JNTU ONLINE EXAMINATIONS [Mid 2 - JAVA]

1.Why does the following class have a syntax error?

```
import java.applet.*;
public class Test extends Applet implements Runnable {
public void init() throws InterruptedException {
Thread t = new Thread(this);
t.sleep(1000);
}
public synchronized void run() {
}
}
```

a. The sleep() method is not invoked correctly; it should be invoked as Thread.sleep(1000).
b. You cannot put the keyword synchronized in the run() method.
c. The init() method is defined in the Applet class, and it is overridden incorrectly because it cannot claim exceptions in the subclass.
d. The sleep() method should be put in the try-catch block. This is the only reason for the compilation failure.

2. Analyze the following code:

```
public class Test implements Runnable {
public static void main(String[] args) {
Test t = new Test();
}
public Test() {
Thread t = new Thread(this);
t.start();
}
public void run() {
System.out.println("test");
}
}
```

a. The program has a compilation error because it is defined in both the main() method and the constructor Test().
b. The program compiles fine, but it does not run because you cannot use the keyword this in the constructor.
c. The program compiles and runs and displays nothing.
d. The program compiles and runs and displays test.

3. Analyze the following code:

```
public class Test implements Runnable {
public static void main(String args) {
Thread t = new Thread(this);
t.start();
}
public void run() {
System.out.println("test'');
}
}
```

a. The program does not compile because this cannot be referenced in a static method.
b. The program compiles fine, but it does not print anything because t does not invoke the run() method.
c. The program compiles and runs fine and displays test on the console.
d. Runtime Error

4. Analyze the following code:

```
public class Test implements Runnable {
public static void main(String[ ] args) {
Test t = new Test();
t.start();
}
public void run() {
}
}
```

a. The program does not compile because the start() method is not defined in the Test class.
b. The program compiles, but it does not run because the start() method is not defined.
c. The program compiles, but it does not run because the run() method is not implemented.
d. The program compiles and runs fine.

5. You can use the _ _ _ _ _ _ _ _ _ method to force one thread to wait for another thread to finish.
a. sleep(long milliseconds)
b. yield()
c. stop()
d. join()

6. Analyze the following code:

```
public abstract class Test implements Runnable {
public void do Something() {
};
}
```

a. The program will not compile because it does not implement the run() method.
b. The program will not compile because it does not contain abstract methods.
c. The program compiles fine.
d. Runtime Error.

7.Which of the following expressions must be true if you create a thread using Thread = new Thread(object)?
a. object instanceof Thread
b. object instanceof Frame
c. object instanceof Applet
d. object instanceof Runnable

8. The following method in the Thread class is not deprecated:
a. yield()
b. stop();
c. resume();
d. suspend();

9. You can use the _ _ _ _ _ _ _ _ _ _ method to temporarily to release time for other threads.
a. sleep(int milliseconds)

b. yield()
c. stop()
d. suspend()

10. Which of the following statement is not defined in the Object class?
a. sleep(long milliseconds)
b. wait()
c. notify()
d. notifyAll()

11. When you run the following program, what will happen?

```
public class Test extends Thread {
public static void main(String[] args) {
Test t = new Test();
t.start();
t.start();
}
public void run() {
System.out.println("test");
}
}
```

a. Nothing is displayed.
b. The program displays test twice.
c. The program displays test once.
d. An illegal java.lang. Illegal Thread State Exception may be thrown because you just started thread and thread might have not yet finished before you start it again.

12. Given the following code, which set of code can be used to replace the comment so that the program displays time to the console every second?

```
import java.applet.*;
import java.util.*;
public class Test extends Applet implements Runnable {
public void init() {
Thread t = new Thread(this);
t.start();
}
public void run() {
for(; ;) {
//display time every second
System.out.println(new Date().toString());
}
}
}
```

a. try { sleep(1000); } catch(InterruptedException e) { }
b. try { t.sleep(1000); } catch(InterruptedException e) { }
c. try { Thread.sleep(1000); } catch(RuntimeException e) { }
d. try { Thread.sleep(1000); } catch(InterruptedException e) { }

13. How do you create a cashed thread pool?
a. ExecutorService executor = Executors.newCachedThreadPool();
b. ExecutorService executor = Executors.newCachedThreadPool(1);
c. ExecutorService executor = Executors.newCachedThreadPool(2);
d. ExecutorService executor = Executors.newCachedThreadPool(3);

14. The keyword to synchronize methods in Java is _ _ _ _ _ _ _ _ _ _ .
a. synchronize
b. synchronizing
c. synchronized
d. Synchronized

15. Which of the following statement is false?
a. You cannot use a timer or a thread to control animation.
b. A timer is a source component that fires an ActionEvent at a "fixed rate.''
c. The timer and event-handling run on the same event dispatcher thread. If it takes a long time to handle the event, the actual delay time between two events will be longer than the requested delay time.
d. In general, threads are more reliable and responsive than timers.

16. Which of the following statement is false? A. B. C.
a. The javax.swing.SwingUtilities.invokeLater method creates a thread.
b. The javax.swing.SwingUtilities.invokeAndWait method runs the code in the event dispatcher thread.
c. The javax.swing.SwingUtilities.invokeLater method runs the code in the event dispatcher thread and doesn't return until the event-dispatching thread has executed the specified code.
d. GUI event handling are executed in the event dispatcher thread.

17. Which of the following statement is false?
a. A synchronized instance method doesn't acquire a lock on the object for which the method was invoked.
b. A synchronized instance method acquires a lock on the class of the object for which the method was invoked.
c. A synchronized statement can be used to acquire a lock on any object, not just this object, when executing a block of the code in a method.
d. A synchronized statement is placed inside a synchronized block.

18. Which of the following method is a static in java.lang.Thread?
a. run()
b. sleep(long)
c. start()
d. join()

19. Which of the following methods in Thread throws Interrupted Exception?
a. run()
b. sleep(long)
c. start()
d. yield()

20. Suppose there are three Runnable tasks, task1, task2, task3. How do you run them in a thread pool with 2 fixed threads?

a. new Thread(task1).start(); new Thread(task2).start(); new Thread(task3).start();
b. ExecutorService executor = Executors.newFixedThreadPool(3); executor.execute(task1); executor.execute(task2); executor.execute(task3);
c. ExecutorService executor = Executors.newFixedThreadPool(2); executor.execute(task1); executor.execute(task2); executor.execute(task3);
d. ExecutorService executor = Executors.newFixedThreadPool(1); executor.execute(task1); executor.execute(task2); executor.execute(task3);

21. Which of the following are correct statements to create a Lock so the longest-wait thread will obtain the lock first?
a. Lock lock = new Lock();
b. Lock lock = new ReentrantLock();
c. Lock lock = new ReentrantLock(true);
d. Lock lock = new ReentrantLock(false);

22. How do you create a condition on a lock?
a. Condition condition = lock.getCondition();
b. Condition condition = lock.newCondition();
c. Condition condition = Lock.newCondition();
d. Condition condition = Lock.getCondition();

23. Which method on a condition should you invoke to causes the current thread to wait until the condition is signaled?
a. condition.await();
b. condition.wait();
c. condition.waiting();
d. condition.waited();

24. Which method on a condition should you invoke to wake all waiting threads.
a. condition.wake();
b. condition.signal();
c. condition.wakeAll();
d. condition.signalAll();

25. You cannot create a blocking queue using _ _ _ _ _ _ _ _ _ _ _ _ _ .
a. ArrayBlockingQueue
b. LinkedBlockingQueue
c. PriorityBlockingQueue
d. PriorityQueue

26. Which of the following is not a correct statement
a. Lock lock = new Lock();
b. Lock lock = new ReentrantLock();
c. Lock lock = new ReentrantLock(true);
d. Lock lock = new ReentrantLock(false);

27. Which of the following statement is false?
a. A condition is associated with a lock.
b. To invoke methods on a condition, the lock must be obtained first.
c. Once you invoke the await method on a condition, the lock is automatically released.
Once the condition is right, the thread re-acquires the lock and continues executing.
d. The signal method on a condition causes the lock for the condition to be released.

28. Which of the following statements are true?
a. The wait(), notify(), and notifyAll() methods must be invoked from a synchronized method or a synchronized block.
b. When wait() is invoked, it pauses the thread and releases the lock on the object simultaneously. When the thread is restarted after being notified, the lock is automatically reacquired.
c. The notify() method can wake only one waiting thread.
d. An exception would occur if no thread is waiting on the object when the notify() method is invoked on the object.

29. Which of the following statement is false?
a. A blocking queue has a capacity.
b. A blocking queue causes a thread to block when you try to add an element to a full queue.
c. A blocking queue causes a thread to block when you try to remove an element from an empty queue.
d. The BlockingQueue interface is the derived interface for all concrete blocking queue classes.

30. Which of the following statement is false?
a. Semaphores cannot be used to restrict the number of threads that access a shared resource.
b. Before accessing the resource, a thread must acquire a permit from the semaphore.
c. After finishing with the resource, the thread must return the permit back to the semaphore.
d. You can create a Semaphore with a specified number of permits.

31. Which of the following methods can be used to obtain a permit from a Semaphore s?
a. get()
b. ask()
c. acquire()
d. delete()

32. An instance of _ _ _ _ _ _ _ _ _ describes the errors caused by your program and external circumstances. These errors can be caught and handled by your program.
a. Runtime Exception
b. Exception
c. Error
d. Throwable

33. The following code causes Java to throw _ _ _ _ _ _ _ _ _ .
int number = Integer. + 1;
a. Runtime Exception
b. Exception
c. Error
d. No Exceptions

34. An instance of _ _ _ _ _ _ _ _ _ describes system errors. If this type of errors occurs, there is little you can do beyond notifying the user and trying to terminate the program gracefully.

a. Runtime Exception
b. Exception
c. Error
d. Throwable

35. Which of the following methods can be used to return a permit to a Semaphores?
a. return()
b. release()
c. send()
d. add()

36. A Java exception is an instance of _ _ _ _ _ _ _ _ _ _ _ _ .
a. Runtime Exception
b. Exception
c. Error
d. Throwable

37. An instance of _ _ _ _ _ _ _ _ _ describes programming errors, such as bad casting, accessing an out-of-bounds array, and numeric errors.
a. Runtime Exception
b. Exception
c. Error
d. Throwable

38. An instance of _ _ _ _ _ _ _ _ _ are unchecked exceptions.
a. Runtime Exception
b. Exception
c. throwable
d. Error

39. What exception type does the following program throw?
```
public class Test {
public static void main(String[] args) {
System.out.println(1 / 0);
}
}
```
a. ArithmeticException
b. ArrayIndexOutOfBoundsException
c. StringIndexOutOfBoundsException
d. ClassCastException

40. What exception type does the following program throw?
```
public class Test {
public static void main(String[] args) {
Object o = new Object();
String d = (String)o;
}
}
```
a. Arithmetic Exception
b. ArrayIndex Out Of Bounds Exception
c. StringIndex Out Of Bounds Exception
d. Class Cast Exception

41. What exception type does the following program throw?
```
public class Test {
public static void main(String[] args) {
Object o = null;
System.out.println(o);
}
}
```
a. Arithmetic Exception
b. ArrayIndex Out Of BoundsException
c. StringIndex Out Of Bounds Exception
d. No exception

42. Analyze the following code:
```
class Test {
public static void main(String[] args) {
try {
String s = "5.6'';
Integer.parseInt(s); // Cause a NumberFormat Exception
int i = 0;
int y = 2 / i;
}
catch (Exception ex) {
System.out.println("Number Format Exception'');
}
catch (RuntimeException ex) {
System.out.println("Runtime Exception'');
}
}
}
```
a. The program displays NumberFormatException.
b. The program displays RuntimeException.
c. The program displays NumberFormatException followed by RuntimeException.
d. The program has a compilation error.

43. What is displayed on the console when running the following program?
```
class Test {
public static void main(String[] args) {
try {
method();
System.out.println("After the method call");
}
catch (NumberFormatException ex) {
System.out.println("NumberFormatException");
}
catch (RuntimeException ex) {
System.out.println("RuntimeException");
}
}
static void method() {
String s = "5.6'';
Integer.parseInt(s); // Cause a NumberFormat Exception
int i = 0;
int y = 2 / i;
System.out.println("Welcome to Java'');
}
}
```
a. The program displays NumberFormatException.
b. The program displays NumberFormatException followed by After the method call.
c. The program displays NumberFormatException followed by RuntimeException.
d. The program has a compilation error.

44. What exception type does the following program throw?
```
public class Test {
public static void main(String[] args) {
int[] list = new int[5];
System.out.println(list[5]);
}
```

```
}
```
a. Arithmetic Exception
b. ArrayIndex Out Of Bounds Exception
c. StringIndex Out Of Bounds Exception
d. Class Cast Exception

45. Which of the following statement is false?

a. You use the keyword throws to declare exceptions in the method heading.
b. A method may declare to throw multiple exceptions.
c. To throw an exception, use the key word throw.
d. If a checked exception occurs in a method, it need not to be either caught or declared to be thrown from the method.

46. What exception type does the following program throw?

```
public class Test {
public static void main(String[] args) {
String s = "abc'';
System.out.println(s.charAt(3));
}
}
```
a. Arithmetic Exception
b. ArrayIndex Out Of Bounds Exception
c. StringIndex Out Of Bounds Exception
d. Class Cast Exception

47. What exception type does the following program throw?

```
public class Test {
public static void main(String[] args) {
Object o = null;
System.out.println(o.toString());
}
}
```
a. Arithmetic Exception
b. ArrayIndex Out Of Bounds Exception
c. StringIndex Out Of Bounds Exception
d. Class Cast Exception

48. A method must declare to throw _ _ _ _ _ _ _ _ .

a. unchecked exceptions
b. checked exceptions
c. Error
d. Runtime Exception

49. Analyze the following code:

```
class Test {
public static void main(String[] args)
throws MyException {
System.out.println("Welcome to Java");
}
}
class MyException extends Error {
}
```
a. You should not declare a class that extends Error, because Error raises a fatal error that terminates the program.
b. You cannot declare an exception in the main method.
c. You declared an exception in the main method, but you did not throw it.
d. The program has a compilation error.

50. Analyze the following program.

```
class Test {
public static void main(String[] args) {
try {
String s = "5.6";
Integer.parseInt(s); // Cause a NumberFormat Exception
int i = 0;
int y = 2 / i;
System.out.println("Welcome to Java");
}
catch (Exception ex) {
System.out.println(ex);
}
}
}
```
a. An exception is raised due to Integer.parseInt(s);
b. An exception is raised due to 2 / i;
c. The program has a compilation error.
d. The program compiles and runs without exceptions.

51. What is displayed on the console when running the following program?

```
class Test {
public static void main(String[] args) {
try {
System.out.println("Welcome to Java'');
int i = 0;
int y = 2/i;
System.out.println("Welcome to Java'');
}
catch (RuntimeException ex) {
System.out.println("Welcome to Java'');
}
finally {
System.out.println("End of the block'');
}
System.out.println("End of the block'');
}
}
```
a. The program displays Welcome to Java three times followed by End of the block.
b. The program displays Welcome to Java two times followed by End of the block.
c. The program displays Welcome to Java two times followed by End of the block two times.
d. You cannot catch RuntimeException errors.

52. What is displayed on the console when running the following program?

```
class Test {
public static void main(String[] args) {
try {
System.out.println("Welcome to Java'');
int i = 0;
int y = 2/i;
System.out.println("Welcome to Java'');
}
finally {
System.out.println("End of the block'');
}
System.out.println("End of the block'');
}
}
```
a. The program displays Welcome to Java three times followed by End of the block.

b. The program displays Welcome to Java two times followed by End of the block.
c. The program displays Welcome to Java two times followed by End of the block two times.
d. The program displays Welcome to Java and End of the block, and then terminates because of an unhandled exception.

53. Which of the following is not an advantage of Java exception handling?
a. Java separates exception handling from normal processing tasks.
b. Exception handling improves performance.
c. Exception handling makes it possible for the caller's caller to handle the exception.
d. Exception handling simplifies programming because the error-reporting and errorhandling code can be placed at the catch block.

54. What is displayed on the console when running the following program?

```
class Test {
public static void main(String[] args) {
try {
System.out.println("Welcome to Java'');
int i = 0;
int y = 2/i;
System.out.println("Welcome to Java'');
}
catch (RuntimeException ex) {
System.out.println("Welcome to Java'');
}
finally {
System.out.println("End of the block'');
}
}
}
```

a. The program displays Welcome to Java three times followed by End of the block.
b. The program displays Welcome to Java two times followed by End of the block.
c. The program displays Welcome to Java three times.
d. The program displays Welcome to Java two times.

55. What is wrong in the following program?

```
class Test {
public static void main (String[] args) {
try {
System.out.println("Welcome to Java'');
}
}
}
```

a. You cannot have a try block without a catch block.
b. You cannot have a try block without a catch block or a finally block.
c. A method call that does not declare exceptions cannot be placed inside a try block.
d. Nothing is wrong.

56. What is displayed on the console when running the following program?

```
class Test {
public static void main (String[] args) {
try {
System.out.println("Welcome to Java);
}
finally {
System.out.println("The finally clause is executed'');
}
}
}
```

a. Welcome to Java
b. Welcome to Java followed by The finally clause is executed in the next line
c. The finally clause is executed
d. Runtime Error

57. What is displayed on the console when running the following program?

```
class Test {
public static void main(String[] args) {
try {
System.out.println("Welcome to Java'');
int i = 0;
int y = 2/i;
System.out.println("Welcome to HTML'');
}
finally {
System.out.println("The finally clause is executed'');
}
}
}
```

a. Welcome to Java.
b. Welcome to Java followed by The finally clause is executed in the next line.
c. The program displays three lines: Welcome to Java, Welcome to HTML, The finally clause is executed.
d. Runtime Error

58. What is displayed on the console when running the following program?

```
class Test {
public static void main(String[] args) {
try {
System.out.println("Java'');
int i = 0;
double y = 2.0 / i;
System.out.println("Welcome'');
}
finally {
System.out.println("The finally clause is executed'');
}
}
}
```

a. Welcome to Java.
b. Welcome to Java followed by The finally clause is executed in the next line.
c. The program displays three lines: Welcome to Java, Welcome to HTML, The finally clause is executed.
d. Runtime Error

59. The component that processes the listener is called _ _ _ _ _ _ _ _ _ _ _
a. the source object
b. the listener object
c. the adapter object

d. the adaptee object

60. Which of the following statements registers a panel object p as a listener for a button variable jbt?

a. addActionListener(p);

b. jbt.addActionListener(p);

c. jbt.addActionEventListener(p);

d. jbt.addEventListener(p);

61. Which of the following is not a correct assertion statement?

a. assert (i > 10);

b. assert sum > 10 & & sum < 5 * 10 : "sum is '' + sum;

c. assert "sum is'' + sum;

d. assert(i<10);

62. Clicking the closing button on the upper-right corner of a frame generates a(n) _ _ _ _ _ _ _ _ _ _ _ event.

a. ItemEvent

b. WindowEvent

c. MouseMotionEvent

d. ComponentEvent

63. Analyze the following code:

```
class Test {
public static void main(String[] args) {
try {
int zero = 0;
int y = 2/zero;
try {
String s = "5.6'';
Integer.parseInt(s); // Cause a NumberFormatException
}
catch(Exception e) {
}
}
catch(RuntimeException e) {
System.out.println(e);
}
}
}
```

a. A try-catch block cannot be embedded inside another try-catch block.

b. A good programming practice is to avoid nesting try-catch blocks, because nesting makes programs difficult to read. You can rewrite the program using only one try-catch block.

c. The program has a compilation error because Exception appears before RuntimeException.

d. Runtime Error

64. Which of the following statement is correct to rethrow exception ex along with new information?

a. throw new Exception("New info'', ex);

b. throw ex; throw new Exception("Some new info'');

c. throw new Exception(ex, "New info'');

d. throw new Exception("New info"); throw ex;

65. Which of the following is false?

a. Exception handling deals with unusual circumstances during program execution.

Assertions are intended to ensure the correctness of the program.

b. Exception handling addresses robustness whereas assertion addresses correctness.

c. use assertions for argument checking in public methods.

d. Use assertions to reaffirm assumptions.

66. Pressing a button generates a(n) _ _ _ _ _ _ _ _ _ _ _ _ _ event.

a. ItemEvent

b. MouseEvent

c. MouseMotionEvent

d. ActionEvent

67. Which of the following statement is false?

a. If a component can generate an event, any subclass of the component cannot generate the same type of event.

b. All the event classes are subclasses of EventObject.

c. a component on which an event is generated is called the source object.

d. Every GUI component can generate MouseEvent, KeyEvent, FocusEvent, and ComponentEvent.

68. Which of the following statement is false?

a. Each event class has a corresponding listener interface.

b. The listener object's class need not implement the corresponding eventlistener interface.

c. A source may have multiple listeners.

d. The listener object must be registered by the source object.

69. The interface _ _ _ _ _ _ _ _ _ _ should be implemented to listen for a button action event.

a. MouseListener

b. ActionListener

c. FocusListener

d. WindowListener

70. Analyze the following code.

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Test extends JFrame implements ActionListener {
public Test() {
JButton jbtOK = new JButton("OK'');
getContentPane().add(jbtOK);
}
public void actionPerformed(ActionEvent e) {
System.out.println("The OK button is clicked'');
}
public static void main(String[] args) {
JFrame frame = new Test();
frame.setSize(300, 300);
frame.setDefaultCloseOperation(JFrame. );
frame.setVisible(true);
}
}
```

a. The program has a syntax error because no listeners are registered with jbtOK.

b. The program has a runtime error because no listeners are registered with jbtOK.

c. The message ``The OK button is clicked'' is displayed when you click the OK button.
d. The actionPerformed method is not executed when you click the OK button, because no instance of Test is registered with jbtOK.

71. To be a listener for ActionEvent, an object must be an instance of _ _ _ _ _ _ _ _ _ _ _ _ .
a. ActionEvent
b. ActionListener
c. EventObject
d. WindowListener

72. Every event object has the _ _ _ _ _ _ _ _ method.
a. getSource()
b. getActionCommand()
c. getTimeStamp()
d. getWhen()

73. To get the x coordinate of the mouse pointer for the MouseEvent evt, you use _ _ _ _ _ _ _ _ _ _ .
a. evt.getX()
b. Event.getPoint().x
c. evt.getXY( )
d. Event.getPoint(x,y)

74. To detect whether the right button of the mouse is pressed, you use the method _ _ _ _ _ _ _ _ _ _ _ in the MouseEvent object evt.
a. evt.isAltDown()
b. evt.isControlDown()
c. evt.isMetaDown()
d. evt.isShiftDown()

75. The method in the ActionEvent _ _ _ _ _ _ _ _ _ _ returns the action command of the button.
a. getActionCommand()
b. getModifiers()
c. paramString()
d. getID()

76. The handler (e.g., actionPerformed) is a method in _ _ _ _ _ _ _ .
a. a source object
b. a listener object
c. both source and listener object
d. the Object class

77. Analyze the following code.
```
1. import java.awt.*;
2. import java.awt.event.*;
3. import javax.swing.*;
4.
5. public class Test extends JFrame {
6. public Test() {
7. JButton jbtOK = new JButton("OK");
8. JButton jbtCancel = new JButton("Cancel");
9. getContentPane().add(jbtOK);
10. getContentPane().add(jbtCancel);
11. jbtOK.addActionListener(this);
12. jbtCancel.addActionListener(this);
13. }
14.
15. public void actionperformed(ActionEvent e) {
16. System.out.println("A button is clicked");
17. }
18.
19. public static void main(String[] args) {
20. JFrame frame = new Test();
21. frame.setSize(300, 300);
22. frame.setDefaultCloseOperation(JFrame. );
23. frame.setVisible(true);
24. }
25. }
```
a. The program has syntax errors on Lines 11 and 12 because Test does not implement ActionListener.
b. The program has syntax errors on Line 15 because the signature of actionperformed is wrong.
c. The program has syntax errors on Line 20 because new Test() is assigned to frame (a variable of JFrame).
d. The program has runtime errors on Lines 9 and 10 because jbtOK and jbtCancel are added to the same location in the container.

78. To listen to mouse movement events, the listener must implement the _ _ _ _ _ _ _ _ _ _ interface.
a. MouseListener()
b. MouseMotionListener()
c. WindowListener()
d. ComponentListener()

79. Which of the following statement is false?
a. You need not always specify a listener when creating a Timer object.
b. You can add multiple listeners for a Timer object.
c. To stop a timer, invoke timer.stop().
d. To start a timer, invoke timer.start().

80. Which of the following properties are in JApplet?
a. contentPane
b. iconImage
c. resizable
d. title

81. To check whether a DELETE key is pressed or released, which handler should be used?
a. keyActive(KeyEvent e)
b. keyReleased(KeyEvent e)
c. keyTyped(KeyEvent e)
d. keyInactive(KeyEvent e)

82. To be a listener for ActionEvent, an object must be an instance of _ _ _ _ _ _ _ _ _ _ _ _ .
a. ActionEvent
b. ActionListener
c. EventObject
d. WindowListener

83. The listener's method _ _ _ _ _ _ _ _ _ _ _ is invoked when a mouse button is released.
a. public void mousePressed(MouseEvent e)
b. public void mouseReleased(MouseEvent e)
c. public void mouseEntered(MouseEvent e)
d. public void mouseExited(MouseEvent e)

84. To listen to keyboard actions, the listener must implement the _ _ _ _ _ _ _ _ _ _ _

interface.
a. MouseListener
b. KeyListener
c. WindowListener
d. ComponentListener
85. Which of the following statement is false?
a. The keyPressed handler is invoked when a key is pressed.
b. The keyReleased handler is invoked when a key is released.
c. The keyTyped handler is invoked when a key is entered.
d. The keyTyped handler is invoked when a Unicode character is entered.
86. The getKeyCode() method of the KeyEvent returns _ _ _ _ _ _ _ _ _ _ .
a. a character
b. the ASCII code of the character
c. the Unicode code of the character
d. the EBSDIC code of the character
87. What is the value of evt.getKeyCode() or evt.getChar() for the keyTyped() events?
a. A character
b. The ASCII code of the character
c. The Unicode code of the character
d.
88. Which of the following statement is false?
a. You can add a listener in the Timer constructor;
b. You can use the addActionListener method in the Timer class to add a listner.
c. You cannot specify a delay in the Timer constructor;
d. You can specify a delay using the setDelay method;
89. Which of the following are top-level containers (i.e. cannot e embeded in another container)?
a. JOptinPane
b. JApplet
c. JPanel
d. JDialog
90. Which of the following properties in java.awt.Component may effect layout?
a. preferredSize
b. foreground
c. background
d. font
91. Which of the following properties are in FlowLayout?
a. hgap
b. layout
c. visible
d. height
92. Which of the following properties are in GridLayout?
a. hgap
b. alignment
c. layout
d. visible
93. Which of the following properties are in CardLayout?
a. vgap
b. alignment
c. layout
d. visible
94. To use no layout manager in a container c, use _ _ _ _ _ _ _ _ _ .
a. c.setLayout()
b. c.setLayout(new NullLayout())
c. c.setLayout(null)
d. c.setLayout(NullLayout)
95. The preferredSize property is ignored in _ _ _ _ _ _ _ _ _ _ _ _ _ _ .
a. FlowLayout
b. GridLayout
c. BorderLayout
d. For any layout
96. Which of the following properties are in BorderLayout?
a. vgap
b. alignment
c. layout
d. visible
97. Suppose that a container c uses a CardLayout manager p. Which of the following method is not valid?
a. c.p.first()
b. p.last(c)
c. p.next(c)
d. p.previous(c)
98. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ cannot be shared by a GUI component?
a. A GUI component
b. A Color object
c. A Font object
d. A layout object
99. _ _ _ _ _ _ _ _ _ put a flexible spring around a component.
a. BorderLayout
b. GridLayout
c. BoxLayout
d. SpringLayout
100. Which of the following are the invalid methods in JTabbedPane?
a. getTabbedCount( )
b. getTabPlacement( )
c. getTitleAt(index)
d. getToolTipTextAt(index)
101. You can construct a JSplitPane using _ _ _ _ _ _ _ _ _ _ _ _ _ .
a. new JSplitPane()
b. new JSplitPane(Component)
c. new JSplitPane(Component [ ])
d. new JSplitPane(Component, Component)
102. A border object is an instance of _ _ _ _ _ _ _ _ _ .
a. Border
b. BevelBorder
c. TitledBorder
d. LineBorder
103. The Box class uses _ _ _ _ _ _ _ _ _ _ _ _ _ _ .
a. FlowLayout
b. BorderLayout
c. GridLayout
d. BoxLayout
104. _ _ _ _ _ _ _ _ _ _ _ _ is a Swing layout manager that arranges components

on top of each other.
a. OverlayLayout
b. BorderLayout
c. GridLayout
d. BoxLayout
105. Every layout manager is an instance of _ _ _ _ _ _ _ _ _ _ _ .
a. the LayoutManager interface
b. the LayoutManager class
c. the Layout interface
d. the Layout class
106. _ _ _ _ _ _ _ _ _ _ is a Swing layout manager that arranges components in a row or a column.
a. FlowLayout
b. BorderLayout
c. GridLayout
d. BoxLayout
107. You can construct a JScrollPane using _ _ _ _ _ _ _ _ _ _ _ _ .
a. new JScrollPane(JFrame)
b. new JScrollPane(Component)
c. new JScrollPane(Component [ ])
d. new JScrollPane(Component, Component)
108. You can construct a JTabbedPane using _ _ _ _ _ _ _ _ _ _ _ _ .
a. New JTabbedPane( )
b. New JTabbedPane(Component)
c. New JTabbedPane(Component [ ])
d. New JTabbedPane(Component, Component)
109. Which of the following method will not create an EtchedBorder?
a. new EtchedBoarder()
b. new EtchedBorder(Color.YELLOW, Color.RED)
c. BorderFactory.createEtchedBorder()
d. BorderFactory.createEtchedBorder(Color.YELLOW, Color.RED)
110. Which of the following method will not create a BevelBorder?
a. new BevelBorder(BevelBorder.LOWERED)
b. BorderFactory.createBevelBorder(BevelBorder.LOWERED)
c. BorderFactory.createBevelBorder(BevelBorder.RAISED)
d. new BevelBorder()
111. Which of the following method will not create a TitledBorder?
a. new TitledBoarder()
b. new TitledBorder(new BevelBorder(BevelBorder.LOWERED))
c. BorderFactory.createTitledBorder("A title")
d. BorderFactory.createTitledBorder(new BevelBorder(BevelBorder.LOWERED), "A title")
112. A JMenuItem is a subclass of _ _ _ _ _ _ _ _ .
a. JButton
b. AbstractButton
c. JContainer
d. JMenu
113. Which of the following method will not create a LineBorder?
a. new LineBorder()
b. new LineBorder(Color.YELLOW, 3)
c. new LineBorder(Color.YELLOW, 3, true)
d. BorderFactory.createLineBorder(Color.YELLOW)
114. The method _ _ _ _ _ _ _ _ _ _ places a menu bar mb into the frame f.
a. f.setJMenu(mb)
b. f.add(mb)
c. f.setJMenuBar(mb)
d. f.addBar(mb)
115. The method _ _ _ _ _ _ _ _ _ _ places a menu mu into a menu bar mb.
a. mb.add(mu)
b. mb.addMenuItem(mu)
c. mb.addItem(mu)
d. mu.mb.addItem()
116. The method _ _ _ _ _ _ _ _ _ _ places a menu item mi into a menu mu.
a. mu.add(mi)
b. mu.addMenuItem(mi)
c. mu.addItem(mi)
d. mu.mb.addItem()
117. A MenuItem object can generate _ _ _ _ _ _ _ _ _ _ events.
a. ActionEvent
b. ItemEvent
c. ComponentEvent
d. ContainerEvent
118. The method _ _ _ _ _ _ _ _ _ _ separates menu items in a menu mu.
a. mu.add("-")
b. mu.addSeparator()
c. mu.Separator()
d. mu.Separator('-')
119. You cannot set a mnemonic property on _ _ _ _ _ _ _ _ _ _ .
a. a JmenuBar
b. a JMenuItem
c. a JCheckBoxMenuItem
d. a JRadioButtonMenuItem
120. _ _ _ _ _ _ _ _ _ _ _ is the action that causes a popup menu to be displayed.
a. Popup trigger
b. Popup action
c. Popup event
d. Popup reaction
121. A JCheckBoxMenuItem is not a subclass of _ _ _ _ _ _ _ _ .
a. JMenuItem
b. AbstractButton
c. JComponent
d. JMenu
122. A JRadioButtonMenuItem is a subclass of _ _ _ _ _ _ _ _ .
a. JMenuItem
b. AbstractButton
c. JComponent
d. JMenu
123. A JPopupMenu is a subclass of _ _ _ _ _ _ _ _ .
a. JMenuItem
b. AbstractButton

c. JComponent
d. JMenu

124. A JMenu is a not subclass of _ _ _ _ _ _ _ _ _ .
a. JMenuItem
b. AbstractButton
c. JComponent
d. JButton

125. You cannot set an imageIcon property on _ _ _ _ _ _ _ _ _ _ _ _ .
a. a JmenuBar
b. a JMenuItem
c. a JCheckBoxMenuItem
d. a JRadioButtonMenuItem

126. Which of the following statement is false?
a. You can add a JMenuItem to a JMenu.
b. You can add a JMenu to a JMenuItem.
c. You can add a JRadioButtonMenuItem to a JMenuItem.
d. You can add a JRadioButtonMenuItem to a JMenu.

127. Which of the following statement is false?
a. You can add a JMenuItem to a JPopupMenu.
b. You can add a JMenu to a JPopupMenu.
c. You can add a JRadioButtonMenuItem to a JPopupMenu.
d. You can add a JCheckBoxMenuItem to a JPopupMenu.

128. How do you display a JPopupMenu?
a. Invoke the setVisible(true) from a JPopupMenu.
b. Invoke the show() from a JPopupMenu.
c. Invoke the show(Component, x, y) from a JPopupMenu.
d. Add the JPopupMenu to a container.

129. _ _ _ _ _ _ _ _ _ _ _ _ asks a question and requires the user to respond with an appropriate button.
a. A message dialog
b. A confirmation dialog
c. An input dialog
d. An option dialog

130. _ _ _ _ _ _ _ _ _ _ is used to receive input from the user.
a. A message dialog
b. A confirmation dialog
c. An input dialog
d. An option dialog

131. _ _ _ _ _ _ _ _ _ _ _ allows you to create custom buttons.
a. A message dialog
b. A confirmation dialog
c. An input dialog
d. An option dialog

132. Which of the following statements are true?
a. Action is a subclass of ActionListener.
b. JMenu has an add method that enables you to add an instance of ActionListener to the menu.
c. JMenu has an add method that enables you to add an instance of Action to the menu.
d. JToolBar has an add method that enables you to add an instance of ActionListener to the menu.

133. JToolBar uses _ _ _ _ _ _ _ _ _ .
a. FlowLayout
b. SpringLayout
c. BoxLayout
d. GridLayout

134. Which of the following statement is incorrect?
a. You cannot add any GUI component to a JToolBar.
b. A JToolBar is a GUI component, so it can be added to a container.
c. A JToolBar may be floatable.
d. YOu can set orientation of a JToolBar.

135. Which of the following statements are true?
a. JToolBar has an add method that enables you to add an instance of ActionListener to the menu.
b. JToolBar has an add method that enables you to add an instance of Action to the menu.
c. JButton has an add method that enables you to add an instance of ActionListener to the menu.
d. JButton has an add method that enables you to add an instance of Action to the menu.

136. Waits for the user to click the OK button to close the dialog.
a. A message dialog
b. A confirmation dialog
c. An input dialog
d. An option dialog

137. Swing components that don't rely on native GUI are referred to as _ _ _ _ _ _ _ _ _ _ _ _ .
a. lightweight components
b. heavyweight components
c. GUI components
d. Non-GUI components

138. _ _ _ _ _ _ _ _ _ _ are referred to as heavyweight components.
a. AWT components
b. Swing components
c. GUI components
d. Non-GUI components

139. Which of the following are subclasses of java.awt.Component?
a. All Utility classes
b. Swing user interface classes
c. Helper classes such as Color and Font
d. Layout managers

140. What is best to describe the relationship between JComponent and JButton? .
a. Association
b. Aggregation
c. Composition
d. Inheritance

141. Which of the following classes is a heavy weight component?
a. JButton
b. JTextField
c. JPanel
d. JFrame

142. What is best to describe the relationship between a container and a layout manager?
a. Association
b. Aggregation
c. Composition
d. Inheritance

143. Which of the following statement is false?
a. To distinguish new Swing component classes from their AWT counterparts, Swing
GUI component classes are named with a prefix J.
b. All Swing GUI components are lightweight.
c. A user interface object such as (button, list) can appear in one container.
d. A container such as JFrame is also a component.

144. Which component cannot be added to a container? .
a. JPanel
b. JButton
c. JFrame
d. JComponenet

145. What is best to describe the relationship between a container and a SWing GUI object in the container?
a. Association
b. Aggregation
c. Composition
d. Inheritance

146. What is best to describe the relationship between Component and Color?
a. Association
b. Aggregation
c. Composition
d. Inheritance

147. Which of the following classes are not in the java.awt package?
a. Color
b. Font
c. Component
d. JFrame

148. Analyze the following code.
```
import java.awt.*;
import javax.swing.*;
public class Test {
public static void main(String[] args) {
JFrame frame = new JFrame("My Frame'');
frame.getContentPane().add(new
JButton("OK''));
frame.getContentPane().add(new
JButton("Cancel''));
frame.setDefaultCloseOperation(JFrame. );
frame.setSize(200, 200);
frame.setVisible(true);
}
}
```
a. Only button OK is displayed.
b. Only button Cancel is displayed.
c. Both button OK and button Cancel are displayed and button OK is displayed on the
left side of button OK.
d. Both button OK and button Cancel are displayed and button OK is displayed on the
right side of button OK.

149. How many frames are displayed?
```
import javax.swing.*;
public class Test {
public static void main(String[] args) {
JFrame f1 = new JFrame(``My Frame'');
JFrame f2 = f1;
JFrame f3 = f2;
f1.setVisible(true);
f2.setVisible(true);
f3.setVisible(true);
}
}
```
a. 1.
b. 2.
c. 3.
d. 0.

150. Analyze the following code.
```
import java.awt.*;
import javax.swing.*;
public class Test {
public static void main(String[] args) {
Component c = new JButton("OK'');
JFrame frame = new JFrame("My Frame'');
frame.add(c);
frame.setDefaultCloseOperation(JFrame. );
frame.setVisible(true);
}
}
```
a. You cannot assign a JButton to a variable of java.awt.Component.
b. You can only add c to a container because c's type is Component.
c. You cannot add a Swing component directly to a JFrame. Instead, you have to add it to a JFrame's contentPane using frame.getContentPane().add(c).
d. You cannot create a JFrame using new JFrame("My Frame'').

151. Which of the following statement is for terminating the program when closing the frame?
a. frame.setDefaultCloseOperation(JFrame.)
b. frame.setDefaultCloseOperation(null)
c. frame.setDefaultCloseOperation(JFrame.)
d. frame.setDefaultCloseOperation(JFrame.)

152. Which of the following statement is for placing the frame's upper left corner to (200, 100)?
a. frame.setLocation(100, 100)
b. frame.setLocation(100, 200)
c. frame.setLocation(200, 100)
d. frame.setLocation(200, 200)

153. The default layout out of a contentPane in a JFrame is _ _ _ _ _ _ _ _ _ _ .
a. FlowLayout
b. GridLayout
c. BorderLayout
d. GridBagLayout

154. _ _ _ _ _ _ _ _ _ _ _ cannot create a color object.
a. new Color(0, 0, 0)
b. new Color(0, 266, 0)
c. new Color(255, 255, 255)

d. new Color(1, 2, 3)

155. How many frames are displayed?

```
import javax.swing.*;
public class Test extends JFrame {
public static void main(String[] args) {
JFrame f1 = new Test();
JFrame f2 = new Test();
JFrame f3 = new Test();
f1.setVisible(true);
f2.setVisible(true);
f3.setVisible(true);
}
}
```

a. 1.
b. 2.
c. 3.
d. 0.

156. What should you use to position a Button within an application Frame so that the size of the Button is NOT affected by the Frame size?

a. a FlowLayout
b. a GridLayout
c. the center area of a BorderLayout
d. the East or West area of a BorderLayout

157. To set a FlowLayout in panel jp, you can use the method _ _ _ _ _ _ _ _ _ _ .

a. jp.setLayout(new FlowLayout());
b. jp.setLayout(new FlowLayout(FlowLayout));
c. jp.setLayout(new FlowLayout(FlowLayout.center));
d. jp.setLayout(FlowLayout());

158. The method _ _ _ _ _ _ _ _ _ _ sets the font (Helvetica, 20-point bold) in component C.

a. c.setFont(new Font("Helvetica", Font.bold, 20))
b. c.setFont(new Font("helvetica", BOLD, 20))
c. c.setFont(Font("Helvetica", Font.BOLD, 20))
d. c.setFont(new Font("Helvetica", Font.BOLD, 20))

159. The method _ _ _ _ _ _ _ _ _ _ can be used to get the width of the component c.

a. c.getDimension()
b. c.getWidth()
c. c.getHeight()
d. c.getSize()

160. You should override the _ _ _ _ _ _ _ _ _ _ method to draw things on a Swing component.

a. repaint()
b. update()
c. paintComponent()
d. init()

161. To add a component c to a JPanel p, use _ _ _ _ _ _ _ _ _ .

a. p.add(c)
b. p.getContentPane(c)
c. p.insert(c)
d. p.append(c)

162. The method _ _ _ _ _ _ _ _ _ _ sets the background color to yellow in JFrame f.

a. setBackground(Color.yellow)
b. f.setBackground(Color.YELLOW)
c. setBackground(Color.YELLOW)
d. f.setBackGround(Color.yellow)

163. The method _ _ _ _ _ _ _ _ _ _ sets the foreground color to yellow in JFrame f.

a. setForeground(Color.yellow)
b. setForeground(Color.YELLOW)
c. f.setForeground(Color.yellow)
d. f.setForeGround(Color.yellow)

164. To specify a font to be bold and italic, use the font style value _ _ _ _ _ _ _ _ _ _

a. Font.PLAIN
b. Font.BOLD
c. Font.ITALIC
d. Font.BOLD + Font.ITALIC

165. The default layout out of a JPanel is _ _ _ _ _ _ _ _ _ _ _ _ .

a. FlowLayout
b. GridLayout
c. BorderLayout
d. default Layout

166. To create a JPanel of the BorderLayout, use _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ .

a. JPanel p = new JPanel()
b. JPanel p = new JPanel(BorderLayout());
c. JPanel p = new JPanel(new BorderLayout());
d. JPanel p = new JPanel().setLayout(new BorderLayout());

167. Analyze the following code.

```
import java.awt.*;
import javax.swing.*;
public class Test {
public static void main(String[] args) {
JFrame frame = new JFrame("My Frame'');
frame.getContentPane().add(new
MyDrawing("Welcome to Java!''));
frame.setSize(300, 300);
frame.setDefaultCloseOperation(JFrame. );
frame.setVisible(true);
frame.setVisible(true);
}
}
class MyDrawing extends JPanel {
String message;
public MyDrawing(String message) {
this.message = message;
}
public void paintcomponent(Graphics g) {
super.paintComponent(g);
g.drawString(message, 20 ,20);
}
}
```

a. The program runs fine and displays Welcome to Java!
b. The program has a syntax error because the paintcomponent should be spelled as paintComponent.
c. The program has a runtime error because the paintcomponent should be spelled as paintComponent.

d. The program runs, but it does not display the message.

168. The coordinate of the upper-left corner of a frame is _ _ _ _ _ _ _ _ _ _ .

a. (0, 0)
b. (25, 25)
c. (100, 100)
d. (10, 10)

169. Given a Graphics object g, to draw a filled arc with radius 20 centered at (50, 50) and start angle 0 and spanning angle 90, you use _ _ _ _ _ _ _ _ _ _

a. g.fillArc(50, 50, 40, 40, 0, Math.toRadian(90))
b. g.fillArc(50, 50, 40, 40, 0, 90)
c. g.fillArc(30, 30, 40, 40, 0, Math.toRadian(90))
d. g.fillArc(30, 30, 40, 40, 0, 90)

170. Given a Graphics object g, to draw a polygon to connect points (3, 3), (4, 10), (10, 20), (2, 100), you use _ _ _ _ _ _ _ _ _ _

a. g.drawPolyline(new int[]{3, 4, 10, 2}, new int[]{3, 10, 20, 100}, 4)
b. g.drawPolyline({3, 4, 10, 2}, {3, 10, 20, 100}, 4)
c. g.drawPolygon(new int[]{3, 4, 10, 2}, new int[]{3, 10, 20, 100}, 4)
d. g.drawPolygon({3, 4, 10, 2}, {3, 10, 20, 100}, 4)

171. Analyze the following code.

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Test1 extends JFrame {
public Test1() {
getContentPane().add(new MyCanvas());
}
public static void main(String[] args) {
JFrame frame = new Test1();
frame.setSize(300, 300);
frame.setDefaultCloseOperation(JFrame. );
frame.setVisible(true);
}
}
class MyCanvas extends JPanel {
private String message;
public void setMessage(String message) {
this.message = message;
}
public void paintComponent(Graphics g) {
super.paintComponent(g);
g.drawString(message, 20, 20);
}
}
```

a. The program runs fine and displays nothing since you have not set a string value.
b. The program would display Welcome to Java! if you replace new MyCanvas() by new MyCanvas(``Welcome to Java!'').
c. The program has a syntax error because new Test1() is assigned to frame.
d. The program has a NullPointerException since message is null when g.drawString(message, 20, 20) is executed.

172. Analyze the following code.

```
import java.awt.*;
import javax.swing.*;
public class Test extends JFrame {
public Test() {
getContentPane().add(new
MyDrawing("Welcome to Java!''));
}
public static void main(String[] args) {
JFrame frame = new JFrame();
frame.setSize(300, 300);
frame.setDefaultCloseOperation(JFrame. );
frame.setVisible(true);
}
}
class MyDrawing extends JPanel {
String message;
public MyDrawing(String message) {
this.message = message;
}
public void paintComponent(Graphics g) {
super.paintComponent(g);
g.drawString(message, 20 ,20);
}
}
```

a. The program runs fine and displays Welcome to Java!
b. The program would display Welcome to Java! if new JFrame() is replaced by Test().
c. The program would display Welcome to Java! if new JFrame() is replaced by new Test().
d. The program would display Welcome to Java! if new JFrame() is replaced by new Test(``My Frame'').

173. Which of the following statement is false?

a. You may create a Graphics object using new Graphics().
b. Whenever a GUI component is displayed, its Graphics object is automatically created.
c. The paintComponent method is automatically invoked by the JVM. You should never invoke it directly.
d. Invoking repaint() causes paintComponent to be invoked by the JVM.

174. Given a Graphics object g, to draw a line from the upper left corner to the bottom right corner, you use _ _ _ _ _ _ _ _ _ _

a. g.drawLine(0, 0, 100, 100)
b. g.drawLine(0, 0, getWidth(), getHeight())
c. g.drawLine(0, 0, getHeight(), getHeight())
d. g.drawLine(0, 0, getWidth(), getWidth())

175. Given a Graphics object g, to draw an outline of a rectangle of width 20 and height 50 with the upper-left corner at (20, 20), you use _ _ _ _ _ _ _ _ _ _

a. g.drawRect(20, 50, 20, 20)
b. g.drawRectFill(20, 20, 20, 50)
c. g.drawRect(20, 20, 20, 50)
d. g.drawRectFill(20, 50, 20, 20)

176. Given a Graphics object g, to draw an circle with radius 20 centered at (50, 50), you use _ _ _ _ _ _ _ _ _ _

a. g.drawOval(50, 50, 20, 20)

b. g.drawOval(50, 50, 40, 40)
c. g.drawOval(30, 30, 20, 20)
d. g.drawOval(30, 30, 40, 40)

177. Which of the following statement is false?
a. You can create a FontMetric using new FontMetrics()
b. You can obtain a FontMetrics from a Font object using the getFontMetrics() method.
c. A font determines the font metrics.
d. You can obtain the leading, ascent, descent, and height for a font from a FontMetrics object.

178. A listener of a JSpinner must implement _ _ _ _ _ _ _ _ _ .
a. the java.awt.ActionListener interface
b. the java.awt.ItemListener interface
c. javax.swing.event.ChangeListener
d. javax.swing.event.AdjustmentListener

179. Which of the following statement is false ?
a. The model-view-controller (MVC) approach is a way of developing components by separating data storage and handling from the visual representation of the data.
b. The model-view-controller (MVC) approach makes multiple views possible so that data can be shared through the same model.
c. The model-view-controller (MVC) approach simplifies the task of writing complex applications and makes the components scalable and easy to maintain. Changes can be made to the view without affecting the model, and vice versa.
d. Models and views are asynchronized to ensure that a view displays the data consistently.

180. Which of the following are the properties in SpinnerNumberModel?
a. Size
b. nextVal
c. previousValue
d. prevVal

181. If you create a JSpinner object without specifying a model, the spinner displays _ _ _ _ _ _ _ .
a. a sequence of integers
b. a sequence of double values
c. a sequence of positive integers
d. a sequence of non-negative integers

182. JSpinner fires _ _ _ _ _ _ _ _ _ when a new value is selected.
a. java.awt.ActionEvent
b. java.awt.event.ItemEvent
c. javax.swing.event.ChangeEvent
d. javax.swing.event.AdjustmentEvent

183. The data in DefaultListModel are stored in _ _ _ _ _ _ _ _ _ _ .
a. an array.
b. an ArrayList
c. a LinkedList
d. a Vector

184. The data in DefaultListModel are stored in _ _ _ _ _ _ _ _ _ _ .
a. an array.
b. an ArrayList
c. a LinkedList
d. a Vector

185. The component for storing and handling data, known as _ _ _ _ _ _ _ _ _ , contains the actual contents of the component.
a. a model
b. a view
c. a controller
d. a Container

186. _ _ _ _ _ _ _ _ _ uses a java.util.List to store a sequence of custom defined data in the model.
a. SpinnerListModel
b. SpinnerNumberModel
c. SpinnerDateModel
d. SpinnerModel

187. JList fires _ _ _ _ _ _ _ _ _ when a new item is selected.
a. java.awt.ActionEvent
b. java.awt.event.ItemEvent
c. javax.swing.event.ChangeEvent
d. javax.swing.event.AdjustmentEvent

188. A listener of a JList must implement _ _ _ _ _ _ _ _ _ _ .
a. the java.awt.ActionListener interface
b. the java.awt.ItemListener interface
c. javax.swing.event.ChangeListener
d. javax.swing.event.AdjustmentListener

189. DefaultListCellRenderer can display _ _ _ _ _ _ _ _ _ _ .
a. only a string
b. only an icon
c. both string and icon
d. a string or an icon

190. _ _ _ _ _ _ _ _ _ are not the property of JList.
a. visibleRowCount
b. maximumRowCount
c. editable
d. itemCount

191. JComboBox fires _ _ _ _ _ _ _ _ when a new item is selected.
a. java.awt.ActionEvent
b. java.awt.ItemEvent
c. javax.swing.event.ChangeEvent
d. javax.swing.event.AdjustmentEvent

192. You can obtain the server's hostname by invoking _ _ _ _ _ _ _ _ _ on an applet.
a. getCodeBase().host()
b. getCodeBase().getHost()
c. getCodeBase().hostName()
d. getCodeBase().getHostName()

193. To obtain an ObjectInputStream from a socket, use _ _ _ _ _ _ _ _ .
a. socket.getInputStream()
b. socket.getObjectStream()
c. socket.getObjectInputStream()
d. socket.objectInputStream()

194. When a client requests connection to a server that has not yet started, _ _ _ _ _ _ _ _ _ .

a. java.net.BindException occurs.
b. java.net.ConnectionException occurs.
c. the client is blocked until the server is started.
d. the client encounters a fatal error and must be terminated.

195. To create an InputStream on a socket s, you use _ _ _ _ _ _ _ _ _ _ .
a. InputStream in = new InputStream(s);
b. InputStream in = s.getInputStream();
c. InputStream in = s.obtainInputStream();
d. InputStream in = s.getStream();

196. You can invoke _ _ _ _ _ _ _ _ _ _ _ on a Socket object, say socket, to obtain an InetAddress object.
a. socket.InetAddress();
b. socket.getInetAddress();
c. socket.obtainInetAddress();
d. socket.retrieveInetAddress();

197. The _ _ _ _ _ _ _ _ _ _ _ _ method in the InetAddress class returns the IP address.
a. getIP()
b. getIPAddress()
c. getHostAddress()
d. getAddress()

198. To obtain an ObjectOutputStream from a socket, use _ _ _ _ _ _ _ _ .
a. socket.getOutputStream()
b. socket.getObjectStream()
c. socket.getObjectOutputStream()
d. socket.objectOutputStream()

199. When creating a server on a port that is already in use, _ _ _ _ _ _ _ _ _ _ .
a. java.net.BindException occurs.
b. the server is created with no problems.
c. the server is blocked until the port is available.
d. the server encounters a fatal error and must be terminated.

200. When creating a client on a server port that is already in use, _ _ _ _ _ _ _ _ _ _ .
a. the client can connect to the server regardless of whether the port is in use.
b. java.net.BindException occurs.
c. the client is blocked until the port is available.
d. the client encounters a fatal error and must be terminated.

201. The server listens for a connection request from a client using the following statement:
a. Socket s = new Socket(ServerName, port);
b. Socket s = serverSocket.accept()
c. Socket s = serverSocket.getSocket()
d. Socket s = new Socket(ServerName);

202. The client requests a connection to a server using the following statement:
a. Socket s = new Socket(ServerName, port);
b. Socket s = serverSocket.accept();
c. Socket s = serverSocket.getSocket();
d. Socket s = new Socket(ServerName);

203. A ServerSocket can connect to _ _ _ _ _ _ _ clients.
a. one
b. two
c. ten
d. unlimited number of

204. To create an InputStream to read from a file on a Web server, you use the class _ _ _ _ _ _ _ _ _ .URL;
a. URL;
b. Server;
c. ServerSocket;
d. ServerStream;

205. To create an InputStream to read from a file on a Web server, you use the method _ _ _ _ _ _ _ _ _ _ in the URL class.
a. getInputStream();
b. obtainInputStream();
c. openStream();
d. connectStream();