: 5

Click Here for Explanation

174) **Given:**

```
11. Runnable r = new Runnable() {
12. public void run() {
13. System.out.print("Cat");
14. }
15. };
16. Thread t = new Thread(r) {
17. public void run() {
18. System.out.print("Dog");
19. }
20. };
21. t.start();
```

What is the result?

1) **Cat**
2) **Dog**
3) **Compilation fails.**
4) **The code runs with no output.**
5) **An exception is thrown at runtime.**

Your Selected options :: none

Correct Options :: 2

Click Here for Explanation

175) **Click the Exhibit button.**
What is the result?

```
1.  class Computation extends Thread {
2.
3.    private int num;
4.    private boolean isComplete;
5.    private int result;
6.
7.    public Computation(int num) { this.num
= num; }
8.
9.    public synchronized void run() {
10.     result = num * 2;
11.     isComplete = true;
12.     notify();
13.   }
14.
15.   public synchronized int getResult() {
16.     while (!isComplete) {
17.       try {
18.         wait();
19.       } catch (InterruptedException e)
{}
20.     }
21.     return result;
22.   }
23.
24.   public static void main(String[] args)
{
25.     Computation[] computations = new
Computation[4];
26.     for (int i = 0; i <
computations.length; i++) {
27.       computations[i] = new
Computation(i);
28.       computations[i].start();
29.     }
30.     for (Computation c : computations)
31.       System.out.print(c.getResult() + "
");
32.   }
33. }
```

1) **The code will deadlock.**

2) **The code may run with no output.**

3) **An exception is thrown at runtime.**

4) **The code may run with output "0 6".**

5) **The code may run with output "2 0 6 4".**

6) **The code may run with output "0 2 4 6".**

Your Selected options :: none

Correct Options :: 6

Click Here for Explanation

176) **Given:**

1. public class MyLogger {
2. private StringBuilder logger = new StringBuuilder();
3. public void log(String message, String user) {
4. logger.append(message);
5. logger.append(user);
6. }
7. }

The programmer must guarantee that a single MyLogger object works properly for a multi-threaded system. How must this code be changed to be thread-safe?

1) **synchronize the log method**

2) **replace StringBuilder with StringBuffer**

3) **replace StringBuilder with just a String object and use the string concatenation (+=) within the log method**

4) **No change is necessary, the current MyLogger code is already thread-safe.**

Your Selected options :: none

Correct Options :: 1

Click Here for Explanation

177) **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?

1) **Compilation fails.**
2) **An exception is thrown at runtime.**
3) **The code executes normally and prints "foo".**
4) **The code executes normally, but nothing is printed.**

Your Selected options :: none

Correct Options :: 2

Click Here for Explanation

178) **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.)

1) **java.lang.RuntimeException: Problem**
2) **run.**
   java.lang.RuntimeException: Problem
3) **End of method.**
   java.lang.RuntimeException: Problem
4) **End of method.**
   run.
   java.lang.RuntimeException: Problem
5) **run.**
   java.lang.RuntimeException: Problem
   End of method.

Your Selected options :: none

Correct Options :: 4, 5

Click Here for Explanation

179) **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?

1) **Compilation fails.**
2) **An exception is thrown at runtime.**
3) **The code executes and prints "running".**
4) **The code executes and prints "runningrunning".**
5) **The code executes and prints "runningrunningrunning".**

Your Selected options :: none

Correct Options :: 5

Click Here for Explanation

180) **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.)

1) **Thread.wait();**
2) **Thread.join();**
3) **Thread.yield();**
4) **Thread.sleep(1);**
5) **Thread.notify();**

Your Selected options :: none

Correct Options :: 3, 4

Click Here for Explanation

« Prev | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | **18** | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | Next »

Total No.of Questions	::	292
Total No.of Answered Questions	::	0
Total No.of Unanswered Questions	::	292
Marks	::	0/292(0%)

feedback :: **feedback@durgajobs.com**