|  |  |
| --- | --- |
| 1. | What will be the output of the program?  public class Test  {  public static void main(String[] args)  {  int x = 0;  assert (x > 0) ? "assertion failed" : "assertion passed" ;  System.out.println("finished");  }  } |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | A. | |  |  | | --- | --- | | finished | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | B. | |  | | --- | | Compiliation fails. | | |  | C. | |  | | --- | | An AssertionError is thrown and finished is output. | | |  | D. | |  | | --- | | An AssertionError is thrown with the message "assertion failed." | |   Your Answer: Option **A**  Correct Answer: Option **B**  Explanation:  Compilation Fails. You can't use the Assert statement in a similar way to the ternary operator. Don't confuse.  Learn more problems on : [Assertions](http://www.indiabix.com/java-programming/assertions/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/assertions/discussion-216) |

|  |  |
| --- | --- |
| 2. | What will be the output of the program?  int i = 1, j = 10;  do  {  if(i > j)  {  break;  }  j--;  } while (++i < 5);  System.out.println("i = " + i + " and j = " + j); |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | i = 6 and j = 5 | | |  | B. | |  | | --- | | i = 5 and j = 5 | | |  | C. | |  | | --- | | i = 6 and j = 4 | | |  | D. | |  |  | | --- | --- | | i = 5 and j = 6 | http://www.indiabix.com/_files/images/website/accept.png | |   Your Answer: Option **D**  Correct Answer: Option **D**  Explanation:  This loop is a do-while loop, which always executes the code block within the block at least once, due to the testing condition being at the end of the loop, rather than at the beginning. This particular loop is exited prematurely if i becomes greater than j.  The order is, test i against j, if bigger, it breaks from the loop, decrements j by one, and then tests the loop condition, where a pre-incremented by one i is tested for being lower than 5. The test is at the end of the loop, so i can reach the value of 5 before it fails. So it goes, start:  1, 10  2, 9  3, 8  4, 7  5, 6 loop condition fails.  Learn more problems on : [Flow Control](http://www.indiabix.com/java-programming/flow-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/flow-control/discussion-99) |

|  |  |
| --- | --- |
| 3. | What will be the output of the program?  int i = 0;  while(1)  {  if(i == 4)  {  break;  }  ++i;  }  System.out.println("i = " + i); |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | i = 0 | | |  | B. | |  | | --- | | i = 3 | | |  | C. | |  | | --- | | i = 4 | | |  | D. | |  |  | | --- | --- | | Compilation fails. | http://www.indiabix.com/_files/images/website/accept.png | |   Your Answer: Option **D**  Correct Answer: Option **D**  Explanation:  Compilation fails because the argument of the while loop, the condition, must be of primitive type boolean. In Java, 1 does not represent the true state of a boolean, rather it is seen as an integer.  Learn more problems on : [Flow Control](http://www.indiabix.com/java-programming/flow-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/flow-control/discussion-98) |

|  |  |
| --- | --- |
| 4. | public class Test2  {  public static int x;  public static int foo(int y)  {  return y \* 2;  }  public static void main(String [] args)  {  int z = 5;  assert z > 0; /\* Line 11 \*/  assert z > 2: foo(z); /\* Line 12 \*/  if ( z < 7 )  assert z > 4; /\* Line 14 \*/  switch (z)  {  case 4: System.out.println("4 ");  case 5: System.out.println("5 ");  default: assert z < 10;  }  if ( z < 10 )  assert z > 4: z++; /\* Line 22 \*/  System.out.println(z);  }  }  which line is an example of an inappropriate use of assertions? |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | Line 11 | | |  | B. | |  |  | | --- | --- | | Line 12 | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | Line 14 | | |  | D. | |  | | --- | | Line 22 | |   Your Answer: Option **B**  Correct Answer: Option **D**  Explanation:  Assert statements should not cause side effects. Line 22 changes the value of z if the assert statement is false.  Option A is fine; a second expression in an assert statement is not required.  Option B is fine because it is perfectly acceptable to call a method with the second expression of an assert statement.  Option C is fine because it is proper to call an assert statement conditionally.  Learn more problems on : [Assertions](http://www.indiabix.com/java-programming/assertions/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/assertions/discussion-226) |

|  |  |
| --- | --- |
| 5. | public void foo( boolean a, boolean b)  {  if( a )  {  System.out.println("A"); /\* Line 5 \*/  }  else if(a && b) /\* Line 7 \*/  {  System.out.println( "A && B");  }  else /\* Line 11 \*/  {  if ( !b )  {  System.out.println( "notB") ;  }  else  {  System.out.println( "ELSE" ) ;  }  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | If a is true and b is true then the output is "A && B" | | |  | B. | |  | | --- | | If a is true and b is false then the output is "notB" | | |  | C. | |  | | --- | | If a is false and b is true then the output is "ELSE" | | |  | D. | |  |  | | --- | --- | | If a is false and b is false then the output is "ELSE" | http://www.indiabix.com/_files/images/website/wrong.gif | |   Your Answer: Option **D**  Correct Answer: Option **C**  Explanation:  Option C is correct. The output is "ELSE". Only when a is false do the output lines after 11 get some chance of executing.  Option A is wrong. The output is "A". When a is true, irrespective of the value of b, only the line 5 output will be executed. The condition at line 7 will never be evaluated (when a is true it will always be trapped by the line 12 condition) therefore the output will never be "A && B".  Option B is wrong. The output is "A". When a is true, irrespective of the value of b, only the line 5 output will be executed.  Option D is wrong. The output is "notB".  Learn more problems on : [Flow Control](http://www.indiabix.com/java-programming/flow-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/flow-control/discussion-82) |

|  |  |
| --- | --- |
| 6. | public class While  {  public void loop()  {  int x= 0;  while ( 1 ) /\* Line 6 \*/  {  System.out.print("x plus one is " + (x + 1)); /\* Line 8 \*/  }  }  }  Which statement is true? |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | There is a syntax error on line 1. | | |  | B. | |  | | --- | | There are syntax errors on lines 1 and 6. | | |  | C. | |  | | --- | | There are syntax errors on lines 1, 6, and 8. | | |  | D. | |  |  | | --- | --- | | There is a syntax error on line 6. | http://www.indiabix.com/_files/images/website/accept.png | |   Your Answer: Option **D**  Correct Answer: Option **D**  Explanation:  Using the integer 1 in the while statement, or any other looping or conditional construct for that matter, will result in a compiler error. This is old C Program syntax, not valid Java.  A, B and C are incorrect because line 1 is valid (Java is case sensitive so While is a valid class name). Line 8 is also valid because an equation may be placed in a String operation as shown.  Learn more problems on : [Flow Control](http://www.indiabix.com/java-programming/flow-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/flow-control/discussion-81) |

|  |  |
| --- | --- |
| 7. | What will be the output of the program?  interface Count  {  short counter = 0;  void countUp();  }  public class TestCount implements Count  {  public static void main(String [] args)  {  TestCount t = new TestCount();  t.countUp();  }  public void countUp()  {  for (int x = 6; x>counter; x--, ++counter) /\* Line 14 \*/  {  System.out.print(" " + counter);  }  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 0 1 2 | | |  | B. | |  |  | | --- | --- | | 1 2 3 | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | 0 1 2 3 | | |  | D. | |  | | --- | | 1 2 3 4 | | |  | E. | |  | | --- | | Compilation fails | |   Your Answer: Option **B**  Correct Answer: Option **E**  Explanation:  The code will not compile because the variable counter is an interface variable that is by default final static. The compiler will complain at line 14 when the code attempts to increment counter.  Learn more problems on : [Declarations and Access Control](http://www.indiabix.com/java-programming/declarations-and-access-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/declarations-and-access-control/discussion-42) |

|  |  |
| --- | --- |
| 8. | What will be the output of the program?  public class Test  {  public int aMethod()  {  static int i = 0;  i++;  return i;  }  public static void main(String args[])  {  Test test = new Test();  test.aMethod();  int j = test.aMethod();  System.out.println(j);  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 0 | | |  | B. | |  | | --- | | 1 | | |  | C. | |  |  | | --- | --- | | 2 | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | D. | |  | | --- | | Compilation fails. | |   Your Answer: Option **C**  Correct Answer: Option **D**  Explanation:  Compilation failed because static was an illegal start of expression - method variables do not have a modifier (they are always considered local).  Learn more problems on : [Declarations and Access Control](http://www.indiabix.com/java-programming/declarations-and-access-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/declarations-and-access-control/discussion-45) |

|  |  |
| --- | --- |
| 9. | Which of the following statements is true? |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | It is sometimes good practice to throw an AssertionError explicitly. | | |  | B. | |  | | --- | | Private getter() and setter() methods should not use assertions to verify arguments. | | |  | C. | |  | | --- | | If an AssertionError is thrown in a try-catch block, the finally block will be bypassed. | | |  | D. | |  |  | | --- | --- | | It is proper to handle assertion statement failures using a catch (AssertionException ae)block. | http://www.indiabix.com/_files/images/website/wrong.gif | |   Your Answer: Option **D**  Correct Answer: Option **A**  Explanation:  Option A is correct because it is sometimes advisable to thrown an assertion error even if assertions have been disabled.  Option B is incorrect because it is considered appropriate to check argument values in private methods using assertions.  Option C is incorrect; finally is never bypassed.  Option D is incorrect because AssertionErrors should never be handled.  Learn more problems on : [Assertions](http://www.indiabix.com/java-programming/assertions/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/assertions/discussion-220) |

|  |  |
| --- | --- |
| 10. | What will be the output of the program?  public class RTExcept  {  public static void throwit ()  {  System.out.print("throwit ");  throw new RuntimeException();  }  public static void main(String [] args)  {  try  {  System.out.print("hello ");  throwit();  }  catch (Exception re )  {  System.out.print("caught ");  }  finally  {  System.out.print("finally ");  }  System.out.println("after ");  }  } |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | A. | |  |  | | --- | --- | | hello throwit caught | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | B. | |  | | --- | | Compilation fails | | |  | C. | |  | | --- | | hello throwit RuntimeException caught after | | |  | D. | |  | | --- | | hello throwit caught finally after | |   Your Answer: Option **A**  Correct Answer: Option **D**  Explanation:  The main() method properly catches and handles the RuntimeException in the catch block, finally runs (as it always does), and then the code returns to normal.  A, B and C are incorrect based on the program logic described above. Remember that properly handled exceptions do not cause the program to stop executing.  Learn more problems on : [Exceptions](http://www.indiabix.com/java-programming/exceptions/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/exceptions/discussion-105) |

|  |  |
| --- | --- |
| 11. | What will be the output of the program?  public abstract class AbstractTest  {  public int getNum()  {  return 45;  }  public abstract class Bar  {  public int getNum()  {  return 38;  }  }  public static void main (String [] args)  {  AbstractTest t = new AbstractTest()  {  public int getNum()  {  return 22;  }  };  AbstractTest.Bar f = t.new Bar()  {  public int getNum()  {  return 57;  }  };    System.out.println(f.getNum() + " " + t.getNum());  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 57 22 | | |  | B. | |  |  | | --- | --- | | 45 38 | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | 45 57 | | |  | D. | |  | | --- | | An exception occurs at runtime. | |   Your Answer: Option **B**  Correct Answer: Option **A**  Explanation:  You can define an inner class as abstract, which means you can instantiate only concrete subclasses of the abstract inner class. The object referenced by the variable t is an instance of an anonymous subclass of AbstractTest, and the anonymous class overrides the getNum() method to return 22. The variable referenced by f is an instance of an anonymous subclass of Bar, and the anonymous Bar subclass also overrides the getNum() method (to return 57). Remember that to instantiate a Bar instance, we need an instance of the enclosing AbstractTest class to tie to the new Bar inner class instance. AbstractTest can't be instantiated because it's abstract, so we created an anonymous subclass (non-abstract) and then used the instance of that anonymous subclass to tie to the new Bar subclass instance.  Learn more problems on : [Inner Classes](http://www.indiabix.com/java-programming/inner-classes/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/inner-classes/discussion-160) |

|  |  |
| --- | --- |
| 12. | Which three guarantee that a thread will leave the running state?   1. yield() 2. wait() 3. notify() 4. notifyAll() 5. sleep(1000) 6. aLiveThread.join() 7. Thread.killThread() |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 1, 2 and 4 | | |  | B. | |  |  | | --- | --- | | 2, 5 and 6 | http://www.indiabix.com/_files/images/website/accept.png | | |  | C. | |  | | --- | | 3, 4 and 7 | | |  | D. | |  | | --- | | 4, 5 and 7 | |   Your Answer: Option **B**  Correct Answer: Option **B**  Explanation:  (2) is correct because wait() always causes the current thread to go into the object's wait pool.  (5) is correct because sleep() will always pause the currently running thread for at least the duration specified in the sleep argument (unless an interrupted exception is thrown).  (6) is correct because, assuming that the thread you're calling join() on is alive, the thread calling join() will immediately block until the thread you're calling join() on is no longer alive.  (1) is wrong, but tempting. The yield() method is not guaranteed to cause a thread to leave the running state, although if there are runnable threads of the same priority as the currently running thread, then the current thread will probably leave the running state.  (3) and (4) are incorrect because they don't cause the thread invoking them to leave the running state.  (7) is wrong because there's no such method.  Learn more problems on : [Threads](http://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/threads/discussion-176) |

|  |  |
| --- | --- |
| 13. | Which of the following will produce an answer that is closest in value to a double, d, while not being greater than d? |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | (int)Math.min(d); | | |  | B. | |  |  | | --- | --- | | (int)Math.max(d); | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | (int)Math.abs(d); | | |  | D. | |  | | --- | | (int)Math.floor(d); | |   Your Answer: Option **B**  Correct Answer: Option **D**  Explanation:  The casting to an int is a smokescreen. Use a process of elimination to answer this question:  Option D is the correct answer, it is syntathecially correct and will consistently return a value less than d.  Option A and B are wrong because both the min() and max() methods require 2 arguments whereas here they are passed only one parameter.  Option C is wrong because it could return a value greater than d (if d was negative).  Learn more problems on : [Java.lang Class](http://www.indiabix.com/java-programming/java-lang-class/" \t "_blank)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/java-lang-class/discussion-261) |

|  |  |
| --- | --- |
| 14. | interface DoMath  {  double getArea(int rad);  }  interface MathPlus  {  double getVol(int b, int h);  }  /\* Missing Statements ? \*/  which two code fragments inserted at end of the program, will allow to compile?   1. class AllMath extends DoMath { double getArea(int r); } 2. interface AllMath implements MathPlus { double getVol(int x, int y); } 3. interface AllMath extends DoMath { float getAvg(int h, int l); } 4. class AllMath implements MathPlus { double getArea(int rad); } 5. abstract class AllMath implements DoMath, MathPlus { public double getArea(int rad) { return rad \* rad \* 3.14; } } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 1 only | | |  | B. | |  |  | | --- | --- | | 2 only | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | 3 and 5 | | |  | D. | |  | | --- | | 1 and 4 | |   Your Answer: Option **B**  Correct Answer: Option **C**  Explanation:  (3) are (5) are correct because interfaces and abstract classes do not need to fully implement the interfaces they extend or implement (respectively).  (1) is incorrect because a class cannot extend an interface. (2) is incorrect because an interface cannot implement anything. (4) is incorrect because the method being implemented is from the wrong interface.  Learn more problems on : [Declarations and Access Control](http://www.indiabix.com/java-programming/declarations-and-access-control/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/declarations-and-access-control/discussion-56) |

|  |  |
| --- | --- |
| 15. | Which statement is true? |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | A. | |  |  | | --- | --- | | catch(X x) can catch subclasses of X where X is a subclass of Exception. | http://www.indiabix.com/_files/images/website/accept.png | | |  | B. | |  | | --- | | The Error class is a RuntimeException. | | |  | C. | |  | | --- | | Any statement that can throw an Error must be enclosed in a try block. | | |  | D. | |  | | --- | | Any statement that can throw an Exception must be enclosed in a try block. | |   Your Answer: Option **A**  Correct Answer: Option **A**  Explanation:  Option A is correct. If the class specified in the catch clause does have subclasses, any exception object that subclasses the specified class will be caught as well.  Option B is wrong. The error class is a subclass of Throwable and not Runtime Exception.  Option C is wrong. You do not catch this class of error.  Option D is wrong. An exception can be thrown to the next method higher up the call stack.  Learn more problems on : [Exceptions](http://www.indiabix.com/java-programming/exceptions/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/exceptions/discussion-118) |

|  |  |
| --- | --- |
| 16. | What will be the output of the program?  public class SqrtExample  {  public static void main(String [] args)  {  double value = -9.0;  System.out.println( Math.sqrt(value));  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 3.0 | | |  | B. | |  | | --- | | -3.0 | | |  | C. | |  |  | | --- | --- | | NaN | http://www.indiabix.com/_files/images/website/accept.png | | |  | D. | |  | | --- | | Compilation fails. | |   Your Answer: Option **C**  Correct Answer: Option **C**  Explanation:  The sqrt() method returns NaN (not a number) when it's argument is less than zero.  Learn more problems on : [Java.lang Class](http://www.indiabix.com/java-programming/java-lang-class/" \t "_blank)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/java-lang-class/discussion-237) |

|  |  |
| --- | --- |
| 17. | What will be the output of the program?  int i = (int) Math.random(); |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | i = 0 | | |  | B. | |  | | --- | | i = 1 | | |  | C. | |  | | --- | | value of i is undetermined | | |  | D. | |  |  | | --- | --- | | Statement causes a compile error | http://www.indiabix.com/_files/images/website/wrong.gif | |   Your Answer: Option **D**  Correct Answer: Option **A**  Explanation:  Math.random() returns a double value greater than or equal to 0 and less than 1. Its value is stored to an intbut as this is a narrowing conversion, a cast is needed to tell the compiler that you are aware that there may be a loss of precision.  The value after the decimal point is lost when you cast a double to int and you are left with 0.  Learn more problems on : [Java.lang Class](http://www.indiabix.com/java-programming/java-lang-class/" \t "_blank)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/java-lang-class/discussion-258) |

|  |  |
| --- | --- |
| 18. | What will be the output of the program?  class MyThread extends Thread  {  public static void main(String [] args)  {  MyThread t = new MyThread(); /\* Line 5 \*/  t.run(); /\* Line 6 \*/  }  public void run()  {  for(int i=1; i < 3; ++i)  {  System.out.print(i + "..");  }  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | This code will not compile due to line 5. | | |  | B. | |  |  | | --- | --- | | This code will not compile due to line 6. | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | C. | |  | | --- | | 1..2.. | | |  | D. | |  | | --- | | 1..2..3.. | |   Your Answer: Option **B**  Correct Answer: Option **C**  Explanation:  Line 6 calls the run() method, so the run() method executes as a normal method should and it prints "1..2.."  A is incorrect because line 5 is the proper way to create an object.  B is incorrect because it is legal to call the run() method, even though this will not start a true thread of execution. The code after line 6 will not execute until the run() method is complete.  D is incorrect because the for loop only does two iterations.  Learn more problems on : [Threads](http://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/threads/discussion-178) |

|  |  |
| --- | --- |
| 19. | What will be the output of the program?  public class ThreadDemo  {  private int count = 1;  public synchronized void doSomething()  {  for (int i = 0; i < 10; i++)  System.out.println(count++);  }  public static void main(String[] args)  {  ThreadDemo demo = new ThreadDemo();  Thread a1 = new A(demo);  Thread a2 = new A(demo);  a1.start();  a2.start();  }  }  class A extends Thread  {  ThreadDemo demo;  public A(ThreadDemo td)  {  demo = td;  }  public void run()  {  demo.doSomething();  }  } |
| |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | It will print the numbers 0 to 19 sequentially | | |  | B. | |  | | --- | | It will print the numbers 1 to 20 sequentially | | |  | C. | |  |  | | --- | --- | | It will print the numbers 1 to 20, but the order cannot be determined | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | D. | |  | | --- | | The code will not compile. | |   Your Answer: Option **C**  Correct Answer: Option **B**  Explanation:  You have two different threads that share one reference to a common object.  The updating and output takes place inside synchronized code.  One thread will run to completion printing the numbers 1-10.  The second thread will then run to completion printing the numbers 11-20.  Learn more problems on : [Threads](http://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/threads/discussion-192) |

|  |  |
| --- | --- |
| 20. | What will be the output of the program?  class s1 extends Thread  {  public void run()  {  for(int i = 0; i < 3; i++)  {  System.out.println("A");  System.out.println("B");  }  }  }  class Test120 extends Thread  {  public void run()  {  for(int i = 0; i < 3; i++)  {  System.out.println("C");  System.out.println("D");  }  }  public static void main(String args[])  {  s1 t1 = new s1();  Test120 t2 = new Test120();  t1.start();  t2.start();  }  } |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | A. | |  |  | | --- | --- | | Compile time Error There is no start() method | http://www.indiabix.com/_files/images/website/wrong.gif | | |  | B. | |  | | --- | | Will print in this order AB CD AB... | | |  | C. | |  | | --- | | Will print but not be able to predict the Order | | |  | D. | |  | | --- | | Will print in this order ABCD...ABCD... | |   Your Answer: Option **A**  Correct Answer: Option **C**  Explanation:  We cannot predict the order in which threads are going to run.  Learn more problems on : [Threads](http://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](http://www.indiabix.com/java-programming/threads/discussion-190)   |  |  | | --- | --- | | 2. | Which one is a valid declaration of a boolean? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | boolean b1 = 0; | | |  | B. | |  | | --- | | boolean b2 = 'false'; | | |  | C. | |  | | --- | | boolean b3 = false; | | |  | D. | |  | | --- | | boolean b4 = Boolean.false(); | | |  | E. | |  | | --- | | boolean b5 = no; | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  A boolean can only be assigned the literal true or false.  Learn more problems on : [Language Fundamentals](https://www.indiabix.com/java-programming/language-fundamentals/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/language-fundamentals/discussion-5) |  |  |  | | --- | --- | | 3. | public interface Foo  {  int k = 4; /\* Line 3 \*/  }  Which three piece of codes are equivalent to line 3?   1. final int k = 4; 2. public int k = 4; 3. static int k = 4; 4. abstract int k = 4; 5. volatile int k = 4; 6. protected int k = 4; | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 1, 2 and 3 | | |  | B. | |  | | --- | | 2, 3 and 4 | | |  | C. | |  | | --- | | 3, 4 and 5 | | |  | D. | |  | | --- | | 4, 5 and 6 | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **A**  Explanation:  (1), (2) and (3) are correct. Interfaces can have constants, which are always implicitly public, static, and final. Interface constant declarations of public, static, and final are optional in any combination.  Learn more problems on : [Language Fundamentals](https://www.indiabix.com/java-programming/language-fundamentals/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/language-fundamentals/discussion-13) |  |  |  | | --- | --- | | 4. | What will be the output of the program?  public class If2  {  static boolean b1, b2;  public static void main(String [] args)  {  int x = 0;  if ( !b1 ) /\* Line 7 \*/  {  if ( !b2 ) /\* Line 9 \*/  {  b1 = true;  x++;  if ( 5 > 6 )  {  x++;  }  if ( !b1 )  x = x + 10;  else if ( b2 = true ) /\* Line 19 \*/  x = x + 100;  else if ( b1 | b2 ) /\* Line 21 \*/  x = x + 1000;  }  }  System.out.println(x);  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 0 | | |  | B. | |  | | --- | | 1 | | |  | C. | |  | | --- | | 101 | | |  | D. | |  | | --- | | 111 | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  As instance variables, b1 and b2 are initialized to false. The if tests on lines 7 and 9 are successful so b1 is set to true and x is incremented. The next if test to succeed is on line 19 (note that the code is not testing to see if b2 is true, it is setting b2 to be true). Since line 19 was successful, subsequent else-if's (line 21) will be skipped.  Learn more problems on : [Flow Control](https://www.indiabix.com/java-programming/flow-control/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/flow-control/discussion-87) |  |  |  | | --- | --- | | 5. | Which of the following are Java reserved words?   1. run 2. import 3. default 4. implement | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 1 and 2 | | |  | B. | |  | | --- | | 2 and 3 | | |  | C. | |  | | --- | | 3 and 4 | | |  | D. | |  | | --- | | 2 and 4 | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **B**  Explanation:  (2) - This is a Java keyword  (3) - This is a Java keyword  (1) - Is incorrect because although it is a method of Thread/Runnable it is not a keyword  (4) - This is not a Java keyword the keyword is implements  Learn more problems on : [Objects and Collections](https://www.indiabix.com/java-programming/objects-and-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/objects-and-collections/discussion-126) |  |  |  | | --- | --- | | 6. | Which statement is true? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | If only one thread is blocked in the wait method of an object, and another thread executes the modify on that same object, then the first thread immediately resumes execution. | | |  | B. | |  | | --- | | If a thread is blocked in the wait method of an object, and another thread executes the notify method on the same object, it is still possible that the first thread might never resume execution. | | |  | C. | |  | | --- | | If a thread is blocked in the wait method of an object, and another thread executes the notify method on the same object, then the first thread definitely resumes execution as a direct and sole consequence of the notify call. | | |  | D. | |  | | --- | | If two threads are blocked in the wait method of one object, and another thread executes the notify method on the same object, then the first thread that executed the wait call first definitely resumes execution as a direct and sole consequence of the notify call. | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **B**  Explanation:  Option B is correct - The notify method only wakes the thread. It does not guarantee that the thread will run.  Option A is incorrect - just because another thread activates the modify method in A this does not mean that the thread will automatically resume execution  Option C is incorrect - This is incorrect because as said in Answer B notify only wakes the thread but further to this once it is awake it goes back into the stack and awaits execution therefore it is not a "direct and sole consequence of the notify call"  Option D is incorrect - The notify method wakes one waiting thread up. If there are more than one sleeping threads then the choice as to which thread to wake is made by the machine rather than you therefore you cannot guarantee that the notify'ed thread will be the first waiting thread.  Learn more problems on : [Threads](https://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/threads/discussion-198) |  |  |  | | --- | --- | | 7. | public class Test { }  What is the prototype of the default constructor? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | Test( ) | | |  | B. | |  | | --- | | Test(void) | | |  | C. | |  | | --- | | public Test( ) | | |  | D. | |  | | --- | | public Test(void) | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  Option A and B are wrong because they use the default access modifier and the access modifier for the class is public (remember, the default constructor has the same access modifier as the class).  Option D is wrong. The void makes the compiler think that this is a method specification - in fact if it were a method specification the compiler would spit it out.  Learn more problems on : [Declarations and Access Control](https://www.indiabix.com/java-programming/declarations-and-access-control/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/declarations-and-access-control/discussion-39) |  |  |  | | --- | --- | | 8. | Which of the following class level (nonlocal) variable declarations will not compile? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | protected int a; | | |  | B. | |  | | --- | | transient int b = 3; | | |  | C. | |  | | --- | | private synchronized int e; | | |  | D. | |  | | --- | | volatile int d; | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  Option C will not compile; the synchronized modifier applies only to methods.  Option A and B will compile because protected and transient are legal variable modifiers. Option D will compile because volatile is a proper variable modifier.  Learn more problems on : [Declarations and Access Control](https://www.indiabix.com/java-programming/declarations-and-access-control/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/declarations-and-access-control/discussion-29) |  |  |  | | --- | --- | | 9. | What is the name of the method used to start a thread execution? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | init(); | | |  | B. | |  | | --- | | start(); | | |  | C. | |  | | --- | | run(); | | |  | D. | |  | | --- | | resume(); | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **B**  Explanation:  Option B is Correct. The start() method causes this thread to begin execution; the Java Virtual Machine calls the run method of this thread.  Option A is wrong. There is no init() method in the Thread class.  Option C is wrong. The run() method of a thread is like the main() method to an application. Starting the thread causes the object's run method to be called in that separately executing thread.  Option D is wrong. The resume() method is deprecated. It resumes a suspended thread.  Learn more problems on : [Threads](https://www.indiabix.com/java-programming/threads/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/threads/discussion-162) |  |  |  | | --- | --- | | 10. | What will be the output of the program?  public class ExamQuestion7  {  static int j;  static void methodA(int i)  {  boolean b;  do  {  b = i<10 | methodB(4); /\* Line 9 \*/  b = i<10 || methodB(8); /\* Line 10 \*/  }while (!b);  }  static boolean methodB(int i)  {  j += i;  return true;  }  public static void main(String[] args)  {  methodA(0);  System.out.println( "j = " + j );  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | j = 0 | | |  | B. | |  | | --- | | j = 4 | | |  | C. | |  | | --- | | j = 8 | | |  | D. | |  | | --- | | The code will run with no output | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **B**  Explanation:  The lines to watch here are lines 9 & 10. Line 9 features the non-shortcut version of the OR operator so both of its operands will be evaluated and therefore methodB(4) is executed.  However line 10 has the shortcut version of the OR operator and if the 1st of its operands evaluates to true (which in this case is true), then the 2nd operand isn't evaluated, so methodB(8) never gets called.  The loop is only executed once, b is initialized to false and is assigned true on line 9. Thus j = 4.  Learn more problems on : [Java.lang Class](https://www.indiabix.com/java-programming/java-lang-class/" \t "_blank)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/java-lang-class/discussion-253) |  |  |  | | --- | --- | | 11. | What will be the output of the program?  class PassS  {  public static void main(String [] args)  {  PassS p = new PassS();  p.start();  }  void start()  {  String s1 = "slip";  String s2 = fix(s1);  System.out.println(s1 + " " + s2);  }  String fix(String s1)  {  s1 = s1 + "stream";  System.out.print(s1 + " ");  return "stream";  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | slip stream | | |  | B. | |  | | --- | | slipstream stream | | |  | C. | |  | | --- | | stream slip stream | | |  | D. | |  | | --- | | slipstream slip stream | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  When the fix() method is first entered, start()'s s1 and fix()'s s1 reference variables both refer to the same String object (with a value of "slip"). Fix()'s s1 is reassigned to a new object that is created when the concatenation occurs (this second String object has a value of "slipstream"). When the program returns to start(), another String object is created, referred to by s2 and with a value of "stream".  Learn more problems on : [Operators and Assignments](https://www.indiabix.com/java-programming/operators-and-assignments/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/operators-and-assignments/discussion-68) |  |  |  | | --- | --- | | 12. | What will be the output of the program?  class Two  {  byte x;  }  class PassO  {  public static void main(String [] args)  {  PassO p = new PassO();  p.start();  }  void start()  {  Two t = new Two();  System.out.print(t.x + " ");  Two t2 = fix(t);  System.out.println(t.x + " " + t2.x);  }  Two fix(Two tt)  {  tt.x = 42;  return tt;  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | null null 42 | | |  | B. | |  | | --- | | 0 0 42 | | |  | C. | |  | | --- | | 0 42 42 | | |  | D. | |  | | --- | | 0 0 0 | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  In the fix() method, the reference variable tt refers to the same object (class Two) as the t reference variable. Updating tt.x in the fix() method updates t.x (they are one in the same object). Remember also that the instance variable x in the Two class is initialized to 0.  Learn more problems on : [Operators and Assignments](https://www.indiabix.com/java-programming/operators-and-assignments/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/operators-and-assignments/discussion-71) |  |  |  | | --- | --- | | 13. | What will be the output of the program?  class SC2  {  public static void main(String [] args)  {  SC2 s = new SC2();  s.start();  }  void start()  {  int a = 3;  int b = 4;  System.out.print(" " + 7 + 2 + " ");  System.out.print(a + b);  System.out.print(" " + a + b + " ");  System.out.print(foo() + a + b + " ");  System.out.println(a + b + foo());  }  String foo()  {  return "foo";  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 9 7 7 foo 7 7foo | | |  | B. | |  | | --- | | 72 34 34 foo34 34foo | | |  | C. | |  | | --- | | 9 7 7 foo34 34foo | | |  | D. | |  | | --- | | 72 7 34 foo34 7foo | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  Because all of these expressions use the + operator, there is no precedence to worry about and all of the expressions will be evaluated from left to right. If either operand being evaluated is a String, the + operator will concatenate the two operands; if both operands are numeric, the + operator will add the two operands.  Learn more problems on : [Operators and Assignments](https://www.indiabix.com/java-programming/operators-and-assignments/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/operators-and-assignments/discussion-69) |  |  |  | | --- | --- | | 14. | import java.io.\*;  public class MyProgram  {  public static void main(String args[])  {  FileOutputStream out = null;  try  {  out = new FileOutputStream("test.txt");  out.write(122);  }  catch(IOException io)  {  System.out.println("IO Error.");  }  finally  {  out.close();  }  }  }  and given that all methods of class FileOutputStream, including close(), throw an IOException, which of these is true? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | This program will compile successfully. | | |  | B. | |  | | --- | | This program fails to compile due to an error at line 4. | | |  | C. | |  | | --- | | This program fails to compile due to an error at line 6. | | |  | D. | |  | | --- | | This program fails to compile due to an error at line 18. | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  Any method (in this case, the main() method) that throws a checked exception (in this case, out.close() ) must be called within a try clause, or the method must declare that it throws the exception. Either main() must declare that it throws an exception, or the call to out.close() in the finally block must fall inside a (in this case nested) try-catch block.  Learn more problems on : [Exceptions](https://www.indiabix.com/java-programming/exceptions/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/exceptions/discussion-115) |  |  |  | | --- | --- | | 15. | public class MyProgram  {  public static void throwit()  {  throw new RuntimeException();  }  public static void main(String args[])  {  try  {  System.out.println("Hello world ");  throwit();  System.out.println("Done with try block ");  }  finally  {  System.out.println("Finally executing ");  }  }  }  which answer most closely indicates the behavior of the program? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | The program will not compile. | | |  | B. | |  | | --- | | The program will print Hello world, then will print that a RuntimeException has occurred, then will print Done with try block, and then will print Finally executing. | | |  | C. | |  | | --- | | The program will print Hello world, then will print that a RuntimeException has occurred, and then will print Finally executing. | | |  | D. | |  | | --- | | The program will print Hello world, then will print Finally executing, then will print that a RuntimeException has occurred. | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  Once the program throws a RuntimeException (in the throwit() method) that is not caught, the finally block will be executed and the program will be terminated. If a method does not handle an exception, the finally block is executed before the exception is propagated.  Learn more problems on : [Exceptions](https://www.indiabix.com/java-programming/exceptions/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/exceptions/discussion-116) |  |  |  | | --- | --- | | 16. | Which statement is true? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | Calling Runtime.gc() will cause eligible objects to be garbage collected. | | |  | B. | |  | | --- | | The garbage collector uses a mark and sweep algorithm. | | |  | C. | |  | | --- | | If an object can be accessed from a live thread, it can't be garbage collected. | | |  | D. | |  | | --- | | If object 1 refers to object 2, then object 2 can't be garbage collected. | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  This is a great way to think about when objects can be garbage collected.  Option A and B assume guarantees that the garbage collector never makes.  Option D is wrong because of the now famous islands of isolation scenario.  Learn more problems on : [Garbage Collections](https://www.indiabix.com/java-programming/garbage-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/garbage-collections/discussion-211) |  |  |  | | --- | --- | | 17. | class HappyGarbage01  {  public static void main(String args[])  {  HappyGarbage01 h = new HappyGarbage01();  h.methodA(); /\* Line 6 \*/  }  Object methodA()  {  Object obj1 = new Object();  Object [] obj2 = new Object[1];  obj2[0] = obj1;  obj1 = null;  return obj2[0];  }  }  Where will be the most chance of the garbage collector being invoked? | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | After line 9 | | |  | B. | |  | | --- | | After line 10 | | |  | C. | |  | | --- | | After line 11 | | |  | D. | |  | | --- | | Garbage collector never invoked in methodA() | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  Option D is correct. Garbage collection takes place after the method has returned its reference to the object. The method returns to line 6, there is no reference to store the return value. so garbage collection takes place after line 6.  Option A is wrong. Because the reference to obj1 is stored in obj2[0]. The Object obj1 still exists on the heap and can be accessed by an active thread through the reference stored in obj2[0].  Option B is wrong. Because it is only one of the references to the object obj1, the other reference is maintained in obj2[0].  Option C is wrong. The garbage collector will not be called here because a reference to the object is being maintained and returned in obj2[0].  Learn more problems on : [Garbage Collections](https://www.indiabix.com/java-programming/garbage-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/garbage-collections/discussion-209) |  |  |  | | --- | --- | | 18. | What will be the output of the program?  public class Test  {  public static void main (String[] args)  {  String foo = args[1];  String bar = args[2];  String baz = args[3];  System.out.println("baz = " + baz); /\* Line 8 \*/  }  }  And the command line invocation:  > java Test red green blue | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | baz = | | |  | B. | |  | | --- | | baz = null | | |  | C. | |  | | --- | | baz = blue | | |  | D. | |  | | --- | | Runtime Exception | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **D**  Explanation:  When running the program you entered 3 arguments "red", "green" and "blue". When dealing with arrays in java you must remember ALL ARRAYS IN JAVA ARE ZERO BASED therefore args[0] becomes "red", args[1]becomes "green" and args[2] becomes "blue".  When the program entcounters line 8 above at runtime it looks for args[3] which has never been created therefore you get an  ArrayIndexOutOfBoundsException at runtime.  Learn more problems on : [Objects and Collections](https://www.indiabix.com/java-programming/objects-and-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/objects-and-collections/discussion-138) |  |  |  | | --- | --- | | 19. | What will be the output of the program?  import java.util.\*;  class I  {  public static void main (String[] args)  {  Object i = new ArrayList().iterator();  System.out.print((i instanceof List)+",");  System.out.print((i instanceof Iterator)+",");  System.out.print(i instanceof ListIterator);  }  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | Prints: false, false, false | | |  | B. | |  | | --- | | Prints: false, false, true | | |  | C. | |  | | --- | | Prints: false, true, false | | |  | D. | |  | | --- | | Prints: false, true, true | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **C**  Explanation:  The iterator() method returns an iterator over the elements in the list in proper sequence, it doesn't return a List or a ListIterator object.  A ListIterator can be obtained by invoking the listIterator method.  Learn more problems on : [Objects and Collections](https://www.indiabix.com/java-programming/objects-and-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/objects-and-collections/discussion-134) |  |  |  | | --- | --- | | 20. | What will be the output of the program?  public static void main(String[] args)  {  Object obj = new Object()  {  public int hashCode()  {  return 42;  }  };  System.out.println(obj.hashCode());  } | | |  |  |  |  | | --- | --- | --- | --- | |  | A. | |  | | --- | | 42 | | |  | B. | |  | | --- | | Runtime Exception | | |  | C. | |  | | --- | | Compile Error at line 2 | | |  | D. | |  | | --- | | Compile Error at line 5 | |   Your Answer: Option **(Not Answered)**  Correct Answer: Option **A**  Explanation:  This code is an example of an anonymous inner class. They can be declared to extend another class or implement a single interface. Since they have no name you can not use the "new" keyword on them.  In this case the annoynous class is extending the Object class. Within the {} you place the methods you want for that class. After this class has been declared its methods can be used by that object in the usual way e.g. objectname.annoymousClassMethod()  Learn more problems on : [Objects and Collections](https://www.indiabix.com/java-programming/objects-and-collections/)  Discuss about this problem : [Discuss in Forum](https://www.indiabix.com/java-programming/objects-and-collections/discussion-139) | |