Chapter9: Interfaces

1. Which of the following statements about a Java interface is NOT true?
a) A Java interface defines a set of methods that are required.
b) A Java interface must contain more than one method.
c) A Java interface specifies behavior that a class will implement.
d) All methods in a Java interface must be abstract.
Answer: b

2. A method that has no implementation is called a/an _____ method.
a) interface b) implementation
c) overloaded d) abstract
Answer: d

3. Which statement about methods in an interface is true?
a) All methods in an interface are automatically private.
b) All methods in an interface are automatically public.
c) All methods in an interface are automatically static.
d) All methods in an interface must be explicitly declared as private or public.
Answer: b

4. Which of the following statements about abstract methods is true?
a) An abstract method has a name, parameters, and a return type, but no code in the body of the method.
b) An abstract method has parameters, a return type, and code in its body, but has no defined name.
c) An abstract method has a name, a return type, and code in its body, but has no parameters.
d) An abstract method has only a name and a return type, but no parameters or code in its body.
Answer: a

5. Which of the following statements about an interface is true?
a) An interface has methods and instance variables.
b) An interface has methods but no instance variables.
c) An interface has neither methods nor instance variables.
d) An interface has both public and private methods.
Answer: b

6. Which of the following statements about interfaces is NOT true?
a) Interfaces can make code more reusable.
b) Interface types can be used to define a new reference data type.
c) Interface types can be used to express common operations required by a service.
d) Interfaces have both private and public methods.
Answer: d

7. To use an interface, a class header should include which of the following?
a) The keyword `extends` and the name of an abstract method in the interface
b) The keyword `extends` and the name of the interface
c) The keyword `implements` and the name of an abstract method in the interface
d) The keyword `implements` and the name of the interface
Answer: d

8. _____ can reduce the coupling between classes.
a) Static methods b) Abstract methods
c) Interfaces d) Objects
Answer: c

9. Consider the following code snippet:
```
public class Inventory implements Measurable
{
   . . .
   public double getMeasure();
   {
      return onHandCount;
   }
}
```
Why is it necessary to declare `getMeasure` as public ?
a) All methods in a class are not public by default.
b) All methods in an interface are private by default.
c) It is necessary only to allow other classes to use this method.
d) It is not necessary to declare this method as public.
Answer: a

10. Which of the following statements about interfaces is NOT true?
a) Interfaces can make code more reusable.
b) An interface provides no implementation.
c) A class can implement only one interface type.
d) Interfaces can reduce the coupling between classes.
Answer: c

11. ____ methods must be implemented when using an interface.
a) Abstract.
b) Private.
c) Public.
d) Static
Answer: a

12. Suppose you are writing an interface called `Resizable`, which includes one void method called `resize`.

```
public interface Resizable
{
   _________________________
}
```

Which of the following can be used to complete the interface declaration correctly?
a) `private void resize();`
b) `protected void resize();`
c) `void resize();`
d) `public void resize() { System.out.println("resizing ..."); }`
Answer: c

13. Consider the following declarations:

```
public interface Encryptable
{
   void encrypt(String key);
}
public class SecretText implements Encryptable
{
   private String text;

   _____________________________
   {
      // code to encrypt the text using encryption key goes here
   }
}
```

Which of the following method headers should be used to complete the `SecretText` class?

a) `public void encrypt()`
b) `public void encrypt(String aKey)`
c) `public String encrypt(String aKey)`
d) `void encrypt(String aKey)`
Answer: b

14. Consider the following code snippet:

```
public class Inventory implements Measurable
{
   . . .
   double getMeasure();
   {
      return onHandCount;
   }
}
```

What is wrong with this code?

a) The `getMeasure()` method must be declared as `private`.
b) The `getMeasure()` method must include the `implements` keyword.
c) The `getMeasure()` method must be declared as `public`.
d) The `getMeasure()` method must not have any code within it.
Answer: c

15. Consider the following code snippet:
```
public interface Sizable
{
   int LARGE_CHANGE = 100;
   int SMALL_CHANGE = 20;

   void changeSize();
}
```
Which of the following statements is true?
a) `LARGE_CHANGE` and `SMALL_CHANGE` are automatically `public static final`.
b) `LARGE_CHANGE` and `SMALL_CHANGE` are instance variables
c) `LARGE_CHANGE` and `SMALL_CHANGE` must be defined with the keywords `private static final`.
d) `LARGE_CHANGE` and `SMALL_CHANGE` must be defined with the keywords `public static final`.
Answer: a

16. Consider the following code snippet:
```
public class Inventory implements Measurable
{
   . . .
   double getMeasure();
   {
      return onHandCount;
   }
}
```
The compiler complains that the `getMeasure` method has a weaker access level than the `Measurable` interface. Why?
a) All of the methods in a class have a default access level of package access, while the methods of an interface have a default access level of private.
b) All of the methods in a class have a default access level of package access, while the methods of an interface have a default access level of public.
c) The variable `onHandCount` was not declared with public access.
d) The `getMeasure` method was declared as private in the `Measurable` interface.
Answer: b

17. Which of the following is true regarding a class and interface types?
a) You can convert from a class type to any interface type that is in the same package as the class.
b) You can convert from a class type to any interface type that the class implements.
c) You can convert from a class type to any interface type that the class defines.
d) You cannot convert from a class type to any interface type.
Answer: b

18. Consider the following code snippet.
```
public interface Measurable
{
   double getMeasure();
}
public class Coin implements Measurable
{
   public double getMeasure()
   {
      return value;
   }
   ...
}
public class DataSet
{
   ...
   public void add()
   {
      ...
   }
}
public class BankAccount
{
   ...
   public void add()
   {
      ...
   }
}
```

Which of the following statements is correct?
a)
```
Coin dime = new Coin(0.1, "dime");
Measurable x = dime;
```
b)
```
Coin dime = new Coin(0.1, "dime");
Dataset x = dime;
```
c)
```
Coin dime = new Coin(0.1, "dime");
DataSet x == (Measureable)dime;
```
d)
```
Coin dime = new Coin(0.1, "dime");
BankAccount x = dime;
```
Answer: a

19. Which of the following statements about converting between types is true?
a) When you cast number types, you take a risk that an exception will occur.
b) When you cast number types, you will not lose information.
c) When you cast object types, you take a risk of losing information.
d) When you cast object types, you take a risk that an exception will occur.
Answer: d

20. Which of the following statements about interfaces is true?
a) You can define an interface variable that refers to an object of any class in the same package.
b) You cannot define a variable whose type is an interface.
c) You can instantiate an object from an interface class.
d) You can define an interface variable that refers to an object only if the object belongs to a class that implements the interface.
Answer: d

21. You have created a class named `Motor` that implements an interface named `Measurable`. You have declared a variable of type `Measureable` named `motorTemperature`. Which of the following statements is true?
a) The object to which `motorTemperature` refers has type `Measurable`.
b) The object to which `motorTemperature` refers has type `Motor`.
c) This declaration is illegal.
d) You must construct a `motorTemperature` object from the `Measurable` interface.
Answer: b

22. ____ occurs when a single class has several methods with the same name but different parameter types.
a) Casting
b) Polymorphism
c) Overloading
d) Instantiation
Answer: c

23. Consider the following code snippet:
```
public class Demo
{
   public static void main(String[] args)
   {
      Point[] p = new Point[4];

      p[0] = new Colored3DPoint(4, 4, 4, Color.BLACK);
      p[1] = new ThreeDimensionalPoint(2, 2, 2);
      p[2] = new ColoredPoint(3, 3, Color.RED);
      p[3] = new Point(4, 4);

      for (int i = 0; i < p.length; i++)
      {
         String s = p[i].toString();
         System.out.println("p[" + i + "] : " + s);
      }
      return;
   }
}
```
This code is an example of ____.
a) overloading b) callback
c) early binding d) polymorphism
Answer: d

24. If you have multiple classes in your program that have implemented the same interface in different ways, how is the correct method executed?
a) The Java virtual machine must locate the correct method by looking at the class of the actual object.
b) The compiler must determine which method implementation to use.
c) The method must be qualified with the class name to determine the correct method.
d) You cannot have multiple classes in the same program with different implementations of the same interface.
Answer: a

25. Consider the following declarations:
```
public interface Displayable
{
   void display();
}
public class Picture implements Displayable
{
   private int size;
   public void increaseSize()
   {
      size++;
   }

   public void decreaseSize()
   {
      size--;
   }
   public void display()
   {
      System.out.println(size);
   }
   public void display(int value)
   {
      System.out.println(value * size);
   }
}
```
What method invocation can be used to complete the code segment below?
```
Displayable picture = new Picture();
picture.__________________;
```

a) `increaseSize()`
b) `decreaseSize()`
c) `display()`
d) `display(5)`
Answer: c

26. Which of the following can potentially be changed when implementing an interface?
a) The parameters of a method in the interface.
b) The name of a method in the interface.
c) The return type of a method in the interface.
d) You cannot change the name, return type, or parameters of a method in the interface.
Answer: d

27. Consider the following class:
```
public class Player implements Comparable
{
   private String name;
   private int goalsScored;
   // other methods go here
   public int compareTo(Object otherObject)
   {
      __________________________________
      return (goalsScored - otherPlayer.goalsScored);
   }
}
```
What statement can be used to complete the `compareTo()` method?
a) `Player otherPlayer = otherObject;`
b) `Object otherPlayer = otherObject;`
c) `Player otherPlayer = (Player) otherObject;`
d) `Object otherPlayer = (Player) otherObject;`
Answer: c

28. The method below is designed to print the smaller of two values received as arguments. Select the correct expression to complete the method.

```
public void showSmaller(Comparable value1, Comparable value2)
{
   if ( _________________________ )
      System.out.println(value1 + " is smaller.");
   else
      System.out.println(value2 + " is smaller.");
}
```

a) `value1 < value2`
b) `value1.compareTo(value2) > 0`
c) `value1.compareTo(value2) == 0`
d) `value1.compareTo(value2) < 0`
Answer: d

29. Which of the following statements about a callback is NOT true?
a) A callback can allow you to implement a new method for a class that is not under your control.
b) A callback can be implemented using an interface.
c) A callback is a mechanism for specifying code to be executed later.
d) A callback method declared in an interface must specify the class of objects that it will manipulate.
Answer: d

30. In Java, ____ can be used for callbacks.
a) Objects
b) Interfaces
c) Classes
d) Operators
Answer: b

31. You wish to implement a callback method for an object created from a system class that you cannot change. What approach does the textbook recommend to accomplish this?

a) Create a new class that mimics the system class.
b) Extend the system class.
c) Use an inner class in the interface.
d) Use a helper class that implements the callback method.
Answer: d

32. Which of the following statements about helper classes is true?
a) Helper classes must be inner classes.
b) Helper classes must implement interfaces.
c) Helper classes help reduce coupling.
d) Helper classes cannot contain instance variables.
Answer: c

33. Consider the following declarations:

```
public interface Measurer
{
   int measure(Object anObject);
}

public class StringLengthMeasurer implements Measurer
{
   public int measure(_________________)
   {
      String str = (String) anObject;
      return str.length();
   }
}
```

What parameter declaration can be used to complete the callback `measure` method?
a) `Object anObject`
b) `String anObject`
c) `Object aString`
d) `String aString`
Answer: a

34. Assuming that interface `Resizable` is declared elsewhere, consider the following class declaration:

```
public class InnerClassExample
{
   public static void main(String[] args)
   {
      class SizeModifier implements Resizable
      {
         // class methods
      }
      __________________________         // missing statement
   }
}
```

Which of the following declarations can be used to complete the `main` method?
a) `Resizable something = new Resizable();`
b) `Resizable something = new SizeModifier();`
c) `Resizable something = new InnerClassExample();`
d) `SizeModifier something = new Resizable();`
Answer: b

35. Which of the following statements about an inner class is true?
a) An inner class can only be defined within a specific method of its enclosing class.
b) An inner class can only be defined outside of any method within its enclosing class.
c) An inner class must implement an interface.
d) An inner class may be anonymous.
Answer: d

36. A/an ____ class defined in a method signals to the reader of your program that the class is not interesting beyond the scope of the method.
a) A class cannot be defined within a method
b) abstract
c) interface
d) inner
Answer: d

37. Which of the following statements about an inner class is true?

a) An inner class that is defined inside a method is publicly accessible.
b) An inner class that is defined inside a method is not publicly accessible.
c) An inner class that is defined inside an enclosing class but outside of its methods is not available to all methods of the enclosing class.
d) An inner class is used for a utility class that should be visible elsewhere in the program.
Answer: b

38. Which of the following is a good use for an anonymous class?

a) Use an anonymous class to implement polymorphism.
b) Use an anonymous class to implement a callback.
c) Use an anonymous class when all methods of the class will be static methods.
d) Use an anonymous class when the program will only need one object of the class.
Answer: d

39. Consider the following code snippet:

```
myImage.add(new Rectangle(10,10,10,10));
```

This code is an example of using ____.
a) An anonymous class.
b) An anonymous object.
c) An abstract object.
d) An abstract class.
Answer: b

40. Which of the following statements about a mock class is true?
a) A mock class does not provide an implementation of the services of the actual class.
b) A mock class provides a complete implementation of the services of the actual class.
c) A mock class provides a simplified implementation of the services of the actual class.
d) A mock class must be an interface.
Answer: c

41. What role does an interface play when using a mock class?

a) The mock class should be an interface that will be implemented by the real class.
b) The real class should be an interface that will be implemented by the mock class.
c) Interfaces are not involved when using mock classes.
d) An interface should be implemented by both the real class and the mock class to guarantee that the mock class accurately simulates the real class when used in a program.
Answer: d

42. Which of the following statements about events and graphical user interface programs is true?

a) Your program must instruct the Java window manager to send it notifications about specific types of events to which the program wishes to respond.
b) The Java window manager will automatically send your program notifications about all events that have occurred.
c) Your program must respond to notifications of all types of events that are sent to it by the Java window manager.
D) Your program must override the default methods to handle events.
Answer: a

43. Consider the following class:
```
public class ClickListener implements ActionListener
{
   __________________________________________
   {
      System.out.println("mouse event ...");
   }
}
```
Which of the following method headers should be used to complete the `ClickListener` class?
a) `public void actionPerformed(ActionEvent event)`
b) `public void actionPerformed(ClickListener event)`
c) `public void actionPerformed()`
d) `public void actionPerformed(ActionListener event)`
Answer: a

44. ____ are generated when the user presses a key, clicks a button, or selects a menu item.
a) Listeners
b) Interfaces.
c) Events.
d) Errors.
Answer: c

45. A/an ____ belongs to a class whose methods describe the actions to be taken when a user clicks a user-interface graphical object.
a) Event listener
b) Event source
c) Action listener
d) Action method
Answer: a

46. Which of the following is an event source?
a) A JButton object.
b) An event listener.
c) An inner class.
d) An event adapter.
Answer: a

47. To respond to a button event, a listener must supply instructions for the ____ method of the `ActionListener` interface.
a) `actionEvent`.
b) `actionPerformed`
c) `eventAction`
d) `eventResponse`
Answer: b

48. To associate an event listener with a `JButton` component, you must use the ___ method of the `JButton` class.
a) `addEventListener`.
b) `addActionListener`.
c) `addButtonListener`.
d) `addListener`
Answer: b

49. The methods of a/an ____ describe the actions to be taken when an event occurs.

a) event source
b) event listener
c) event interface
d) action source
Answer: b

50. When an event occurs, the event source notifies all ____.

a) components
b) panels
c) interfaces
d) event listeners
Answer: d

51. Which container is used to group multiple user-interface components together?

a) text area
b) table
c) panel
d) rectangle
Answer: c

52. Which of the following statements will compile without error?

a)
```
public interface AccountListener()
{
   void actionPerformed(AccountEvent event);
}
```
b)
```
public interface AccountListener()
{
   void actionPerformed(AccountEvent);
}
```
c)
```
public interface AccountListener
{
   void actionPerformed(AccountEvent event);
}
```
d)
```
public abstract AccountListener
{
   void actionPerformed(AccountEvent event);
}
```
Answer: c

53. Which of the following statements about listener classes is true?

a) A listener class cannot be declared as an anonymous inner class.
b) A listener class cannot be declared as an inner class.
c) A listener class must be declared as an inner class.
d) A listener class must be declared as an anonymous inner class.
Answer: a

54. Consider the following code snippet:

```
JFrame frame = new JFrame();
JPanel panel = new JPanel
```

Which statement would add the panel to the frame?

a) `frame.add(panel);`
b) `frame.add(JPanel panel);`
c) `frame.addComponent(panel);`
d) `frame.addComponent(JPanel panel);`
Answer: a

55. Consider the following code snippet:

```
import ______________________
import java.awt.event.ActionListener;

/**
   An action listener that prints.
*/
public class ClickListener implements ActionListener
{
   public void actionPerformed(ActionEvent event)
   {
      System.out.println("I was clicked.");
   }
}
```

Which of the following statements will complete this code?

a) `java.swing.event.ActionEvent;.`
b) `javax.swing.event.ActionEvent;`
c) `javax.awt.event.ActionEvent;`
d) `java.awt.event.ActionEvent;`
Answer: d

56. Event listeners are often installed as ____ classes so that they can have access to the surrounding fields, methods, and final variables.

a) Inner
b) Interface
c) Abstract
d) Helper
Answer: a

57. Which of the following statements about an inner class is true?

a) An inner class may not be declared within a method of the enclosing scope.
b) An inner class may only be declared within a method of the enclosing scope.
c) An inner class can access variables from the enclosing scope only if they are passed as constructor or method parameters.
d) The methods of an inner class can access variables declared in the enclosing scope.
Answer: d

58. Consider the following code snippet:

```
public class ClickListener implements ActionListener
{
   public void actionPerformed(ActionEvent event)
   {
      System.out.println("I was clicked.");
   }
}
public class ButtonTester
{
   public static void main(String[] args)
   {
      JFrame frame = new JFrame();
      JButton button = new JButton("Click me!");
      frame.add(button);

      ActionListener listener = new ClickListener();
      button.addActionListener(listener);
      ...
   }
}
```

Which of the following statements is correct?
a) Class `ButtonTester` is an interface type.
b) A `ClickListener` object can be added as a listener for the action events that buttons generate.
c) Class `ClickListener` is an interface type.
d) Class `ButtonTester` implements an interface type.
Answer: b

59. An inner class can access local variables from the enclosing scope only if they are declared as ____.

a) private
b) public
c) static
d) final
Answer: b

60. Which of the following code statements creates a graphical button which has "Calculate" as its label ?

a) `Button JButton = new Button("Calculate").`
b) `button = new Button(JButton, "Calculate").`
c) `button = new JButton("Calculate").`
d) `JButton button = new JButton("Calculate").`
Answer: d

61. To build a user interface that contains graphical components, the components ____.

a) Must be added directly to a frame component.
b) Must each be added to a separate panel.
c) Must be added to a panel that is contained within a frame.
d) Must be added to a frame that is contained within a panel.
Answer: c

62. How do you specify what the program should do when the user clicks a button?

a) Specify the actions to take in a class that implements the `ButtonListener` interface.
b) Specify the actions to take in a class that implements the `ButtonEvent` interface .
c) Specify the actions to take in a class that implements the `ActionListener` interface.
d) Specify the actions to take in a class that implements the `EventListener` interface.
Answer: c

63. A(n) ____ has an instance method `addActionListener()` for specifying which object is responsible for implementing the action associated with the object.

a) `JFrame`
b) `JSlider`
c) `JButton`
d) `JLabel`
Answer: c

64. Consider the following code snippet:

```
public static void main(String[] args)
{
   Order myOrder = new Order();
   JButton button = new JButton("Calculate");
   JLabel label = new JLabel("Total amount due");
   . . .
   class MyListener implements ActionListener
   {
      public void actionPerformed(ActionEvent event)
      {
         label.setText("Total amount due " + myOrder.getAmountDue());
      }
   }
}
```

What is wrong with this code?

a) `label` must be declared as final.
b) `myOrder` must be declared as final
c) `label` and `myOrder` must be declared as final
d) `label` and `button` must be declared as final.
Answer: c

65. Consider the following code snippet:
```
public static void main(String[] args)
{
  final Order myOrder = new Order();
  JButton button = new JButton("Calculate");
  final JLabel label = new JLabel("Total amount due");
  . . .
  class MyListener implements ActionListener
  {
     public void actionPerformed(ActionEvent event)
     {
       . . .
     }
  }
}
```

Which of the local variables can be accessed within the `actionPerformed` method?
a) Only `button` can be accessed..
b) All of the local variables can be accessed.
c) `label` and `myOrder` can be accessed.
d) Only `myOrder` can be accessed.
Answer: c

66. Consider the following code snippet which is supposed to show the total order amount when the button is clicked:
```
public static void main(String[] args)
{
   final Order myOrder = new Order();
   JButton button = new JButton("Calculate");
   final JLabel label = new JLabel("Total amount due");
   . . .
   class MyListener implements ActionListener
   {
      public void actionPerformed(ActionEvent event)
      {
         label.setText("Total amount due " + myOrder.getAmountDue());
      }
   }
   ActionListener listener = new MyListener();
}
```
What is wrong with this code?

a) `button` should be declared as final
b) The listener has not been attached to the button.
c) The listener cannot access the methods of the `myOrder` object.
d) This code will display the total order amount.
Answer: b

67. Use the ____ method to specify the height and width of a graphical component when you add it to a `panel`.

a) `setPreferredDimensions`.
b) `setInitialDimensions`.
c) `setPreferredSize`.
d) `setInitialSize`.
Answer: c

68) Assuming that the `ClickListener` class implements the `ActionListener` interface, what statement should be used to complete the following code segment?

```
ClickListener listener = new ClickListener();
JButton myButton = new JButton("Submit");
JPanel myPanel = new JPanel();
myPanel.add(myButton);
______________________                        // missing statement
```

a) `myPanel.addActionListener(listener);`
b) `myButton.addActionListener(listener);`
c) `myPanel.addActionListener(myButton);`
d) `myButton.addActionListener(ClickListener);`
Answer: b

69) Assume that the `TimerListener` class implements the `ActionListener` interface. If the `actionPerformed` method in `TimerListener` needs to be executed every second, what statement should be used to complete the following code segment?

```
ActionListener listener = new TimerListener();
________________________________                    // missing statement
timer.start();
```

a) `Timer timer = new Timer(listener);`
b) `Timer timer = new Timer(1, listener);`
c) `Timer timer = new Timer(100, listener);`
d) `Timer timer = new Timer(1000, listener);`
Answer: d

70. The `Timer` class is found in the ____ package.

a) `java.awt`.
b) `javax.awt`.
c) `java.swing`.
d) `javax.swing`.
Answer: d

71. When you use a timer, you need to define a class that implements the ____ interface.
a) `TimerListener`
b) `TimerActionListener`
c) `ActionListener`
d) `StartTimerListener`
Answer: c

72. The ____ class in the `javax.swing` package generates a sequence of events, spaced apart at even time intervals.
a) `TimeClock`
b) `Timer`
c) `Clock`
d) `StopWatch`
Answer: b

73. Consider the following code snippet:
```
final RectangleComponent component = new RectangleComponent();

MouseListener listener = new MousePressListener();
component.addMouseListener(listener);

______________
frame.add(component);
frame.setSize(FRAME_WIDTH, FRAME_HEIGHT);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setVisible(true);
```

Which of the following statements completes this code?
a) `private static final int FRAME_WIDTH = 300;`
b) `private static final int FRAME_HEIGHT = 400;`
c) `Frame frame = new Frame();`
d) `JFrame frame = new JFrame();`
Answer: d

74. Consider the code snippet below:
```
public class RectangleComponent extends JComponent
{
   private Rectangle box;
   private static final int BOX_X = 100;
   private static final int BOX_Y = 100;
   private static final int BOX_WIDTH = 20;
   private static final int BOX_HEIGHT = 30;

   public RectangleComponent()
   {
      // The rectangle that the paint method draws
      box = new Rectangle(BOX_X, BOX_Y,
      BOX_WIDTH, BOX_HEIGHT);
   }
   public void paintComponent(Graphics g)
   {
      super.paintComponent(g);
      Graphics2D g2 = (Graphics2D) g;

      g2.draw(box);
   }
   public void moveTo(int x, int y)
   {
      box.setLocation(x, y);
      repaint();
   }
}
```
Which statement causes the rectangle to appear at an updated location?
a) `repaint();`
b) `g2.draw(box);`
c) `box.setLocation(x, y);`
d) `private Rectangle box;`
Answer: a

75. Consider the following code snippet:
```
class MyListener implements ActionListener
{
   public void actionPerformed(ActionEvent event)
   {
      System.out.println(event);
   }
}
Timer t = new Timer(interval, listener);
t.start();
```
What is wrong with this code?
a) The `Timer` object should be declared before the `MyListener` class.
b) The listener has not been attached to the `Timer` object.
c) The `Timer` object must be declared as `final`.
d) There is nothing wrong with the code.
Answer: b
76. You have a class which extends the `JComponent` class. In this class you have created a painted graphical object. Your code will change the data in the graphical object. What additional code is needed to ensure that the graphical object will be updated with the changed data?
a) You must call the component object's `paintComponent()` method.
b) You must call the component object's `repaint()` method.
c) You do not have to do anything – the component object will be automatically repainted when its data changes.
d) You must use a `Timer` object to cause the component object to be updated.
Answer: b

77. The ____ method should be called whenever you modify the shapes that the `paintComponent` method draws.
a) `draw`
b) `paint`
c) `paintComponent`
d) `repaint`
Answer: d

78. If the user presses and releases a mouse button in quick succession, which methods of the `MouseListener` interface are called?
a) `mousePressed, mouseReleased, and mouseClicked.`
b) `mousePressed, mouseReleased, and mouseDblClicked`
c) Only the `mouseClicked` method.
d) Only the `mouseDblClicked` method.
Answer: b

79. Suppose `listener` is an instance of a class that implements the `MouseListener` interface. How many methods does `listener` have?
a) 0 b) 1
c) 3 d) at least 5
Answer: d

80. Use the ____ method to add a mouse listener to a component.
a) `addListener`
b) `addMouseListener`
c) `addActionListener`
d) `addMouseActionListener`
Answer: b

81. A/an ____ is used to capture mouse events.
a) action listener
b) event listener
c) mouse listener
d) component listener
Answer: c

82. You wish to detect when the mouse is moved into a graphical component. Which methods of the `MouseListener` interface will provide this information?
a) `mouseMoved`
b) `mouseEntered`
c) `mouseOver`
d) `mouseIncoming`
Answer: b

83. Consider the following code snippet:

```
class MouseClickedListener implements ActionListener
{
   public void mouseClicked(MouseEvent event)
   {
      int x = event.getX();
      int y = event.getY();
      component.moveTo(x,y);
   }
}
```

What is wrong with this code?
a) The `mouseClicked` method cannot access the x and y coordinates of the mouse.
b) `repaint()` method was not called.
c) The class has implemented the wrong interface.
d) There is nothing wrong with this code.
Answer: c
84. Consider the following code snippet:

```
public class MyMouseListener
{
   public void mousePressed(MouseEvent event)
   {
      double x;
      double y;
      _______
      System.out.println("x: " + x + ", y: " + y);
   }
}
```

Which of the following statements should be in the indicated position to print out where the mouse was pressed?

a) `x = event.getXposition(); y = event.getYposition();`
b) `x = (MouseEvent) getX(); y = (MouseEvent) getY();`
c) `x = event.printX(); y = event.printY();`
d) `x = event.getX(); y = event.getY();`
Answer: d

85. What does the `MouseAdapter` class provide?

a) `MouseAdapter` class implements all of the methods of the `MouseListener` interface as do nothing methods, eliminating the need to implement the `MouseListener` interface.
b) `MouseAdapter` class allows your program to accept input from multiple mice.
c) `MouseAdapter` class implements all of the methods of the `ActionListener` interface as do nothing methods, eliminating the need to implement the `ActionListener` interface.
d) A class can implement the `MouseAdapter` class to handle mouse events.
Answer: a

Chapter 10: Inheritance

1. A class that represents the most general entity in an inheritance hierarchy is called a/an ____.
a) Default class. b) Superclass.
c) Subclass. d) Inheritance class.
Answer: b

2. A class that represents a more specific entity in an inheritance hierarchy is called a/an ____.
a) Default class b) Superclass
c) Subclass. d) Inheritance class.
Answer: c

3. You are creating a class inheritance hierarchy about motor vehicles that will contain classes named `Vehicle`, `Auto`, and `Motorcycle`. Which of the following statements is correct?
a) `Vehicle` should be the default class, while `Auto` and `Motorcycle` should be the subclasses.
b) `Vehicle` should be the superclass, while `Auto` and `Motorcycle` should be the subclasses.
c) `Vehicle` should be the subclass, while `Auto` and `Motorcycle` should be the superclasses.
d) `Vehicle` should be the subclass, while `Auto` and `Motorcycle` should be the default classes.
Answer: b

4. Which of the following statements about inheritance is correct?
a) You can always use a superclass object in place of a subclass object.
b) You can always use a subclass object in place of a superclass object.
c) A superclass inherits data and behavior from a subclass.
d) A superclass inherits only behavior from a subclass.
Answer: b

5. Insert the missing code in the following code fragment. This fragment is intended to call the `Vehicle` class's method.

```
public class Vehicle
{
   . . .
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
public class Motorcycle extends Vehicle
{
   . . .
   public Motorcycle()
   {
      _______________;
   }
}
```

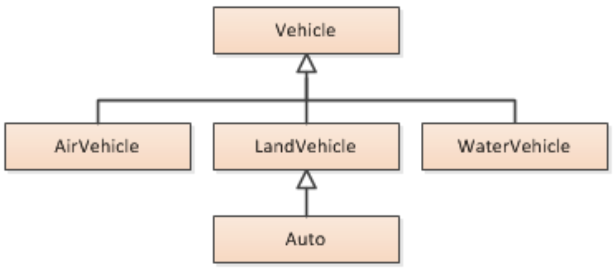
a) `Motorcyle.setVehicleClass(2.0);`
b) `Vehicle.setVehicleClass(2.0);`
c) `this;setVehicleClass(2.0);`
d) `setVehicleClass(2.0);`
Answer: d

6. Insert the missing code in the following code fragment. This fragment is intended to call the `Vessel` class's method.

```
public class Vessel
{
   . . .
   public void set VesselClass(double vesselLength)
   {
      . . .
   }
}
public class SpeedBoat extends Vessel
{
   . . .
   public SpeedBoat()
   {
      _______________;
   }}
```

a) `SpeedBoat.vesselLength(26.0);`
b) `Vessel.vesselLength(26.0);`
c) `this;vesselLength(26.0);`
d) `vesselLength(26.0);`
Answer: d

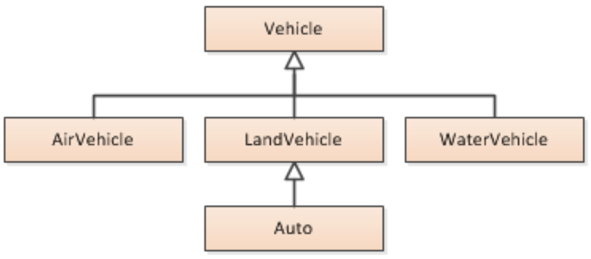
7. Consider the following inheritance hierarchy diagram:



Which of the following statements is correct?

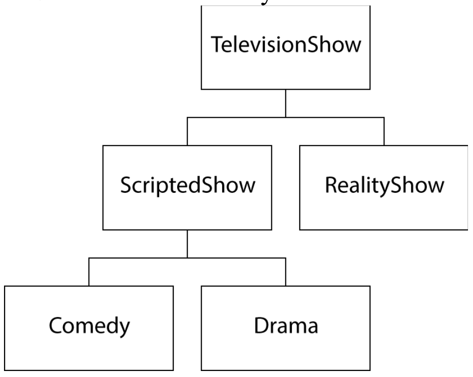
a) `Auto` is a superclass of `LandVehicle`, and `LandVehicle` is a superclass of Vehicle.
b) `Auto` is a superclass of `LandVehicle`, and `LandVehicle` is a subclass of Vehicle.
c) `Auto` is a subclass of `LandVehicle`, and `LandVehicle` is a superclass of Vehicle.
d) `Auto` is a subclass of `LandVehicle`, and `LandVehicle` is a subclass of Vehicle.
Answer: d

8. Consider the following inheritance hierarchy diagram:



Which of the following statements is correct?
a) `Auto` class inherits from `LandVehicle` class, and `LandVehicle` class inherits from `Vehicle` class.
b) `Auto` class inherits from `LandVehicle` class, and `Vehicle` class inherits from `LandVehicle` class.
c) `LandVehicle` class inherits from `Auto` class, and `LandVehicle` class inherits from `Vehicle` class.
d) `LandVehicle` class inherits from `Auto` class, and `Vehicle` class inherits from `LandVehicle` class.
Answer: a

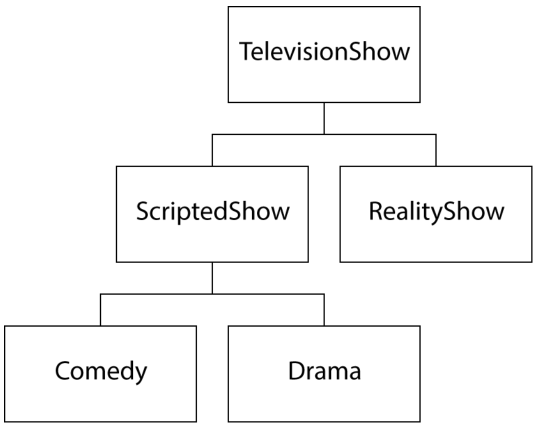
9. Consider the hierarchy of classes shown below.



Which represent valid class headers that would be found in this hierarchy?

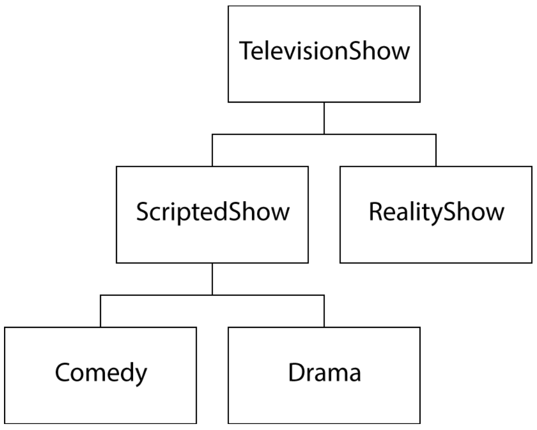
```
a) public class ScriptedShow extends TelevisionShow {. . .
   public class Comedy extends ScriptedShow {. . .
b) public class TelevisionShow extends ScriptedShow {. . .
   public class ScriptedShow extends Comedy {. . .
c) public class Drama extends TelevisionShow {. . .
   public class Comedy extends Drama {. . .
d) public class ScriptedShow extends RealityShow {. . .
   public class RealityShow extends ScriptedShow {. . .
```
Answer: a

10. Consider the hierarchy of classes shown below.



What is the superclass of the class `ScriptedShow`?
a) `Comedy`
b) `RealityShow`
c) `Object`
d) `TelevisionShow`
Answer: d

11. Consider the hierarchy of classes shown below.



What is the superclass of the class `TelevisionShow`?

a) `Object`
b) `Comedy`
c) `RealityShow`
d) This class has no superclass
Answer: a

12. All hamsters are rodents and all rodents are mammals. What hierarchy best captures this information?

a) Hamster is a superclass of Rodent and Rodent is a superclass of Mammal
b) Mammal is a superclass of Rodent and Rodent is a superclass of Hamster
c) Mammal is a superclass of Rodent and Hamster
d) Hamster is a superclass of Rodent and Mammal
Answer: b

13. All rodents are mammals and all canines are mammals. No canines are rodents and no rodents are canines. What hierarchy best captures this information?

a) Mammal is a superclass of Rodent and Mammal
b) Rodent is a superclass of Mammal and Canine is a superclass of Mammal
c) Mammal is a superclass of Rodent and Rodent is a superclass of Canine
d) Mammal is a superclass of Canine and Canine is a superclass of Rodent
Answer: a

14. Consider the classes shown below:

```
public class Parent
{
   public void doSomething(){/* Implementation not shown */}
}
public class Child extends Parent
{
   public void doAnotherThing(){/* Implementation not shown */}
}
```

Which lines in the following code will compile without error?

```
   Child kid = new Child();
   kid.doSomething(); // line 1
   kid.doAnotherThing(); // line 2
```

a) Line 1 only
b) Line 2 only
c) Lines 1 and 2
d) Neither line will compile without error
Answer: c

15. Consider the classes shown below:

```
public class Parent
{
   public void doSomething(){/* Implementation not shown */}
}

public class Child extends Parent
{
   public void doAnotherThing(){/* Implementation not shown */}
}
```

Which lines in the following code will compile without error?

```
   Parent kid = new Child();
   kid.doSomething(); // line 1
   kid.doAnotherThing(); // line 2
```

a) Line 1 only   b) Line 2 only
c) Lines 1 and 2   d) Neither line will compile without error
Answer: a

16. Consider the classes shown below:

```
public class Parent
{
   private int value = 100;
   public int getValue()
   {
      return value;
   }
}
public class Child extends Parent
{
private int value;
   public Child(int number)
   {
      value = number;
   }
}
```

What is the output of the following lines of code?

```
Child kid = new Child(-14);
   Parent adult = new Parent();
   System.out.println(kid.getValue() + " "
+adult.getValue());
```

a) `100 100`   b) `-14 100`
c) `-14 -14`   d) `100 -14`
Answer: a

17. Consider the classes shown below:

```
public class Parent
{
   private int value = 100;
   public int getValue()
   {
      return value;
   }
}
public class Child extends Parent
{
private int value;
   public Child(int number)
   {
      value = number;
   }
}
```

What is the output of the following lines of code?

```
Child kid = new Child(-14);
   Child kid2 = new Child(21);
   System.out.println(kid.getValue() + " "
         + kid2.getValue());
```

a) `-14 21`   b) `21 21`
c) `21 100`   d) `100 100`
Answer: d

18. Consider the classes shown below:
public class Parent

```
{
   private int value = 100;
   public int getValue()
   {
      return value;
   }
}
public class Child extends Parent
{
   private int value;
   public Child(int number)
   {
      value = number;
   }
}
```

What is the output of the following lines of code?

```
   Child kid = new Child(-14);
   Parent kid2 = new Child(21);
   System.out.println(kid.getValue() + " "
         + kid2.getValue());
```

a) `100 100`   b) `-14 21`
c) `21 21`   d) `-14 100`
Answer: a

19. Suppose the class `Value` is partially defined below

```
public class Value
{
   private int number;

   public int getValue()
   {
      return number;
   }
}
```

A subclass of `Value`, `LargerValue`, is defined with a `getValue` method that returns twice the value of the parent. Which line is the body of `LargerValue`'s `getValue` method?
a) `return getValue() * 2;`
b) `return super.getValue() * 2;`
c) `return number * 2;`
d) `return super.number * 2;`
Answer: b

20. Suppose the class `Message` is partially defined as shown below

```
public class Message
{
   private String value;

   public Message(String initial)
   {
      value = initial;
   }

   public String getMessage()
   {
      return value;
   }
}
```

A subclass of Message, `ExcitedMessage`, is defined that will behave like `Message`, except that it will add two exclamation points to the end of the message. Sample code that uses `ExcitedMessage` is shown below.

```
   ExcitedMessage greeting = new ExcitedMessage("Hello");
    System.out.print(greeting.getMessage());
                                // will print "Hello!!"
```

Which `ExcitedMessage` constructor will give this behavior?

```
a) public ExcitedMessage(String line)
   {
      super(line + "!!");
   }
b) public ExcitedMessage(String line)
   {
      value = line + "!!";
   }
c) public ExcitedMessage(String line)
   {
      line = line + "!!";
      super(line);
   }
d) public ExcitedMessage(String line)
   {
      new Message(line + "!!");
   }
```

Answer: a

21. To create a subclass, use the ____ keyword.

a) `inherits`
b) `implements`
c) `interface`
d) `extends`
Answer: d

22. You are creating a `Motorcycle` class which is supposed to inherit from the `Vehicle` class. Which of the following class declaration statements will accomplish this?

a) `public class Motorcycle inherits Vehicle`
b) `public class Motorcycle implements Vehicle`
c) `public class Motorcycle interfaces Vehicle`
d) `public class Motorcycle extends Vehicle`
Answer: d

23. You are creating a `Motorcycle` class which is supposed to be a subclass of the `Vehicle` class. Which of the following class declaration statements will accomplish this?

a) `public class Motorcycle extends Vehicle`
b) `public class Motorcycle implements Vehicle`
c) `public class Motorcycle interfaces Vehicle`
d) `public class Motorcycle inherits Vehicle`
Answer: a

24. You are creating a `Motorcycle` class that is supposed to be a subclass of the `Vehicle` class. Which of the following class declaration statements will accomplish this?

a) `public class Vehicle extends Motorcycle`
b) `public class Motorcycle extends Vehicle`
c) `public class Vehicle inherits Motorcycle`
d) `public class Motorcycle inherits Vehicle`
Answer: b

25. Which of the following is true regarding subclasses?

a) A subclass inherits methods from its superclass but not instance variables.
b) A subclass inherits instance variables from its superclass but not methods.
c) A subclass inherits methods and instance variables from its superclass.
d) A subclass does not inherit methods or instance variables from its superclass.
Answer: c

26. Which of the following is true regarding subclasses?

a) A subclass that inherits methods from its superclass may not override the methods.
b) A subclass that inherits instance variables from its superclass may not declare additional instance variables.
c) A subclass may inherit methods or instance variables from its superclass but not both.
d) A subclass may inherit methods and instance variables from its superclass, and may also implement its own methods and declare its own instance variables.
Answer: d

27. Which of the following is true regarding subclasses?

a) A subclass has access to private instance variables of its superclass.
b) A subclass does not have access to public instance variables of its superclass.
c) A subclass must specify the implicit parameter to use methods inherited from its superclass.
d) A subclass has no access to private instance variables of its superclass.
Answer: d

28. Which of the following indicates that a class named `Class1` is a subclass of a class named `Class2`?

a) `public class Class1 extends Class2`
b) `public class Class1 implements Class2`
c) `public class Class2 extends Class1`
d) `public class Class2 implements Class1`
Answer: a

29. Which of the following indicates that a class named `ClassA` class is a superclass of the `ClassB` class?

a) `public class ClassB extends ClassA`
b) `public class ClassB implements ClassA`
c) `public class ClassA extends ClassB`
d) `public class ClassA implements ClassB`
Answer: a

30. What must a subclass do to modify a private superclass instance variable?

a) The subclass must simply use the name of the superclass instance variable.
b) The subclass must declare its own instance variable with the same name as the superclass instance variable.
c) The subclass must use a public method of the superclass (if it exists) to update the superclass's private instance variable.
d) The subclass must have its own public method to update the superclass's private instance variable.
Answer: c

31. Which of the following indicates that the `Motorcycle` class is a subclass of the `Vehicle` class?

a) `public class Motorcycle extends Vehicle`
b) `public class Motorcycle implements Vehicle`
c) `public class Vehicle extends Motorcycle`
d) `public class Vehicle implements Motorcycle`
Answer: a

32. Consider the following code snippet:

```
public class Vehicle
{
   private String manufacturer;
   . . .
   public void setVehicleClass(double numberAxles)
   {
       . . .
   }
}
```

If a `Motorcycle` class is created as a subclass of the `Vehicle` class, which of the following statements is correct?

a) A `Motorcycle` object inherits and can directly use both the instance variable `manufacturer` and the method `setVehicleClass`.
b) A `Motorcycle` object inherits and can directly use the instance variable `manufacturer` but not the method `setVehicleClass`.
c) A `Motorcycle` object inherits but cannot directly use either the instance variable `manufacturer` or the method `setVehicleClass`.
d) A `Motorcycle` object inherits and can directly use the method `setVehicleClass` but cannot directly use the instance variable `manufacturer`.
Answer: d

33. Consider the following code snippet:

```
public class Vessel
{
   private String manufacturer;
   . . .
   public void setVesselClass(double engineRPM)
   {
      . . .
   }
}
```

If a `Speedboat` class is created as a subclass of the `Vessel` class, which of the following statements is correct?

a) A `Speedboat` object inherits and can directly use both the instance variable `manufacturer` and the method `setVesselClass`.
b) A `Speedboat` object inherits and can directly use the instance variable `manufacturer` but not the method `setVesselClass`.
c) A `Speedboat` object inherits but cannot directly use either the instance variable `manufacturer` or the method `setVesselClass`.
d) A `Speedboat` object inherits and can directly use the method `setVesselClass` but cannot directly use the instance variable `manufacturer`.
Answer: d

34. Consider the following class hierarchy:

```
public class Vehicle
{
   private String type;
   public Vehicle(String type)
   {
      this.type = type;
   }
   public String displayInfo()
   {
      return type;
   }
}
public class LandVehicle extends Vehicle
{
   public LandVehicle(String type)
   {
      super(type);
   }
}
public class Auto extends LandVehicle
{
   public Auto(String type)
   {
      super(type);
   }
}
```

You have written a program to use these classes, as shown in the following code snippet:

```
public class VehicleTester
{
   public static void main(String[] args)
   {
      Auto myAuto = new Auto("sedan");
      System.out.println("MyAuto type = " + ______);
   }
}
```

Complete the code in this program snippet to correctly display the auto's type.

a) `myAuto.displayInfo()`
b) `myAuto.super.displayInfo()`
c) `myAuto.super.super.displayInfo()`
d) This cannot be done unless the `Auto` class overrides the `displayInfo` method.
Answer: a

35. Consider the following class hierarchy:

```
public class Vehicle
{
   private String type;
    public Vehicle(String type)
   {
      this.type = type;
   }
   public String displayInfo()
   {
      return type;
   }
}

public class LandVehicle extends Vehicle
{
   public LandVehicle(String type)
   {
      super(type);
   }
}

public class Auto extends LandVehicle
{
   public Auto(String type)
   {
      _________;
   }
}
```

Complete the code in the `Auto` class constructor to store the `type` data.
a) super(type);
b) super(super(type));
c) super.super(type);
d) This cannot be done unless the Auto declares an instance variable named type.
Answer: a

36. Consider the following class hierarchy:

```
public class Vehicle
{
   private String type;
   public Vehicle(String type)
   {
      this.type = type;
   }
   public String displayInfo()
   {
      return type;
   }
}
```

```
public class LandVehicle extends Vehicle
{
   public LandVehicle(String type)
   {
      . . .
   }
}

public class Auto extends LandVehicle
{
   public Auto(String type)
   {
      . . .
   }
   public String displayAutoType()
   {
      return _____;
   }
}
```
Complete the code in the `Auto` class method named `displayAutoType` to return the `type` data.
a) `super(type);`
b) `super.type;`
c) `super.super.type;`
d) `super.displayInfo()`
Answer: d

37. Which of the following statements about superclasses and subclasses is true?

a) A superclass is larger than its subclass.
b) A superclass inherits from a subclass.
c) A superclass extends a subclass.
d) A subclass extends a superclass.
Answer: d

38. Consider the following code snippet:

```
public class Vehicle
{
   . . .
   public void setVehicleAtrributes()
   {
      . . .
   }
}
public class Auto extends Vehicle
{
   . . .
   public void setVehicleAtrributes()
   {
   . . .
   }
}
```
Which of the following statements is correct?

a) The subclass is shadowing a superclass method.
b) The subclass is overloading a superclass method.
c) The subclass is overriding a superclass method.
d) This code will not compile.
Answer: c

39. Consider the following code snippet:

```
public class Vessel
{
   . . .
   public void setVesselAtrributes()
   {
      . . .
   }
}
```

```
public class Speedboat extends Vessel
{
   . . .
   public void setVesselAtrributes()
   {
   . . .
   }
}
```
Which of the following statements is correct?

a) The subclass is shadowing a superclass method.
b) The subclass is overloading a superclass method.
c) The subclass is overriding a superclass method.
d) This code will not compile.
Answer: c

40. Consider the following code snippet:

```
public void deposit(double amount)
{
   transactionCount ++;
   super.deposit(amount);
}
```
Which of the following statements is true?
a) This method will call itself.
b) This method calls a public method in its subclass.
c) This method calls a private method in its superclass
d) This method calls a public method in its superclass.
Answer: d

41. Which reserved word must be used to call a method of a superclass?

a) `this` b) `my`
c) `parent` d) `super`
Answer: d

42. If a subclass defines the same method name and the same parameter types for a method that appears in its superclass, ____.
a) the subclass method overloads the superclass method.
b) the subclass method overrides the superclass method.
c) the subclass has implemented its superclass's method.
d) a compiler error will occur.
Answer: b

43. Consider the following code snippet:

```
public class Vehicle
{
   . . .
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
public class Motorcycle extends Vehicle
{
   . . .
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
```
Which of the following statements is correct? (I changed the wording below, because methods override methods. . .classes don't override methods)

a) The `Motorcycle` class's `setVehicleClass` method overrides the `Vehicle` class's `setVehicleClass` method.
b) The `Vehicle` class's `setVehicleClass` method overrides the `Motorcycle` class's `setVehicleClass` method.
c) The `Motorcycle` class's `setVehicleClass` method overloads the `Vehicle` class's `setVehicleClass` method.
d) The `Vehicle` class's `setVehicleClass` method overloads the `Motorcycle` class's `setVehicleClass` method.
Answer: a

44. Consider the following code snippet:

```
public class Vessel
{
   . . .
   public void setVesselClass(double numberAxles)
   {
      . . .
   }
}
public class Speedboat extends Vessel
{
   . . .
   public void setVesselClass(double numberAxles)
   {
      . . .
   }
}
```

Which of the following statements is correct?
a) The `Speedboat` class's `setVesselClass` method overrides the `Vessel` class's `setVesselClass` method.
b) The `Vessel` class's `setVesselClass` method overrides the `Speedboat` class's `setVesselClass` method.
c) The `Speedboat` class's `setVesselClass` method overloads the `Vessel` class's `setVesselClass` method.
d) The `Vessel` class's `setVesselClass` method overloads the `Speedboat` class's `setVesselClass` method.
Answer: a

45. Consider the following code snippet:

```
public class Vehicle
{
   . . .
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
public class Motorcycle extends Vehicle
{
   . . .
   public void setModelName(String model)
   {
      . . .
   }
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
```

Which of the following statements is NOT correct?
a) The `Motorcycle` class can call the `setVehicleClass` method of the `Vehicle` class.
b) The `Vehicle` class can call the `setModelName` method.
c) The `Motorcycle` class's `SetVehicleClass` method overrides the `Vehicle` class's `setVehicleClass` method.
d) The `Motorcycle` class can call the `setVehicleClass` method of the `Motorcycle` class.
Answer: b

46. Consider the following code snippet:

```
public class Employee
{
   . . .
   public void setDepartment(String deptName)
   {
      . . .
  }
}
public class Programmer extends Employee
{
   . . .
   public void setProjectName(String projName)
   {
      . . .
   }
   public void setDepartment(String deptName)
   {
      . . .
   }
}
```

Which of the following statements is NOT correct?
a) The `Programmer` class can call the `setDepartment` method of the `Employee` class.
b) The `Employee` class can call the `setProjectName` method. c) The `Programmer` class's `setDepartment` method overrides the `Employee` class's `setDepartment` method.
d) The `Programmer` class can call the `setDepartment` method of the `Programmer` class.
Answer: b

47. To override a superclass method in a subclass, the subclass method ____.
a) Must use a different method name.
b) Must use the same method name and the same parameter types.
c) Must use a different method name and the same parameter types.
d) Must use a different method name and different parameter types.
Answer: b

48. Consider the following code snippet:

```
public class Vehicle
{
   . . .
   public void setVehicleClass(double numberAxles)
   {
      . . .
   }
}
public class Auto extends Vehicle
{
   . . .
   public void setVehicleClass(int numberAxles)
   {
      . . .
   }
}
```

Which of the following statements is correct?
a) The `Auto` class overrides the `setVehicleClass` method.
b) The `Vehicle` class overrides the `setVehicleClass` method.
c) The `Auto` class overloads the `setVehicleClass` method.
d) The `Vehicle` class overloads the `setVehicleClass` method.
Answer: c

49. Consider the following code snippet:

```
public class Employee
{
   . . .
   public void setEmployeeDept(String deptNum)
   {
      . . .
   }
}
public class Programmer extends Employee
{
   . . .
   public void setEmployeeDept(int deptNum)
   {
      . . .
   }
}
```

Which of the following statements is correct?

a) The `Programmer` class overrides the `setEmployeeDept` method.
b) The `Employee` class overrides the `setEmployeeDept` method.
c) The `Programmer` class overloads the `setEmployeeDept` method.
d) The `Employee` class overloads the `setEmployeeDept` method.
Answer: c

50. If a subclass contains a method with the same name as a method in its superclass, but with different parameter types, the subclass method is said to ____ the method of the superclass.

a) implement
b) inherit
c) override
d) overload
Answer: d

51. If a subclass uses the same method name but different parameter types for a method that appears in its superclass, ____.

a) the subclass method has overloaded its superclass's method.
b) the subclass method has overridden its superclass's method.
c) the subclass has implemented its superclass's method.
d) a compiler error will occur.
Answer: a

52. Consider the following code snippet that appears in a subclass:

```
public void deposit(double amount)
{
   transactionCount ++;
   deposit(amount);
}
```

Which of the following statements is true?
a) This method will call itself.
b) This method calls a public method in its subclass.
c) This method calls a private method in its superclass
d) This method calls a public method in its superclass.
Answer: a

53. Consider the following code snippet:

```
public class Auto extends Vehicle
{
   . . .
   public Auto(int numberAxles)
   {
      super(numberAxles);
   }
}
```
What does this code do?

a) It invokes the constructor of the `Vehicle` class from within the constructor of the `Auto` class.
b) It invokes the constructor of the `Auto` class from within the constructor of the `Vehicle` class.
c) It invokes a private method of the `Vehicle` class from within a method of the `Auto` class.
d) This code will not compile.
Answer: a

54. Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   . . .
   public Motorcycle(int numberAxles)
   {
      super(numberAxles);
   }
}
```
What does this code do?

a) It invokes the constructor of the `Vehicle` class from within the constructor of the `Motorcycle` class.
b) It invokes the constructor of the `Motorcycle` class from within the constructor of the `Vehicle` class.
c) It invokes a private method of the `Vehicle` class from within a method of the `Motorcycle` class.
d) This code will not compile.
Answer: a

55. Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   . . .
   public Motorcycle(int numberAxles)
   {
      super(numberAxles);  //line #1
   }
}
```

If the line marked "`//line #1`" was missing, which of these statements would be correct?

a) The `Motorcycle` class constructor would invoke the constructor of the `Vehicle` class with no parameters.
b) The `Vehicle` class constructor would invoke the constructor of the `Motorcycle` class with no parameters.
c) The `Motorcycle` class constructor would invoke the constructor of the `Vehicle` class with a parameter value of 0.
d) This code would not compile.
Answer: a

56. Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   . . .
   public Motorcycle(int numberAxles)
   {
      super.numberAxles;
   }
}
```

What does this code do?
a) It invokes the constructor of the `Vehicle` class from within the constructor of the `Motorcycle` class.
b) It invokes the constructor of the `Motorcycle` class from within the constructor of the `Vehicle` class.
c) It invokes a private method of the `Vehicle` class from within a method of the `Motorcycle` class.
d) This code will not compile.
Answer: d

57. Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   private String model;
   . . .
   public Motorcycle(int numberAxles, String modelName)
   {
      model = modelName;
      super(numberAxles);
   }
}
```

What does this code do?
a) It invokes the constructor of the `Vehicle` class from within the constructor of the `Motorcycle` class.
b) It invokes the constructor of the `Motorcycle` class from within the constructor of the `Vehicle` class.
c) It invokes a private method of the `Vehicle` class from within a method of the `Motorcycle` class.
d) This code will not compile.
Answer: d

58. Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   private String model;
   . . .
   public Motorcycle(int numberAxles, String modelName)
   {
      super(numberAxles);
      model = modelName;
   }
}
```

What does this code do?
a) It invokes the constructor of the `Vehicle` class from within the constructor of the `Motorcycle` class.
b) It invokes the constructor of the `Motorcycle` class from within the constructor of the `Vehicle` class.
c) It invokes a private method of the `Vehicle` class from within a method of the `Motorcycle` class.
d) This code will not compile.
Answer: a

59. Consider the classes shown below:

public class Parent

```
{
   public void doSomething()       //  method 1
   { /* Implementation not shown */ }
}
public class Child extends Parent
{
   public void doSomething(int n)    //  method 2
   { /* Implementation not shown */ }
   public void doSomething()         //  method 3
   { /* Implementation not shown */ }
}
```

If the variable kid is defined below, which version of the `doSomething` method can be called on the variable kid?

```
Child kid = new Child();
```

a) Methods 1 and 2 only
b) Method 2 only
c) Methods 2 and 3 only
d) Methods 1, 2, and 3
Answer: b

60. Consider the classes shown below:

```
public class Parent
{
   public void doSomething()       //  method 1
   { /* Implementation not shown */ }
}

public class Child extends Parent {
   public void doSomething(int n)    //  method 2
   { /* Implementation not shown */ }

   public void doSomething()         //  method 3
   { /* Implementation not shown */ }
}
```

If the variable `kid` is defined below, which version of the `doSomething` method can be called on the variable `kid`?

```
Parent kid = new Child();
```

a) Method 1 only
b) Methods 2 and 3 only
c) Methods 1 and 2 only
d) Methods 1, 2, and 3
Answer: a

61. Consider the classes shown below:

```
public class Parent
{
   public int getValue()
   {
      return 24;
   }
   public void display()
   {
      System.out.print(getValue() + " ");
   }
}
public class Child extends Parent
{
   public int getValue()
   {
      return -7;
   }
}
```

Using the classes above, what is the output of the following lines of code?

```
Child kid = new Child();
```

```
   Parent adult = new Parent();
   kid.display();
adult.display();
```

a) `24 24`
b) `-7 -7`
c) `-7 24`
d) `24 -7`
Answer: c

62. Consider the classes shown below:

```
public class Parent
{
   public int getValue()
   {
      return 24;
   }
   public void display()
   {
      System.out.print(getValue() + " ");
   }
}
public class Child extends Parent
{
   public int getValue()
   {
      return -7;
   }
}
```

Using the classes above, what is the output of the following lines of code?

```
   Parent kid = new Child();
   Parent adult = new Parent();
   kid.display();
   adult.display();
```

a) `-7 24`
b) `24 24`
c) `-7 -7`
d) `24 -7`
Answer: a

63. Suppose the abstract class `Message` is defined below

```
public abstract class Message {
   private String value;

   public Message(String initial)
   {
      value = initial;
   }
   public String getMessage()
   {
      return value;
   }

   public abstract String translate();
}
```

A concrete subclass of `Message`, `FrenchMessage`, is defined. Which methods must `FrenchMessage` define?

a) `translate()` only
b) `getMessage()` only
c) The `FrenchMessage` constructor and `translate()` only
d) The `FrenchMessage` constructor, `getMessage()`, and `translate()`
Answer: a

64. Consider the `Counter` class below.

```
public class Counter
{
   public int count = 0;

   public int getCount()
   {
      return count;
   }

   public void increment()
   {
      count++;
   }
}
```

Using the class above and the variable declared below, what is the value of `num.toString()`?

```
Counter num = new Counter();
```

a) a string with `count`'s value
b) a string with `num`'s type and hashcode
c) a string with `count`'s type and hashcode
d) nothing since `toString` is not defined for `Counter`
Answer: b

65. Consider the `Counter` class below.

```
public class Counter
{
   public int count = 0;

   public int getCount()
   {
      return count;
   }

   public void increment()
   {
      count++;
   }
}
```

Using the class above and the variables declared below, what is the value of `num1.equals(num2)`?

```
Counter num1 = new Counter();
Counter num2 = new Counter();
```

a) true
b) false
c) nothing since `equals` is not defined for `Counter`
Answer: b

66. Consider the `Counter` class below.

```
public class Counter
{
   public int count = 0;

   public int getCount()
   {
      return count;
   }

   public void increment()
   {
      count++;
   }
}
```

Using the class above and the variables declared below, what is the value of `num1.equals(num2)`?

```
Counter num1 = new Counter();
Counter num2 = num1;
```

a) true
b) false
c) nothing since `equals` is not defined for `Counter`
Answer: a

67. Consider the following class hierarchy:
```
public class Vehicle
{
   private String type;
   public Vehicle(String type)
   {
      this.type = type;
   }
   public String getType()
   {
      return type;
   }
}
public class LandVehicle extends Vehicle
{
   public LandVehicle(String type)
   {
      . . .
   }
}
public class Auto extends LandVehicle
{
   public Auto(String type)
   {
      . . .
   }
}
```
Which of the following code fragments is NOT valid in Java?
a) `Vehicle myAuto = new Auto("sedan");`
b) `LandVehicle myAuto = new Auto("sedan");`
c) `Auto myAuto = new Auto("sedan");`
d) `LandVehicle myAuto = new Vehicle("sedan");`
Answer: d

68) Consider the following code snippet:
```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```
If the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type, which statement is correct?

a) The `moveForward` method of the `Auto` class will be executed.
b) The `moveForward` method of the `Vehicle` class will be executed.
c) You must specify in the code which class's `moveForward` method is to be used.
d) It is not possible to determine which class's method is called.
Answer: a

69) Consider the following code snippet:
```
Employee anEmployee = new Programmer();
anEmployee.increaseSalary(2500);
```

If the `Programmer` class inherits from the `Employee` class, and both classes have an implementation of the `increaseSalary` method with the same set of parameters and the same return type, which statement is correct?

a) The `increaseSalary` method of the `Programmer` class will be executed.
b) The `increaseSalary` method of the `Employee` class will be executed.
c) You must specify in the code which class's `increaseSalary` method is to be used.
d) It is not possible to determine which class's method is called.
Answer: a

70. Consider the following code snippet:

```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```
Assume that the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type. The process for determining which class's `moveForward` method to execute is called ____.
a) inheritance disambiguation. b) inheritance hierarchy.
c) dynamic inheritance. d) dynamic lookup.
Answer: d

71. Consider the following code snippet:
```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```

Assume that the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type. Which class's `moveForward` method is to be executed is determined by ____.

a) the actual object type.
b) the variable's type.
c) the hierarchy of the classes.
d) it is not possible to determine which method is executed.
Answer: a

72. Consider the following code snippet:

```
Employee anEmployee = new Programmer();
anEmployee.increaseSalary(2500);
```

Assume that the `Programmer` class inherits from the `Employee` class, and both classes have an implementation of the `increaseSalary` method with the same set of parameters and the same return type. Which class's `increaseSalary` method is to be executed is determined by ____.
a) the hierarchy of the classes.
b) the variable's type.
c) the actual object type.
d) it is not possible to determine which method is executed.
Answer: c

73. Which of the following statements about abstract methods is true
a) An abstract method has a name, parameters, and a return type, but no code in the body of the method.
b) An abstract method has parameters, a return type, and code in its body, but has no defined name.
c) An abstract method has a name, a return type, and code in its body, but has no parameters.
d) An abstract method has only a name and a return type, but no parameters or code in its body.
Answer: a

74. Which of the following statements about classes is true?

a) You can create an object from a concrete class, but not from an abstract class.
b) You can create an object from an abstract class, but not from a concrete class.
c) You cannot have an object reference whose type is an abstract class.
d) You cannot create subclasses from abstract classes.
Answer: a

75. If a class has an abstract method, which of the following statements is NOT true?

a) You can construct an object from this class.
b) You can have an object reference whose type is this class.
c) You can inherit from this class.
d) All non-abstract subclasses of this class must implement this method.
Answer: a

76. Consider the following code snippet:

```
public abstract class Machine
{
   public abstract void setRPMs();
   . . .
}
```

You wish to create a concrete subclass named `PolisherMachine`. Which of the following is the correct way to declare this subclass?
a)
```
public class PolisherMachine implements Machine
{
   public void setRPMs() { . . . }
}
```
b)
```
public class PolisherMachine extends Machine
{
   void setRPMs() { . . . }
}
```
c)
```
public class PolisherMachine implements Machine
{
   void setRPMs() { . . . }
}
```
d)
```
public class PolisherMachine extends Machine
{
   public void setRPMs() { . . . }
}
```
Answer: d

77. Consider the following code snippet:
```
public abstract class Employee
{
   public abstract void setSalary();
   . . .
}
```
You wish to create a concrete subclass named `Programmer`. Which of the following is the correct way to declare this subclass?
a)
```
public class Programmer implements Employee
{
   public void setSalary() { . . . }
}
```
b)
```
public class Programmer extends Employee
{
   void setSalary() { . . . }
}
```

c)
```
public class Programmer implements Employee
{
   void setSalary() { . . . }
}
```

d)
```
public class Programmer extends Employee
{
   public void setSalary() { . . . }
}
```
Answer: d

78. A class from which you cannot create objects is called a/an ____.
a) Abstract class.
b) Concrete class.
c) Non-inheritable class.
d) Superclass.
Answer: a

79. A class that cannot be instantiated is called a/an ____.
a) Abstract class.
b) Anonymous class.
c) Concrete class.
d) Non-inheritable class.
Answer: a

80. Which of the following statements about classes is true?
a) You can create an object from a class declared with the keyword `final`.
b) You can override methods in a class declared with the keyword `final`.
c) You can extend a class declared with the keyword `final`.
d) You can create subclasses from a class declared with the keyword `final`.
Answer: a

81. The ____ reserved word in a class definition ensures that subclasses cannot be created from this class.
a) `abstract`   b) `anonymous`
c) `final`   d) `static`
Answer: c

82. The ____ reserved word in a method definition ensures that subclasses cannot override this method.
a) `abstract` b) `anonymous`
c) `final` d) `static`
Ans: c

83. Which of the following is true regarding inheritance?
a) When creating a subclass, all methods of the superclass must be overridden.
b) When creating a subclass, no methods of a superclass can be overridden.
c) A superclass can force a programmer to override a method in any subclass created from it.
d) A superclass cannot prevent a programmer from overriding a method in any subclass created from it.
Answer: c
84. When declared as `protected`, data in an object can be accessed by ____.
a) Only by that class's methods and by all of its subclasses
b) Only by that class's methods, by all of its subclasses, and by methods in classes within the same package.
c) Only by that class's methods.
d) By any class.
Answer: b

85. Consider the following code snippet:
```
public class Vehicle
{
   protected int numberAxles;
   . . .
}
```
Data in the `numberAxles` variable can be accessed by ____.
a) Only by the `Vehicle` class's methods and by all of its subclasses
b) Only by the `Vehicle` class's methods, by all of its subclasses, and by methods in classes within the same package.
c) Only by the `Vehicle` class's methods.
d) By any class.
Answer: b

86. To ensure that an instance variable can only be accessed by the class that declared it, the variable should be declared as ____.
a) `public`
b) `private`
c) `protected`
d) `final`
Answer: b

87. With a few exceptions, instance variables of classes should always have ___ access.
a) `final`
b) `private`
c) `public`
d) `protected`
Answer: b

88. Consider the following code snippet:
```
Vehicle aVehicle = new Auto(4,"gasoline");
String s = aVehicle.toString();
```
Assume that the `Auto` class inherits from the `Vehicle` class, and neither class has an implementation of the `toString()` method. Which of the following statements is correct?
a) The `toString()` method of the `Object` class will be used when this code is executed. b) The `toString()` method of the `String` class will be used when this code is executed.
c) This code will not compile because there is no `toString()` method in the `Vehicle` class.
d) This code will not compile because there is no `toString()` method in the `Auto` class.
Answer: a

89. Consider the following code snippet:
```
Employee anEmployee = new Programmer();
String emp = anEmployee.toString();
```

Assume that the `Programmer` class inherits from the `Employee` class, and neither class has an implementation of the `toString()` method. Which of the following statements is correct?
a) The `toString()` method of the `Object` class will be used when this code is executed.
b) The `toString()` method of the `String` class will be used when this code is executed.
c) This code will not compile because there is no `toString()` method in the `Employee` class.
d) This code will not compile because there is no `toString()` method in the `Programmer` class.
Answer: a

90. Consider the following code snippet:
```
int numAxles = 4;
String s = "Number of axles is " + numAxles;
```

Which of the following statements is correct?
a) The `toString()` method of the `Object` class is being used to set the value of `s`.
b) The `toString()` method of the `Integer` class is being used to set the value of `s`.
c) No `toString()` method is being used to set the value of `s`.
d) This code will not compile.
Answer: c

91. Consider the following code snippet:
```
int vacationDays = 10;
String output = "Number of earned vacation days is " + vacationDays;
```

Which of the following statements is correct?
a) The `toString()` method of the `Object` class is being used to set the value of `output`.
b) The `toString()` method of the `Integer` class is being used to set the value of `output`.
c) No `toString()` method is being used to set the value of `output`.
d) This code will not compile.
Answer: c

92. Consider the following code snippet:
```
double salary = 45000.00;
String sal = "Current salary is " + salary;
```

Which of the following statements is correct?
a) The `toString()` method of the `Object` class is being used to set the value of `salary`.
b) The `toString()` method of the `Double` class is being used to set the value of `salary`.
c) No `toString()` method is being used to set the value of `salary`.
d) This code will not compile.
Answer: c

93. Consider the following code snippet:
```
Auto consumerAuto = new Auto(4, "gasoline");
String s = consumerAuto.toString();
```

Assume that the `Auto` class has not implemented its own `toString()` method. What value will `s` contain when this code is executed? a) `s` will contain the values of the instance variables in `consumerAuto`.
b) `s` will contain only the class name of the `consumerAuto` object.
c) `s` will contain the class name of the `consumerAuto` object followed by a hash code.
d) This code will not compile.
Answer: c

94. Consider the following code snippet:

```
Employee programmer = new Employee(10254, "exempt");
String s = programmer.toString();
```

Assume that the `Employee` class has not implemented its own `toString()` method. What value will `s` contain when this code is executed?
a) `s` will contain the values of the instance variables in `programmer`.
b) `s` will contain only the class name of the `programmer` object.
c) `s` will contain the class name of the `programmer` object followed by a hash code.
d) This code will not compile.
Answer: c

95. Which of the following statements about comparing objects is correct?
a) The `equals` method is used to compare whether two references are to the same object.
b) The `equals` method is used to compare whether two objects have the same contents.
c) The == operator is used to compare whether two objects have the same contents.
d) The `equals` method and the == operator perform the same actions.
Answer: b
96. Consider the following code snippet, which is meant to override the `equals()` method of the `Object` class:
```
public class Coin
{
   . . .
   public boolean equals(Coin otherCoin)
```

```
    {
       . . .
    }
    . . .
}
```
What is wrong with this code?
a) A class cannot override the `equals()` method of the `Object` class.
b) The `equals()` method must be declared as `private`.
c) A class cannot change the parameters of a superclass method when overriding it.
d) There is nothing wrong with this code.
Answer: c

97. Consider the following code snippet:

```
public class Coin
{
   private String name;
   . . .
   public boolean equals(Object otherCoin)
   {
      return name.equals(otherCoin.name);
   }
   . . .
}
```
What is wrong with this code?
a) The `return` statement should use the == operator instead of the `equals` method.
b) The parameter in the `equals` method should be declared as `Coin otherCoin`.
c) `otherCoin` must be cast as a `Coin` object before using the `equals` method.
d) There is nothing wrong with this code.
Answer: c

98. Consider the following code snippet:
```
public class Score
{
   private String name;
   . . .
   public boolean equals(Object otherScore)
   {
      return name.equals(otherScore.name);
   }
   . . .
}
```
What is wrong with this code?
a) The `return` statement should use the == operator instead of the `equals` method.
b) The parameter in the `equals` method should be declared as `Score otherScore`.
c) `otherScore` must be cast as a `Score` object before using the `equals` method.
d) There is nothing wrong with this code.
Answer: c

99. Consider the following code snippet of a function object

```
public interface Measurer
{
   double measure(______ anObject);
}
```

Complete this code to allow the interface to handle all classes?

a) `Class`	b) `Object`
c) `Any`	d) `Void`

Answer: b

100. Consider the following code snippet:

```
if(anObject instanceof Auto)
{
   Auto anAuto = (Auto) anObject;
   . . .
}
```
What does this code do?

a) This code tests whether `anObject` was created from a superclass of `Auto`.
b) This code creates a subclass type object from a superclass type object.
c) This class safely converts an object of any type to an object of type `Auto`.
d) This code safely converts an object of type `Auto` or a subclass of `Auto` to an object of type `Auto`.
Answer: d

101. To test whether an object belongs to a particular type, use ___.
a) the `this` reserved word.
b) the `subclassOf` reserved word.
c) the `instanceof` operator.
d) the `equals` method.
Answer: c

Chapter12: Object-Oriented Design

1) Which of the following most likely indicates that you have chosen a good name for your class?
a) The name consists of a single word.
b) You can tell by the name what an object of the class is supposed to represent.
c) You can easily determine by the name how many concepts the class represents.
d) The name describes what task the class will perform.
Answer: b

2) Which of the following questions should you ask yourself in order to determine if you have named your class properly?
a) Does the class name contain 8 or fewer characters?
b) Is the class name a verb?
c) Can I visualize an object of the class?
d) Does the class name describe the tasks that this class will accomplish?
Answer: c
3) Which of the following would be an appropriate name for a game-related class?
a) `FinalizeLevelOne`
b) `InitialLevel`
c) `ResetTheCurrentLevel`
d) `AscendToFinalLevel`
Answer: b

4) Which of the following would be an appropriate name for a class used in an inventory application?
a) `Product`
b) `InitializeInventoryLevel`
c) `ResetCurrentInventoryLevel`
d) `ReceiveNewProducts`
Answer: a

5) You are designing an application to support an automobile rental agency. Which of the following probably should NOT be represented as an object?
a) Auto
b) Customer
c) Payment amount
d) Rental contract
Answer: c

6) You are designing an application to support a veterinary clinic. Which of the following probably should NOT be represented as an object?
a) Pet
b) Customer
c) Medical service performed
d) Drug dosage
Answer: d

7) Classes often correspond to ____ in a requirements description.
a) Verbs b) Nouns
c) Dependencies d) Relationships
Answer: b

8) A/an ____ represents a set of objects with the same behavior.

a) Association b) Aggregation
c) Dependency d) Class
Answer: d

9) After you have identified a set of classes needed for a program, you should now ____.
a) Define the behavior of each class.
b) Look for nouns that describe the tasks.
c) Begin writing the code for the classes.
d) Establish the relationships between the classes.
Answer: a

10) A CRC card describes ____.
a) A class, its responsibilities, and its collaborating classes.
b) A class, its methods, and its constructors.
c) A class, its relationships, and its collaborating classes.
d) A class, its requirements, and its concurrent classes.
Answer: a

11) When using the CRC method, other classes that are needed to fulfill the responsibilities of a given class are called its ____.
a) Related classes. b) Responsible classes.
c) Collaborators. d) Concurrent classes.
Answer: c

12) When using the CRC method, if you determine that a class cannot carry out a particular task by itself and needs the help of another class to complete the task, you should ____.
a) Do nothing on this class's CRC card. The task will be shown on the other class's CRC card.
b) Add this class onto the other class's CRC card as a collaborator.
c) Add the other class onto this class's CRC card as a collaborator.
d) Add each class onto the other class's CRC card as a collaborator.
Answer: c

13) Which of the following is the most important consideration when designing a class?
a) Each class should represent an appropriate mathematical concept.
b) Each class should represent a single concept or object from the problem domain.
c) Each class should represent no more than three specific concepts.
d) Each class should represent multiple concepts only if they are closely related.
Answer: b

14) Under which of the following conditions would the public interface of a class be considered cohesive?
a) All of its features are public and none of its features are static.
b) The quality of the public interface is rated as moderate to high.
c) All of its features are related to the concept that the class represents.
d) It is obvious that the public interface refers to multiple concepts.
Answer: c

15) A class (`ClassOne`) is considered to have a dependency on another class (`ClassTwo`) under which of the following conditions?
a) Each class uses objects of the other.
b) The public interfaces of both classes are cohesive.
c) `ClassTwo` uses objects of `ClassOne`.
d) `ClassOne` uses objects of `ClassTwo`.
Answer: d

16) A UML class diagram would be most useful in visually illustrating which of the following?
a) the cohesiveness of a class's interface.
b) the amount of complexity of a class's interface.
c) dependencies between classes.
d) relationships between classes and their interfaces.
Answer: c

17) Why is it generally considered good practice to minimize coupling between classes?
a) Low coupling increases the operational efficiency of a program.
b) High coupling implies less interface cohesion.
c) High coupling increases program maintenance and hinders code reuse.
d) Low coupling decreases the probability of code redundancy.
Answer: c

18) Dependency between classes denotes that ____.
a) Objects of one class inherit from objects of its superclass.
b) Objects of one class can be inherited by its subclasses.
c) Objects of one class implement an interface.
d) Methods in the objects of one class use an object of another class.
Answer: d

19) In a UML diagram, dependency is denoted by ____.

a) A dotted/dashed line with a closed arrow tip.
b) A solid line with a closed arrow tip.
c) A dotted/dashed line with an open arrow tip.
d) A solid line with an open arrow tip.
Answer: c

20) If many classes of a program depend on each other, we say that ____.

a) cohesiveness is high. b) cohesiveness is low.
c) coupling is high. d) coupling is low.
Answer: c

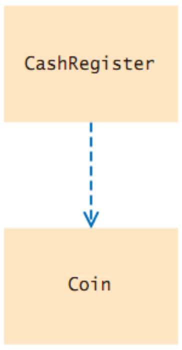
21) UML means____.

a) Unified Mode Language
b) User Modeling Language
c) User Mode Language.
d) Unified Modeling Language.
Answer: d

22) The dependency relationship is sometimes referred to as the ____ relationship.

a) is-a
b) has-a
c) depends-on
d) knows-about
Answer: d

23) Which statement correctly describes the class relationship shown in this diagram?



a) `CashRegister` class depends on `Coin` class
b) `Coin` class depends on `CashRegister` class
c) `CashRegister` class aggregates `Coin` class.
d) `Coin` class inherits from `CashRegister` class.
Answer: a

24) Which of the following code snippets denotes that the `Purse` class depends on the `Wallet` class?

a)
```
public class Purse extends Wallet
```
b)
```
public class Wallet extends Purse
```
c)
```
public class Wallet
{
   private Purse aPurse;
}
```
d)
```
public class Purse
{
   private Wallet aWallet;
}
```
Answer: d

25) Which of the following code snippets denotes that the `Pen` class depends on the `Ink` class?

a)
```
public class Pen extends Ink
```
b)
```
public class Ink extends Pen
```
c)
```
public class Pen
{
   private Ink anInk;
}
```
d)
```
public class Ink
{
   private Pen aPen;
}
```
Answer: c

26) Which of the following code snippets denotes that the `Kitchen` class depends on the `Stove` class?

a)
```
public class Kitchen extends Stove
```
b)
```
public class Stove
{
   private Kitchen[] kitchens;
}
```
c)
```
public class Kitchen
{
   private Stove aStove;
}
```
d)
```
public class Kitchen implements Stove
```
Answer: c

27) ____ is often described as the *has-a* relationship.

a) Inheritance. b) Aggregation.
c) Polymorphism. d) Dependency.
Answer: b

28) Which of the following statements is correct?

a) Dependency is a stronger form of aggregation.
b) Aggregation indicates high cohesiveness.
c) Aggregation indicates low coupling.
d) Aggregation is a stronger form of dependency.
Answer: d

29) Consider the following code snippet:

```
public class Motorcycle
{
   private Tire[] tires;
   . . .
}
```
This code is best described as an example of ____.
a) Inheritance. b) Aggregation.
c) Polymorphism. d) Association.
Answer: b

30) Consider the following code snippet:

```
public class Motorcycle
{
   private Tire[] tires;
   . . .
}
```
Which of the following statements correctly describes the relationship between the `Motorcycle` and `Tire` classes?
a) `Motorcycle` exhibits the *has-a* relationship with regards to `Tire`.

b) `Motorcycle` exhibits the *is-a* relationship with regard to `Tire`.
c) `Tire` exhibits the *has-a* relationship with regards to `Motorcycle`.
d) `Tire` exhibits the *is-a* relationship with regards to `Motorcycle`.
Answer: a

31) Consider the following code snippet:
```
public class Purse
{
   private Coin[] coins;
   . . .
}
```
This code is best described as an example of ____.

a) Inheritance. b) Aggregation.
c) Polymorphism. d) Association.
Answer: b

32) Consider the following code snippet:
```
public class Purse
{
   private Coin[] coins;
   . . .
}
```
Which of the following statements is correct?

a) `Purse` aggregates `Coin`.
b) `Coin` depends on `Purse`.
c) The `Coin` class must be declared as an inner class within `Purse`.
d) `Coin` inherits from `Purse`.
Answer: a

33) Aggregation denotes that ____.

a) Objects of one class inherit from objects of its superclass.
b) Objects of one class can be inherited by its subclasses.
c) Objects of one class contain references to objects of its superclass.
d) Objects of one class contain references to objects of another class.
Answer: d

34) A `CashRegister` class contains an array list of `Coin` objects. This is best described as an example of ____.

a) Association
b) Inheritance
c) Cohesiveness
d) Aggregation
Answer: d

35) A `Quiz` class contains an array of `Question` objects. This is best described as an example of ____.

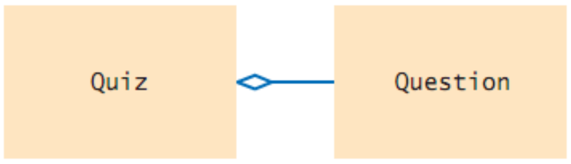
a) Aggregation
b) Cohesiveness
c) Association
d) Inheritance
Answer: a

36) In a UML diagram, aggregation is denoted by ____.

a) A solid line with a diamond-shaped symbol next to the aggregating class.
b) An arrow with an open triangle pointing to the aggregating class.
c) A dotted line with a diamond-shaped symbol next to the aggregating class.
d) A dotted line with an open arrow tip pointing to the aggregating class.
Answer: a

37) In general, you need ____ when an object needs to remember another object between method calls.
a) inheritance
b) aggregation
c) association
d) an interface implementation
Answer: b

38) Which statement correctly describes the class relationship shown in this diagram?



a) `Quiz` class depends on `Question` class
b) `Question` class aggregates `Quiz` class
c) `Quiz` class aggregates `Question` class.
d) `Question` class inherits from `Quiz` class.
Answer: c

39) Which of the following code snippets denotes that the `Fleet` class aggregates the `Taxi` class?

a)
```
public class Fleet extends Taxi
```
b)
```
public class Taxi extends Fleet
```
c)
```
public class Fleet
{
   private ArrayList<Taxi> taxicabs;
}
```
d)
```
public class Taxi
{
   private Fleet myFleet;
}
```
Answer: c

40) Which of the following code snippets denotes that the `Kitchen` class aggregates the `Dish` class?
a)
```
public class Kitchen extends Dish
```
b)
```
public class Dish extends Kitchen
```
c)
```
public class Kitchen
{
   private Dish[] dishes;
}
```
d)
```
public class Dish
{
   private Kitchen myKitchen;
}
```
Answer: c

41) ____ is often described as the *is-a* relationship.

a) Inheritance. b) Aggregation.
c) Polymorphism. d) Dependency.
Answer: a

42) Consider the following code snippet:
```
public class Motorcycle extends Vehicle
{
   private Tire[] tires;
   . . .
}
```
This code is best described as an example of ____.

a) Inheritance and multiplicities.
b) Aggregation and dependency.
c) Inheritance and aggregation.
d) Inheritance and association.
Answer: c

43) Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{ . . . }
```

Which of the following statements correctly describes the relationship between the `Motorcycle` and `Vehicle` classes?

a) `Motorcycle` exhibits the *has-a* relationship with regards to `Vehicle`.
b) `Motorcycle` exhibits the *is-a* relationship with regard to `Vehicle`.
c) `Vehicle` exhibits the *has-a* relationship with regards to `Motorcycle`.
d) `Vehicle` exhibits the *is-a* relationship with regards to `Motorcycle`.
Answer: b

44) Consider the following code snippet:

```
public class Motorcycle extends Vehicle
{
   private Tire[] tires;
   private Engine anEngine;
   . . .
}
```

Which of the following statements describing relationships between the `Motorcycle, Tire, Engine` and `Vehicle` classes is NOT correct?

a) `Motorcycle` exhibits the *has-a* relationship with regards to `Tire`.
b) `Motorcycle` exhibits the *is-a* relationship with regard to `Vehicle`.
c) `Vehicle` exhibits the *has-a* relationship with regards to `Motorcycle`.
d) `Motorcycle` exhibits the *knows-about* relationship with regards to `Engine`.
Answer: c

45) Consider the following code snippet:

```
public class PowerBoat extends Vessel
{
   private Engine[] engines;
   . . .
}
```

This code is best described as an example of ____.
a) Inheritance and multiplicities.
b) Aggregation and association.
c) Inheritance and aggregation.
d) Inheritance and dependency.
Answer: c

46) Consider the following code snippet:

```
public class SailBoat extends Vessel
{
   private Engine[] engines;
   private Sail mainsail;
   . . .

}
```

Which of the following statements is NOT correct?
a) `SailBoat` inherits from `Vessel`.
b) `Sail` depends on `Sailboat`.
c) `Sailboat` aggregates `Engine`.
d) `SailBoat` aggregates `Sail`.
Answer: b

47) Consider the following code snippet:

```
public class SailBoat extends Vessel
{ . . . }
public class Catamaran extends SailBoat
{ . . . }
```

Which of the following statements is NOT correct?
a) `SailBoat` inherits from `Vessel`.
b) `Catamaran` inherits from `Vessel`.
c) `Catamaran` inherits from `Sailboat`.
d) `Catamaran` depends on `SailBoat`.
Answer: d

48) Consider the following code snippet:
```
public class Manager extends Employee
{
   private Project[] projects;
   private Address address;
   . . .
}
```
Which of the following statements is NOT correct?

a) `Manager` inherits from `Employee`.
b) `Address` aggregates `Manager`.
c) `Manager` aggregates `Address`.
d) `Manager` aggregates `Project`.
Answer: b

49) In a UML diagram, inheritance is denoted by ____.

a) A solid line with a diamond-shaped symbol next to the superclass.
b) A solid line with a closed arrow tip pointing to the superclass.
c) A dotted line with an open arrow tip pointing to the superclass.
d) A solid line with an open arrow tip pointing to the superclass.
Answer: b

50) Which of the followings statements about class relationships is correct?
a) Inheritance represents the *is-a* relationship, while aggregation represents the *has-a* relationship.
b) Inheritance represents the *has-a* relationship, while dependency represents the *uses* relationship.
c) Aggregation represents the *is-a* relationship, while inheritance represents the *uses* relationship.
d) Aggregation represents the *has-a* relationship, while dependence represents the *is-a* relationship.
Answer: a

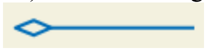
51) Which of the following code snippets denotes that the `Lime` class inherits from the `Citrus` class?
a)
```
public class Lime extends Citrus
```
b)
```
public class Citrus extends Lime
```
c)
```
public class Lime
{
   private ArrayList<Citrus> fruits;
}
```
d)
```
public class Citrus
{
   private ArrayList<Lime> limes;
}
```
Answer: a

52) Which of the following code snippets denotes that the `Rose` class inherits from the `Flower` class?
a)
```
public class Flower extends Rose
```
b)
```
public class Rose extends Flower
```
c)
```
public class Rose
{
   private Flower[] flowers;
}
```
d)
```
public class Flower
{
   private Rose[] limes;
}
```
Answer: b

53) When designing classes, if you find classes with common behavior you should ____.
a) Combine them all into a single class.
b) Place the common behavior into a superclass.
c) Place the common behavior into a subclass.
d) Give them similar names.
Answer: b

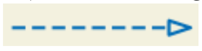
54) In a UML diagram, an interface implementation is denoted by ____.
a) A dotted line with a closed arrow tip.
b) A solid line with a closed arrow tip.
c) A dotted line with an open arrow tip.
d) A solid line with an open arrow tip.
Answer: a

55) In a UML diagram, the relationship symbol shown below denotes ____.



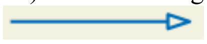
a) inheritance b) aggregation
c) dependency d) interface implementation
Answer: b

56) In a UML diagram, the relationship symbol shown below denotes ____.



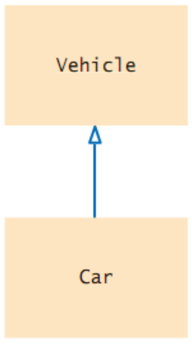
a) inheritance b) aggregation
c) dependency d) interface implementation
Answer: d

57) In a UML diagram, the relationship symbol shown below denotes ____.



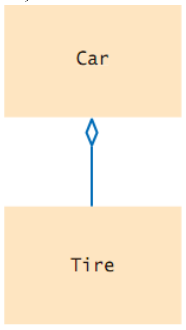
a) inheritance b) aggregation
c) dependency d) interface implementation
Answer: a

58) Which statement correctly describes the class relationship shown in this diagram?



a) `Vehicle` class depends on `Car` class
b) `Car` class depends on `Vehicle` class
c) `Vehicle` class inherits from `Car` class.
d) `Car` class inherits from `Vehicle` class.
Answer: d

59) Which statement correctly describes the class relationship shown in this diagram?



a) `Tire` class aggregates `Car` class.
b) `Car` class aggregates `Tire` class.
c) `Car` class inherits from `Tire` class.
d) `Tire` class inherits from `Car` class.
Answer: b

60) You have determined a need for a `Book` class and a `Page` class in your program. Which relationship is most appropriate between these classes?
a) Inheritance
b) Aggregation
c) Dependency
d) Interface implementation
Answer: b

61) You have determined a need for an `Employee` class and a `TemporaryEmployee` class in your program. Which relationship is most appropriate between these classes?
a) Inheritance
b) Aggregation
c) Dependency
d) Interface implementation
Answer: a
62) ____ relationships come from the collaboration columns on the CRC cards.
a) Association
b) Inheritance
c) Dependency
d) Attribute
Answer: c

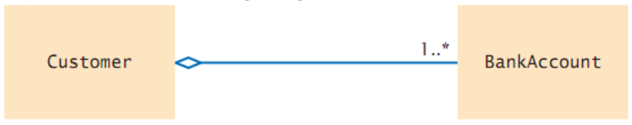
63) When using CRC cards, UML diagrams should be created ___.
a) Prior to discovering classes.
b) After you have discovered classes and their relationships.
c) During the implementation phase.
d) To produce final documentation of the system.
Answer: b

64) Which of the following statements about class attributes is true?
a) An attribute always corresponds to an instance variable of the class.
b) An attribute is an externally observable property that objects of the class have.
c) An attribute is used to represent aggregation with another class.
d) Attributes cannot be represented in UML diagrams.
Answer: b

65) How does a UML diagram denote classes and their attributes and methods?
a) A class rectangle contains the class name in the top, methods in the middle, and attributes in the bottom.
b) A class rectangle contains the class name in the top, attributes in the middle, and methods in the bottom.
c) A class rectangle contains the methods in the top, class name in the middle, and attributes in the bottom.
d) A class rectangle contains the attributes in the top, methods in the middle, and class name in the bottom.
Answer: b

66) How does a UML diagram denote a multiplicity of one or more in an aggregation relationship?
a) *
b) 1..*
c) *..1
d) 1..n
Answer: b

67) Given the following diagram:



What does this diagram indicate about the relationship between the customers and bank accounts?
a) A bank account may be owned by only one customer.
b) A bank account may be owned by one or more customers.
c) A customer may have only one bank account.
d) A customer may have one or more bank accounts.
Answer: d

68) Which of the following statements about associations between classes is true?
a) A class is associated with another class if both classes are in the same package.
b) A class is associated with another class if the class inherits from the other class.
c) A class is associated with another class if the other class inherits from this class.
d) A class is associated with another class if you can navigate from objects of one class to objects of the other class.
Answer: d

69) You have determined the need for a `File` class and a `Folder` class in your program. Which of the following would best describe the relationship between these classes?
a) Aggregation
b) Association
c) Composition
d) Inheritance
Answer: c

70) If you have parallel arrays or array lists of the same length which each store a part of what could be considered an object, _____.
a) You should rewrite the program to use a two-dimensional array.
b) You should rewrite the program to use a two-dimensional array list.
c) You should create a class to hold the related data in the ith slice of each array, and then create an arraylist of objects.
d) There are no alternatives to this type of program design.
Answer: c
71) Parallel arrays are _____.

a) Two or more arrays or array lists of the same length which each store a part of what could be considered an object.
b) Two or more arrays or array lists of the same length.
c) Two or more arrays or array lists of the same data type.
d) Two or more arrays or array lists that are declared one after the other in a program.
Answer: a

72) To improve the quality of the public interface of a class, _____.

a) You should maximize cohesion and minimize inheritance.
b) You should minimize cohesion and maximize coupling.
c) You should maximize cohesion and minimize coupling.
d) You should minimize cohesion and maximize aggregation.
Answer: c

73) Select a code segment to complete the `Name` class, so that it reflects a dependency relationship between `Name` and `String`.

```
public class Name ________________________________
```
a)
```
extends String
{
   …
}
```
b)
```
{
   private String first;
   private String second;
   …
}
```
c)
```
implements String
{
   …
}
```

d)
```
{
   private ArrayList<String> names;
   …
}
```
Answer: b

74) Select a code segment to complete the `Team` class, so that it reflects an aggregation relationship between `Team` and `Player`.
```
public class Team ________________________________
```

a)
```
extends Player
{
   …
}
```
b)
```
{
   private Player aPlayer;
   …
}
```
c)
```
implements Player
{
   …
}
```
d)
```
{
   private ArrayList<Player> players;
   …
}
```
Answer: d

75) Select a code segment to complete the `Player` class, so that it reflects an inheritance relationship between `Player` and `Person`.
```
public class Player ___________________________
```
a)
```
extends Person
{
   …
}
```
b)
```
{
   private Person thePlayer;
   …
}
```
c)
```
implements Person
{
   …
}
```
d)
```
{
   private ArrayList<Person> players;
   …
}
```
Answer: a

76) Select a code segment to complete the `SmartPhone` class, so that it reflects an interface implementation relationship between `SmartPhone` and `MP3Player`.
```
public class SmartPhone ___________________________
```
a)
```
extends MP3Player
{
   …
}
```
b)
```
{
   private MP3Player musicPlayer;
   …
}
```
c)
```
implements MP3Player
{
   …
}
```
d)
```
implements interface MP3Player
{
   …
}
```
Answer: c

77) The textbook recommends a five-part program development process consisting of the following activities:
I Use UML diagrams to record class relationships.
II Gather requirements.
III Implement the program.
IV Use CRC cards to identify classes.
V Use javadoc to document method behavior.

Which is the correct order in which these activities should be completed?
a) IV, II, I, III, V
b) IV, II, I, V, III
c) II, IV, I, III, V
d) II, IV, I, V, III
Answer: d

78) Before you begin designing a solution, you should ____.
a) Document the methods that will be needed.
b) Gather and document all requirements for the program in plain English.
c) Determine the classes that will be needed.
d) Create the UML diagrams to represent the classes.
Answer: b

79) You are designing a software solution for an automobile rental company. You have decided that the following nouns apply to the requirements: Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate, Total. Which of these should be represented as classes?
a) All of them.
b) Auto, Customer, Address, Rental Contract.
c) Auto, Customer, Address, Rental Contract, Mileage.
d) Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate.
Answer: b

80) You are designing a software solution for an automobile rental company. You have decided that the following nouns apply to the requirements: Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate, Total. Which of these should be represented as instance variables?
a) Mileage
b) Mileage, Daily Rate.
c) Mileage, Rental Date, Daily Rate.
d) Mileage, Rental Date, Daily Rate, Total.
Answer: c

81) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following nouns apply to the requirements: Customer, Address, Pet, Visit, Visit Date, Service Charge, Total Charge, Next Appointment. Which of these should be represented as classes?
a) All of them.
b) Customer, Address, Pet, Visit.
c) Customer, Address, Pet, Visit, Service Charge.
d) Customer, Address, Pet, Visit, Service Charge, Total Charge, Visit Date, Next Appointment.
Answer: b

82) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following nouns apply to the requirements: Customer, Address, Pet, Visit, Visit Date, Service Charge, Total Charge, Next Appointment. Which of these should be represented as instance variables?

a) Service Charge
b) Service Charge, Visit Date.
c) Service Charge, Visit Date, Next Appointment.
d) Service Charge, Visit Date, Next Appointment, Total Charge.
Answer: c

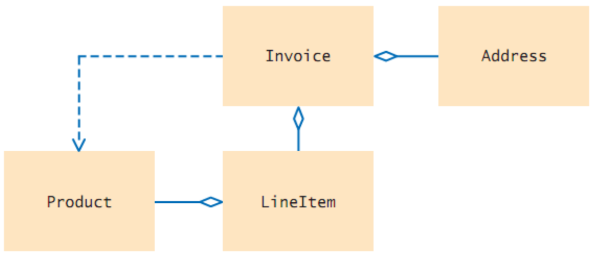
83) You are designing a software solution for an automobile rental company. A customer may rent only a single auto at a given time. You have decided that the following classes are needed: Auto, Customer, Address, Rental Contract. Which of these should be represented as aggregation?

a) Rental Contract class aggregates Address class
b) Rental Contract class aggregates Customer and Address classes.
c) Rental Contract class aggregates Customer, Address, and Auto classes.
d) Rental Contract class does not aggregate any of the other classes.
Answer: c

84) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following classes are needed: Customer, Address, Pet, and Visit. Which of these should be represented as aggregation?

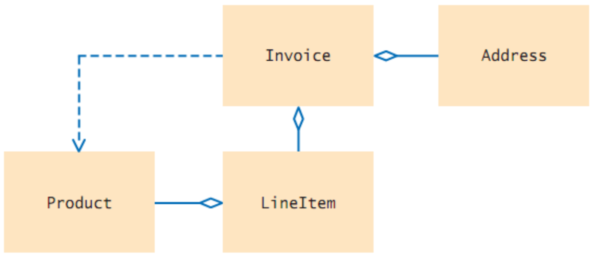
a) Customer class aggregates Address, Pet, and Visit classes.
b) Visit class aggregates Address class.
c) Pet class aggregates Address class.
d) Pet class aggregates Customer class.
Answer: a

85) Given the following diagram showing class relationships:



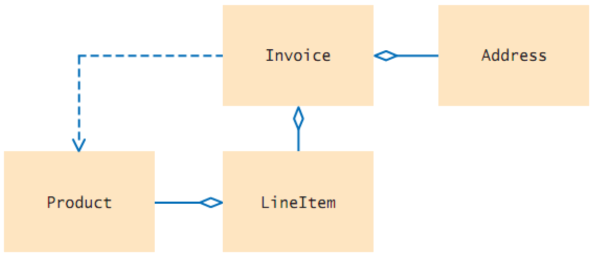
What type of relationship is shown between `Invoice` and `LineItem`?
a) `LineItem` inherits from `Invoice`.
b) `LineItem` aggregates `Invoice`.
c) `Invoice` aggregates `LineItem`.
d) `Invoice` inherits from `LineItem`.
Answer: c

86) Given the following diagram showing class relationships:



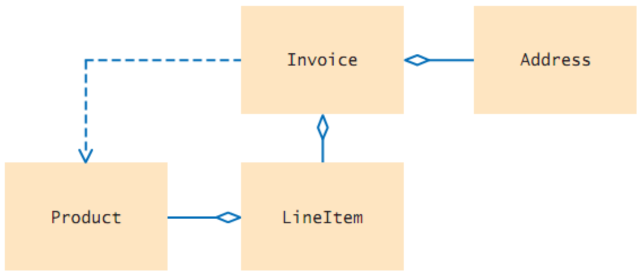
What type of relationship is shown between `Invoice` and `Product`?
a) `Product` inherits from `Invoice`.
b) `Product` aggregates `Invoice`.
c) `Invoice` inherits from `Product`.
d) `Invoice` is dependent on `Product`.
Answer: d

87) Given the following diagram showing class relationships:



What type of relationship is shown between `LineItem` and `Product`?
a) `Product` inherits from `LineItem`.
b) `Product` aggregates `LineItem`.
c) `LineItem` aggregates `Product`.
d) `LineItem` inherits from `Product`.
Answer: c

88) Given the following diagram showing class relationships:



What type of relationship is shown between `Invoice` and `Address`?
a) `Address` inherits from `Invoice`.
b) `Invoice` aggregates `Address`.
c) `Address` aggregates `Invoice`.
d) `Invoice` inherits from `Address`.
Answer: b

89) The final step of the design phase recommended by the textbook is to ____.

a) Write the documentation of the discovered classes and methods
b) Write the code for each class and all of its methods.
c) Identify the classes and methods that will be required.
d) Create the UML diagrams to represent the classes.
Answer: a

90) When documenting discovered classes and methods during the design phase, you should ____.

a) Create a Java source file for each class, write the method signatures, and use javadoc comments to describe the methods, but leave the method bodies blank.
b) Create a Java source file for each class and write code to implement all of its methods, and use javadoc comments to document them.
c) Create a Java source file for each class, use javadoc comments to document the methods that are to be added later, but do not write any code for the methods.
d) Documentation of methods should not be done during the design phase.
Answer: a

91) Which of the following can be used to record the behavior of classes?

a) Javadoc comments
b) Nouns and verbs in the problem description
c) Polymorphism
d) UML notation
Answer: a

92) During the implementation phase, which of the following statements is true?

a) Aggregated classes will yield subclasses.
b) Aggregated classes will yield instance variables.
c) Aggregated classes will yield class packages.
d) Aggregated classes will yield interfaces.
Answer: b

93) During the implementation phase, which of the following statements is true?

a) Method parameter variables are available from the CRC cards.
b) Method parameter variables are available from the javadocs created during the design phase.
c) You must determine the method parameter variables during implementation phase.
d) Method parameter variables are available from the UML diagrams.
Answer: b

94) During the implementation phase, which of the following statements is true?

a) Class instance variables are available from the CRC cards.
b) Class instance variables are available from the javadocs created during the design phase.
c) You must determine the class instance variables during implementation phase.
d) Class instance variables can be determined from the UML diagrams.
Answer: d

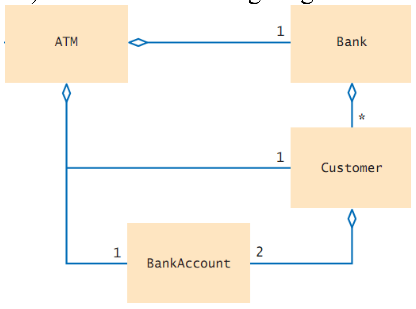
95) When using UML to create state diagrams, a state is denoted by ____.

a) A rectangle with double border.
b) A rectangle with rounded corners.
c) An oval with double border.
d) A standard rectangle with square corners.
Answer: b

96) When using UML to create state diagrams, state change is denoted by ____.

a) Arrows with labels that connect rectangles.
b) Double arrows with labels that connect rectangles.
c) An oval between two rectangles, connected with arrows.
d) Dividing the state rectangle into several parts, one for each state change.
Answer: a

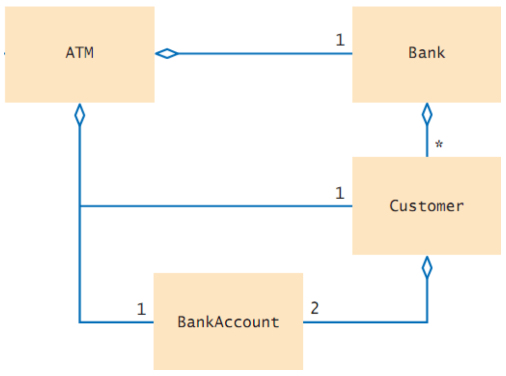
97) Given the following diagram showing class relationships:



What of the following best describes the type of relationship shown between `Bank` and `Customer`?

a) `Customer` aggregates `Bank`, indicating that a customer may deal with many banks.
b) `Bank` aggregates `Customer`, indicating that a bank may deal with many customers.
c) `Bank` depends on `Customer`, indicating that a bank may deal with many customers.
d) `Customer` depends on `Bank`, indicating that a customer may deal with many banks.
Answer: b

98) Given the following diagram showing class relationships:



What type of relationship is shown between `BankAccount` and `Customer`?
a) `Customer` aggregates `BankAccount`, indicating that a customer may have 2 bank accounts.
b) `BankAccount` aggregates `Customer`, indicating that 2 customers may share a bank account.
c) `BankAccount` depends on `Customer`, indicating that many customers may have 2 bank accounts.
d) `Customer` depends on `BankAccount`, indicating that a customer may have 2 bank accounts.
Answer: a

99) Suppose that the invoice-printing application from section 12.3 needs to be enhanced by including the computation and printing of sales taxes as needed. Since some vendors charge no sales taxes, the original `Invoice` class needs to be preserved. Select the code segment that best illustrates reuse of the existing `Invoice` class.

```
public class TaxableInvoice ______________________________
```
a)
```
extends Invoice
{
   …
}
```
b)
```
{
   private Invoice originalInvoice;
   …
}
```
c)
```
implements Invoice
{
   …
}
```
d)
```
{
   private ArrayList<Invoice> invoices;
   …
}
```
Answer: a

100) Regarding the invoice-printing application from section 12.3, it is decided that a `Customer` class is needed to store customer information, including the name, billing address, and a customer id number to be printed at the top of each invoice. Since the existing `Address` class already stores the customer's name and physical address, the new `Customer` class can simply reuse `Address`. Select the code segment that best illustrates how the `Customer` class can reuse the existing `Address` class.

```
public class Customer ______________________________
```

a)

```
extends Address
{
   private String idNumber;
   …
}
```

b)

```
{
   private Address information;
   private String idNumber;
   …
}
```

c)

```
implements Address
{
   private String idNumber;
   …
}
```

d)

```
{
   private Address information;
   private String name;
   private String street;
   private String city;
   private String state;
   private String zip;
   private String idNumber;
   …
}
```

Answer: b

101) Suppose that the invoice-printing application from section 12.3 needs to be enhanced by making it possible for class `InvoicePrinter` to store several invoices to be printed. Select the code segment that best illustrates reuse of the existing `Invoice` class for this purpose.

```
public class InvoicePrinter ______________________________
```

a)

```
extends Invoice
{
   …
}
```

b)

```
{
   public static void main(String[] args)
   {
      Invoice invoice1;
      Invoice invoice2;
      …
}
```

c)

```
implements Invoice
{
   …
}
```

d)

```
{
   public static void main(String[] args)
   {
      ArrayList<Invoice> invoices;
      …
   }
}
```

Answer: d

102) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, the partial `Employee` class below is developed. Select the method header that best completes the class, according to the method comments.

```
public class Employee
{
   private int hoursWorked;

   /**
      Computes the weekly salary for this employee.
      @param hourlyRate the rate per hour earned by the employee
      @return the weekly salary
   */
   __________________________________________
   {
      // method body
   }
}
```

a) `public void computeSalary(double hourlyRate)`
b) `public double computeSalary(double hourlyRate, int hoursWorked)`
c) `public double computeSalary(double hourlyRate)`
d) `public double computeSalary(int hoursWorked)`
Answer: c

103) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, an `Employee` class is developed. In addition, the `Payroll` class is designed to store and process information for various employees. Select the code segment that best completes the `Payroll` class.

```
public class Payroll ____________________________
```

a)
```
extends Employee
{
   …
}
```
b)
```
{
   public static void main(String[] args)
   {
      Employee anEmployee;
      …
}
```
c)
```
{
   public static void main(String[] args)
   {
      int hoursWorked;
      double hourlyRate;
      …
   }
}
```
d)
```
{
   public static void main(String[] args)
   {
      ArrayList<Employee> employees;
      …
   }
}
```
Answer: d

104) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, the partial `Employee` class below is developed. Select the method header that best completes the class, according to the method comments.

```
public class Employee
{
   private double hourlyRate;
   private int hoursWorked;

   /**
      Modifies the hourly rate for this employee.
      @param newHourlyRate the rate per hour earned by the employee
   */
   __________________________________________
   {
      // method body
   }
}
```
a) `public void changeRate(double newHourlyRate)`
b) `public void changeRate(double newHourlyRate, int hoursWorked)`
c) `public double changeRate(double newHourlyRate)`
d) `public double changeRate(double newHourlyRate, int hoursWorked)`
Answer: a

Chapter 13: Recursion

1) What is required to make a recursive method successful?
I special cases that handle the simplest computations directly
II a recursive call to simplify the computation
III a mutual recursion
a) I   b) II
c) I and II   d) I, II, and III
Answer: c

2) Consider the `getArea` method from the textbook shown below.
```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   else if (width == 1) { return 1; }     // line #2
   else
   {
      Triangle smallerTriangle = new Triangle(width - 1); // line #3
      int smallerArea = smallerTriangle.getArea();  // line #4
      return smallerArea + width;  // line #5
   }
}
```
Where is/are the recursive call(s)?
a) line #1   b) line #2
c) lines #1 and #2   d) line #4
Answer: d

3) Consider the `getArea` method from the textbook shown below.
```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   else if (width == 1) { return 1; }     // line #2
   else
   {
      Triangle smallerTriangle = new Triangle(width - 1); // line #3
      int smallerArea = smallerTriangle.getArea();  // line #4
      return smallerArea + width;  // line #5
   }
}
```
Where is/are the terminating condition(s)?
a) line #1   b) line #2
c) lines #1 and #2   d) line #4
Answer:

4) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   else if (width == 1) { return 1; }  // line #2
   else
   {
      Triangle smallerTriangle = new Triangle(width - 1); // line #3
      int smallerArea = smallerTriangle.getArea();  // line #4
      return smallerArea + width;  // line #5
   }
}
```
Assume the code in line #3 is changed to:

```
Triangle smallerTriangle = new Triangle(width);
```

This change would cause infinite recursion for which triangles?
a) Those with width equal to 0.
b) Those with width equal to 1.
c) Those with width greater than or equal to 2.
d) Triangles of any width.
Answer: c

5) Consider the `getArea` method from the book shown below.

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   else if (width == 1) { return 1; }      // line #2
   else
   {
      Triangle smallerTriangle = new Triangle(width - 1); // line #3
      int smallerArea = smallerTriangle.getArea();  // line #4
      return smallerArea + width;  // line #5
   }
}
```

Assume lines #1 and #2 were changed to this:

```
   if (width == 1) { return 1; } // new line #1-2
```

What will happen when this code is executed?
a) A negative or zero width would cause problems.
b) Nothing - the method would still be correct.
c) We would lose our only recursive case.
d) A positive width would reduce the correct area by 1.
Answer: a

6) Consider the `getArea` method from the textbook shown below.

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   else if (width == 1) { return 1; } // line #2
   else
   {
      Triangle smallerTriangle = new Triangle(width - 1); // line #3
      int smallerArea = smallerTriangle.getArea();  // line #4
      return smallerArea + width;  // line #5
   }
}
```

Assume line #1 is replaced with this line:

```
if (width <= 0) {return width;}
```

What will be the result?
a) The method will still return correct results for all non-negative width triangles.
b) The method will return incorrect results for triangles with width equal to 0.
c) The method will return incorrect results for triangles with width equal to 1.
d) The method will return area to be one too high for all triangles.
Answer: a

7) Consider the following code snippet for recursive addition:

```
int add(int i, int j)
{
   // assumes i >= 0
   if (i == 0)
   {
      return j;
   }
   else
   {
      return add(i - 1, j + 1);
   }
}
```

Identify the terminating condition in this recursive method.
a) `if (i == 0)`   b) `return j`
c) `return add(i - 1, j + 1)`   d) there is no terminating condition
Answer: a

8) Consider the following code snippet for calculating Fibonacci numbers recursively:

```
int fib(int n)
{
   // assumes n >= 0
   if (n <= 1)
   {
      return n;
   }
   else
   {
      return (fib(n - 1) + fib(n - 2));
   }
}
```

Identify the terminating condition.

a) `n < 1`
b) `n <= 1`
c) `fib(n – 1)`
d) `fib(n - 1) + fib(n – 1)`
Answer: b

9) Consider the following recursive code snippet:

```
public static int mystery(int n, int m)
{
   if (n <= 0)
   {
      return 0;
   }
   if (n == 1)
   {
      return m;
   }
   return m + mystery(n - 1, m);
}
```

Identify the terminating condition(s) of method `mystery`?

a) `n <= 0`
b) `n == 1`
c) `n <= 0` or `n == 1`
d) `n > 0`
Answer: c

10) Consider the following recursive code snippet:

```
public int mystery(int n, int m)
{
   if (n == 0)
   {
      return 0;
   }
   if (n == 1)
   {
      return m;
   }
   return m + mystery(n - 1, m);
}
```

What value is returned from a call to `mystery(1,5)`?
a) 1
b) 5
c) 6
d) 11
Answer: b

11) Consider the following recursive code snippet:

```
public int mystery(int n, int m)
{
   if (n == 0)
   {
      return 0;
   }
   if (n == 1)
   {
      return m;
   }
   return m + mystery(n - 1, m);
}
```

What value is returned from a call to `mystery(3,6)`?

a) 3
b) 6
c) 18
d) 729
Answer: c

12) Consider the recursive method `myPrint`:

```
public void myPrint(int n)
{
   if (n < 10)
   {
      System.out.print(n);
   }
   else
   {
      int m = n % 10;
      System.out.print(m);
      myPrint(n / 10);
   }
}
```

What is printed for the call `myPrint(8)`?
a) 10
b) 8
c) 4
d) 21
Answer: b

13) Consider the recursive method `myPrint` in this code snippet:

```
public void myPrint(int n)
{
   if (n < 10)
   {
      System.out.print(n);
   }
   else
   {
      int m = n % 10;
      System.out.print(m);
      myPrint(n / 10);
   }
}
```

What is printed for the call `myPrint(821)`?

a) 821
b) 128
c) 12
d) 10
Answer: b

14) Consider the recursive method `myPrint` shown in this code snippet:

```
public void myPrint(int n)
{
   if (n < 10)
   {
      System.out.print(n);
   }
   else
   {
      int m = n % 10;
      System.out.print(m);
      myPrint(n / 10);
   }
}
```

What does this method do?
a) Prints a positive `int` value forward, digit by digit.
b) Prints a positive `int` value backward, digit by digit.
c) Divides the `int` by 10 and prints out its last digit.
d) Divides the `int` by 10 and prints out the result.
Answer: b

15) Complete the code for the recursive method `printSum` shown in this code snippet, which is intended to return the sum of digits from 1 to n:

```
public static int printSum(int n)
{
   if (n == 0)
   {
      return 0;
   }
   else
   {
      ______________________________
   }
}
```

a) `return (printSum(n - 1));`
b) `return (n + printSum(n + 1));`
c) `return (n + printSum(n - 1));`
d) `return (n - printSum(n - 1));`
Answer: c

16) Consider the code for the recursive method `mysteryPrint` shown in this code snippet:

```
public static int mysteryPrint(int n)
{
   if (n == 0)
   {
      return 0;
   }
   else
   {
      return (n + mysteryPrint(n-1));
   }
}
```

What will be printed with a call to `mysteryPrint(-4)`?
a) `0` b) `-10`
c) `-22` d) Nothing – a `StackoverflowError` exception will occur
Answer: d

17) Consider the code for the recursive method `myPrint` shown in this code snippet:

```
public static int myPrint(int n)
{
   if (n == 0)
   {
      return 0;
   {
   else
   {
      return (n + myPrint(n - 1));
   }
}
```

To avoid infinite recursion, which of the following lines of code should replace the current terminating case?

a) `if (n == -1)` b) `if (n <= 0)`

c) `if (n >= 0)` d) The terminating case as shown will avoid infinite recursion.

Answer: b

18) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   if (width == 1) { return 1; } // line #2
   Triangle smallerTriangle = new Triangle(width - 1); // line #3
   int smallerArea = smallerTriangle.getArea();  // line #4
   return smallerArea + width;  // line #5
}
```

Which line has the recursive case?

a) line #1
b) line #2
c) line #3
d) line #4

Answer: d

19) Consider the recursive method shown below:

```
public static int strangeCalc(int bottom, int top)
{
   if (bottom > top)
   {
      return -1;
   }
   else if (bottom == top)
   {
      return 1;
   }
   else
   {
      return bottom * strangeCalc(bottom + 1, top);
   }
}
```

What value will be returned with a call to `strangeCalc(4,7)`?

a) 1
b) -1
c) 120
d) 840

Answer: c

20) Consider the recursive method shown below:

```
public static int strangeCalc(int bottom, int top)
{
   if (bottom > top)
   {
      return -1;
   }
   else if (bottom == top)
   {
      return 1;
   }
   else
   {
      return bottom * strangeCalc(bottom + 1, top);
   }
}
```

What value will be returned with a call to `strangeCalc(2,3)`?

a) 1
b) 2
c) 6
d) 24

Answer: b

21) Insert the missing code in the following code fragment. This fragment is intended to recursively compute x^n, where x and n are both non-negative integers:

```
public int power(int x, int n)
{
   if (n == 0)
   {
      ____________________
   }
   else
   {
      return x * power(x, n - 1);
   }
}
```

a) `return 1;`
b) `return x;`
c) `return power(x, n - 1);`
d) `return x * power(x, n - 1);`
Answer: a

22) If recursion does not have a special terminating case, what error will occur?
a) Index out of range
b) Illegal argument
c) Stack overflow
d) Out of memory
Answer: c

23) A recursive method without a special terminating case would _________
a) end immediately.
b) not be recursive.
c) be more efficient.
d) never terminate.
Answer: d

24) Consider the following recursive code snippet:

```
public int mystery(int n, int m)
{
   if (n == 0)
   {
      return 0;
   }
   if (n == 1)
   {
      return m;
   }
   return m + mystery(n - 1, m);
}
```

What parameter values for n would cause an infinite recursion problem in the following method?
a) `n == 0`
b) `n == 1`
c) all n with `n < 0`
d) all n with `n >= 0`
Answer: c

25) ____ recursion can occur when a recursive algorithm does not contain a special case to handle the simplest computations directly.
a) Mutual
b) Non-mutual
c) Terminating condition
d) Infinite
Answer: d

26) If a recursive method does not simplify the computation within the method and the base case is not called, what will be the result?
a) The terminating condition will be executed and recursion will end.
b) The recursion calculation will occur correctly regardless.
c) This cannot be determined.
d) Infinite recursion will occur.
Answer: d

27) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   Triangle smallerTriangle = new Triangle(width - 1); // line #2
   int smallerArea = smallerTriangle.getArea();  // line #3
   return smallerArea + width;  // line #4
}
```

If line#1 was removed, what would be the result?
a) The recursive method would cause an exception for values below 0.
b) The recursive method would construct triangles whose width was negative.
c) The recursive method would terminate when the width reached 0.
d) The recursive method would correctly calculate the area of the original triangle.
Answer: b

28) When a recursive method is called, and it does not perform recursion, what must be true?
a) The terminating condition was true.
b) One recursive case condition was true.
c) All recursive case conditions were true.
d) An exception will occur in the method.
Answer: a

29) Consider the `getArea` method from the textbook shown below.

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   if (width == 1) { return 1; } // line #2
   Triangle smallerTriangle = new Triangle(width - 1); // line #3
   int smallerArea = smallerTriangle.getArea();  // line #4
   return smallerArea + width;  // line #5
}
```

Assume that line #2 is changed to this:

```
if (width == 1) { return 2; }
```

How would this affect calls to `getArea`?
a) It would add 1 to all calls except those where `width <= 0`.
b) It would make no difference to any call.
c) It would double every triangle area.
d) It would subtract 1 from all calls.
Answer: a

30) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   if (width == 1) { return 1; } // line #2
   Triangle smallerTriangle = new Triangle(width - 1); // line #3
   int smallerArea = smallerTriangle.getArea();  // line #4
   return smallerArea + width;  // line #5
}
```

Assume that line #3 is changed to this:

```
Triangle smallerTriangle = new Triangle(width);
```

This would cause infinite recursion for ____.
a) triangles with width equal to 0
b) triangles with width equal to 1
c) triangles with width greater than or equal to 2
d) triangles of any width
Answer: c

31) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   if (width == 1) { return 1; } // line #2
   Triangle smallerTriangle = new Triangle(width - 1); // line #3
   int smallerArea = smallerTriangle.getArea();  // line #4
   return smallerArea + width;  // line #5
}
```

Assume line #3 is changed to this:

```
Triangle smallerTriangle = new Triangle(width + 1)
```

When calling the `getArea` method on a `Triangle` object with `width = 4`, what result will be produced?
a) area for all triangles will be computed to be too high
b) area for all triangles will be computed to be too low

c) area will only be incorrect for a triangle objects with width = 1
d) infinite recursion will occur for triangle objects with width >= 2
Answer: d

32) Consider the `getArea` method from the textbook shown below:

```
public int getArea()
{
   if (width <= 0) { return 0; } // line #1
   if (width == 1) { return 1; } // line #2
   Triangle smallerTriangle = new Triangle(width – 1); // line #3
   int smallerArea = smallerTriangle.getArea();  // line #4
   return smallerArea + width;  // line #5
}
```

If line #1 was eliminated from the method, what would be the result when calling `getArea`?
a) A negative or zero width would cause problems.
b) Nothing - the method would still work correctly.
c) We would lose our only recursive case.
d) A positive width would reduce the correct area by 1.
Answer: a

33) How many recursive calls to the `fib` method shown below would be made from an original call to `fib(4)`? (Do not count the original call)

```
public int fib(int n)
{  // assumes n >= 0
   if (n <= 1)
   {
      return n
   }
   else
   {
      return (fib(n - 1) + fib(n - 2));
   }
}
```

a) 1	b) 2
c) 4	d) 8

Answer: d

34) Would switching the special case order affect the return value of the following method?

```
public int mystery(int n, int m)
{
   if (n == 0)   // special case #1
   {
      return 0;
   }
   if (n == 1)   // special case #2
   {
      return m;
   }
   return m + mystery(n - 1, m);
}
```

a) No
b) Yes
c) It is impossible to tell.
d) An exception will be thrown.
Answer: a

35) Which of the following options could be used as a terminating condition for a recursive method that finds the middle character of a `String` with any number of characters?
I the length of the `String` is 1
II first and last `String` characters match
III the `String` is not empty

a) I
b) II
c) I, II and III
d) I and III
Answer: a

36) Consider the problem of arranging matchsticks so as to form a row of rectangles, as shown below.

```
 _ _ _ _ _ _
|_ _|_ _|_ _|
```

Complete the recursive method below, which is designed to return the number of matchsticks needed to form n rectangles.

```
public static int matchsticks(int rectangles)
{
   if (rectangles == 1)        // 1 square can be formed with 6 matchsticks
   {
      return 6;
   }
   else
   {
      return ___________________________
   }
}
```

a) `6 + matchsticks(rectangles – 1);`
b) `5 + matchsticks(rectangles – 1);`
c) `6 * rectangles;`
d) `matchsticks(rectangles + 6);`
Answer: b

37) Consider the method below, which implements the exponentiation operation recursively. Select the statement that should be used to complete the method, so that it handles the special case correctly.

```
public static double power(int base, int exponent)
{
   if (exponent == 0)
   {
      _______________
   }
   else
   {
      reurn base * power(base, exponent – 1);
   }
}
```

a) `return 1;`
b) `return base;`
c) `return 0;`
d) `return 1 * power(base, exponent – 1);`
Answer: a

38) Consider the method below, which displays the characters from a String in reverse order. Each character appears on a separate line. Select the statement that should be used to complete the method, so that it performs a recursive method call correctly.

```
public static void printReverse(String word)
{
   if (word.length() > 0)
   {
      ___________________________
      System.out.println(word.charAt(0));
   }
}
```

a) `printReverse(word);`
b) `printReverse(new String(word.charAt(1)));`
c) `printReverse(word.substring(1));`
d) `printReverse(word.length() - 1);`
Answer: c

39) Consider the method below, which prints the digits of an arbitrary integer in reverse order, one digit per line. The method should print the last digit first. Then, it should recursively print the integer obtained by removing the last digit. Select the statements that should be used to complete the method.

```
public static void printReverse(int value)
{
   if (value > 0)
   {
      _____________________     // print last digit
      _____________________     // recursive call to print value without last digit
   }
}
```

a)
```
System.out.println(value / 10);
printReverse(value / 10);
```
b)
```
System.out.println(value % 10);
printReverse(value / 10);
```
c)
```
System.out.println(value / 10);
printReverse(value % 10);
```
d)
```
System.out.println(value % 10);
printReverse(value % 10);
```
Answer: b

40) Consider the method `powerOfTwo` shown below:
```
public boolean powerOfTwo(int n)
{
   if (n == 1)    // line #1
   {
      return true;
   }
   else if (n % 2 == 1) // line #2
   {
      return false;
   }
   else
   {
      return powerOfTwo(n / 2);  // line #3
   }
}
```
What is the best interpretation of line #1?
a) One is a power of two
b) One is not a power of two
c) Any multiple of one is a power of two
d) 1 is an invalid choice for `n`
Answer: a

41) Consider the method `powerOfTwo` shown below:

```
public boolean powerOfTwo(int n)
{
   if (n == 1)    // line #1
   {
      return true;
   }
   else if (n % 2 == 1) // line #2
   {
      return false;
   }
   else
   {
      return powerOfTwo(n / 2);  // line #3
   }
}
```
How many recursive calls are made from the original call `powerOfTwo(63)` (not including the original call)?

a) 6 b) 4
c) 1 d) 0
Answer: d
42) Consider the method `powerOfTwo` shown below:

```
public boolean powerOfTwo(int n)
{
   if (n == 1)    // line #1
   {
      return true;
   }
   else if (n % 2 == 1) // line #2
   {
      return false;
   }
   else
   {
      return powerOfTwo(n / 2);   // line #3
   }
}
```
How many recursive calls are made from the original call of `powerOfTwo(64)` (not including the original call)?

a) 8
b) 6
c) 4
d) 2
Answer: b

43) Complete the code for the `myFactorial` recursive method shown below, which is intended to compute the factorial of the value passed to the method:

```
public int myFactorial(int anInteger)
{
   if (anInteger == 1)
   {
      return 1;
   }
   else
   {
      ______________________
   }
}
```

a) `return(anInteger * (myFactorial(anInteger)));`
b) `return ((anInteger - 1) * (myFactorial(anInteger)));`
c) `return(anInteger * (myFactorial(anInteger - 1)));`
d) `return((anInteger - 1)*(myFactorial(anInteger - 1)));`
Answer: c

44) Complete the code for the `myFactorial` recursive method shown below, which is intended to compute the factorial of the value passed to the method:

```
public int myFactorial(int anInteger)
{
   ______________________________
   {
      return 1;
   }
   else
   {
      return(anInteger * myFactorial(anInteger - 1));
   }
}
```

a) `if (anInteger == 1)`
b) `if ((anInteger - 1) == 1)`
c) `if (anInteger * (anInteger - 1) == 1)`
d) `if (myFactorial(anInteger) == 1)`
Answer: a

45) Complete the code for the `myFactorial` recursive method shown below, which is intended to compute the factorial of the value passed to the method:

```
public int myFactorial(int anInteger)
{
   if (anInteger == 1)
   {
      ______________________
   }
   else
   {
        return (anInteger * myFactorial(anInteger - 1));
    }
}
```
a) `return 0;`
b) `return -anInteger;`
c) `return 1;`
d) `return myFactorial(anInteger);`
Answer: c

46) Complete the code for the recursive method shown below, which is intended to compute the sum of the first *n* integers:
```
public int s(int n)
{
    if (n == 1)
    {
        return 1;
    }
    else
    {
        _________________
    }
}
```

a) `return n + (n - 1);`
b) `return s(n) + n - 1;`
c) `return n + s(n - 1);`
d) `return n + s(n + 1);`
Answer: c

47) Complete the code for the `calcPower` recursive method shown below, which is intended to raise the base number passed into the method to the exponent power passed into the method:
```
public static int calcPower(int baseNum, int exponent)
{
    int answer = 0;
    ________________________
    {
        answer = 1;
    }
    else
    {
        answer = baseNum * calcPower (baseNum, exponent - 1);
    }
    return answer;
}
```
a) `if (exponent == 0)`
b) `if (exponent == 1)`
c) `if (exponent == -1)`
d) `if (exponent != 1)`
Answer: a

48) Complete the code for the `calcPower` recursive method shown below, which is intended to raise the base number passed into the method to the exponent power passed into the method:
```
public static int calcPower(int baseNum, int exponent)
{
    int answer = 0;
    if (exponent == 0)
    {
        _____________________
    }
    else
    {
        answer = baseNum * calcPower (baseNum, exponent - 1);
    }
    return answer;
}
```
a) `answer = 0;`
b) `answer = 1;`
c) `answer = -1;`
d) `answer = calcPower(1);`
Answer: b

49) Complete the code for the `calcPower` recursive method shown below, which is intended to raise the base number passed into the method to the exponent power passed into the method:

```
public static int calcPower(int baseNum, int exponent)
{
   int answer = 0;
   if (exponent == 0)
   {
      answer = 1;
   }
   else
   {
      _______________________________________
   }
   return answer;
}
```

a) `answer = baseNum * calcPower (baseNum -1, exponent);`
b) `answer = baseNum * calcPower (baseNum, exponent - 1);`
c) `answer = baseNum * calcPower (baseNum, exponent);`
d) `answer = baseNum * calcPower (baseNum -1, exponent - 1);`
Answer: b

50) Given the following class code:

```
public class RecurseSample
{
   public static void main(String[] args)
   {
      recurse(3);
   }
   public static int recurse(int n)
   {
      int total = 0;
      if (n == 0)
      {
         return 0;}
      else
      {
         total = 3 + recurse(n - 1);
      }
      System.out.println(total);
      return total;
   }
}
```

What values will be printed?
a) 1, 3, and 6
b) 1, 3, 6, and 9
c) 3, 6, and 9
d) 3, 6, 9, and 12
Answer: c
51) Given the following class code:

```
public class RecurseSample
{
   public static void main(String[] args)
   {
      System.out.println(recurse(3));
   }

   public static int recurse(int n)
   {
      int total = 0;
      if (n == 0)
      {
         return 0;
      }
      else
      {
         total = 3 + recurse(n - 1);
      }
      return total;
   }
}
```
What values will be printed when this code is executed?
a) 6
b) 9
c) 3, 6, and 9
d) 1, 3, 6, and 9
Answer: b

52) Given the following class code:
```
public class RecurseMore
{
   public static void main(String[] args)
   {
      recurse(4);
   }

   public static int recurse(int n)
   {
      int total = 0;
      if (n == 0)
      {
         return 0;
      }
      else
      {
         total = 4 + recurse(n - 2);
      }
      System.out.println(total);
      return total;
   }
}
```
What values will be printed when this code is executed?
a) 0, 4, and 8
b) 4 and 8
c) 4
d) 8
Answer: b

53) Given the following code snippet:
```
public static int newCalc(int n)
{
   if (n < 0)
   {
      return -1;
   }
   else if (n < 10)
   {
      return n;
   }
   else
   {
      return (n % 10) + newCalc(n / 10);
   }
}
```
What value will be returned when this code is executed with a call to `newCalc(15)`?
a) 2
b) 2.5
c) 6
d) 6.5
Answer: c

54) Given the following code snippet:

```
public static int newCalc(int n)
{
   if (n < 0)
   {
      return -1;
   }
   else if (n < 10)
   {
      return n;
   }
   else
   {
      return (n % 10) + newCalc(n / 10);
   }
}
```
What value will be returned when this code is executed with a call to `newCalc(5)`?

a) 2 b) 2.5
c) 5 d) 5.5
Answer: c

55) Given the following code snippet:
```
public static int newCalc(int n)
{
   if (n < 0)
   {
      return -1;
   }
   else if (n < 10)
   {
      return n;
   }
   else
   {
      return (1 + newCalc(n / 10));
   }
}
```
What value will be returned when this code is executed with a call to `newCalc(5)`?

a) 1
b) 1.5
c) 5
d) 5.5
Answer: c

56) Given the following code snippet:
```
public static int newCalc(int n)
{
   if (n < 0)
   {
      return -1;
   }
   else if (n < 10)
   {
      return n;
   }
   else
   {
      return (1 + newCalc(n / 10));
   }
}
```
What value will be returned when this code is executed with a call to `newCalc(15)`?

a) 2 b) 2.5
c) 5 d) 5.5
Answer: a

57) Given the following class code:
```
public class RecurseMore
{
   private static int total;
   public static void main(String[] args)
   {
      System.out.println(recurse(4));
   }

   public static int recurse(int n)
   {
      int total = 0;
      if (n == 0)
```

```
        {
           return 0;
        }
        else
        {
           total = 4 + recurse(n - 2);
        }
           return total;
     }
  }
```

What values will be printed when this code is executed?

a) 0, 4, and 8 b) 4 and 8

c) 4 d) 8

Answer: d

58) Complete the following code snippet, which is intended to be a recursive method that will find the smallest value in an array of double values from `index` to the end of the array:

```
public static double minVal(double[] elements, int index)
{
   if (index == elements.length - 1)
   {
      return elements[index];
   }
   double val = __________________;
   if (elements[index] < val)
   {
      return elements[index];
   }
   else
   {
      return val;
   }
}
```

a) `minVal(elements, index - 1)` b) `minVal(index – 1)`

c) `minVal(elements, index + 1)` d) `minVal(index + 1)`

Answer: c

59) Complete the following code snippet, which is intended to be a recursive method that will find the smallest value in an array of double values from `index` to the end of the array:

```
public static double minVal(double[] elements, int index)
{
   if (index == elements.length - 1)
   {
      __________________
   }
   double val = minVal(elements, index + 1);
   if (elements[index] < val)
   {
      return elements[index];
   }
   else
   {
      return val;
   }
}
```

a) `return elements[index];`

b) `return 1;`

c) `return 0;`

d) `return elements[0];`

Answer: a

60) Complete the following code snippet, which is intended to be a recursive method that will find the sum of all elements in an array of double values from `index` to the end of the array:

```
// return the sum of all elements in arr[]
public static double findSum(double arr[], int index)
{
   if (index == 0)
   {
      return arr[index];
```

```
    }
    else
    {
        _______________________
    }
}
```
Assume that this method would be called using an existing array named `myArray` as follows:

```
findSum(myArray, myArray.length - 1);
```

a) `return (findSum(arr, index + 1));`
b) `return (findSum(arr, index - 1));`
c) `return (arr[index] + findSum(arr, index + 1));`
d) `return (arr[index] + findSum(arr, index - 1));`
Answer: d

61) Complete the following code snippet, which is intended to be a recursive method that will find the sum of all elements in an array of double values from `index` to the end of the array:
```
// return the sum of all elements in arr[]
public static double findSum(double arr[], int index)
{
    if (index == 0)
    {
        _______________________
    }
    else
    {
        return (arr[index] + findSum(arr, index - 1));
    }
}
```
Assume that this method would be called using an existing array named `myArray` as follows:
```
findSum(myArray,myArray.length - 1);
```
a) `return arr[index];`
b) `return arr[index + 1];`
c) `return arr[1];`
d) `return arr[index - 1];`
Answer: a

62) Complete the following code snippet, which is intended to be a recursive method that reverses a `String` value:
```
public static String reverseIt(String s)
{
    if (s.length() <= 1)
    {
        __________
    }
    else
    {
        return reverseIt(s.substring(1)) + s.charAt(0);
    }
}
```
a) `return s;`
b) `return 0;`
c) `return s.charAt(0);`
d) `return s.substring(1);`
Answer: a

63) Complete the following code snippet, which is intended to be a recursive method that reverses a `String` value:
```
public static String reverseIt(String s)
{
    if (s.length() <= 1)
    {
        return s;
    }
    else
    {
        __________________________
    }
}
```
a) `return reverseIt(s.substring(0)) + s.charAt(1);`

b) `return reverseIt(s.substring(0)) + s.charAt(0);`
c) `return reverseIt(s.substring(1)) + s.charAt(1);`
d) `return reverseIt(s.substring(1)) + s.charAt(0);`
Answer: d

64) Consider the code for the recursive method `printSum` shown in this code snippet, which is intended to return the sum of digits from 1 to `n`:

```
public static int printSum(int n)
{
   if (n <= 0)    // line #1
   {
      return 0;   // line #
   }
   else
   {
      return (n + printSum(n));  //line #3
   }
}
```

Which of the following statements is correct?
a) line #1 is incorrect, and should be changed to `if (n <= 1)`
b) line #3 is incorrect, and should be changed to `return (printSum (n - 1));`
c) line #3 is incorrect, and should be changed to `return (n + printSum (n + 1));`
d) line #3 is incorrect, and should be changed to `return (n + printSum (n - 1));`
Answer: d

65) A palindrome is a word or phrase that reads the same forward or backward. Consider the methods `palindrome` and `isPal` shown below:

```
public boolean palindrome(String string)
{
   return isPal(string, 0, string.length() - 1);
}

private boolean isPal(String string, int left, int right)
{
   if (left >= right)
   {
      return true;
   }
   else if (string.charAt(left) == string.charAt(right))
   {
      return isPal(string, left + 1, right - 1);
   }
   else
   {
      return false;
   }
}
```

The method `palindrome` as shown here would be considered to be a _____ method.
a) recursive   b) terminating
c) helper   d) static
Answer: c

66) A palindrome is a word or phrase that reads the same forward or backward. Consider the following code snippet:

```
public boolean palindrome(String string)
{
   return isPal(string, 0, string.length() - 1);
}

private boolean isPal(String string, int left, int right)
{
   if (left >= right)
   {
      return true;
   }
   else if (string.charAt(left) == string.charAt(right))
   {
      return isPal(string, left + 1, right - 1);
   }
   else
```

```
    {
       return false;
    }
}
```
What is the purpose of the `palindrome` method?
a) Return the palindrome to the calling method.
b) Provide the string, along with its first and last indexes to the recursive `isPal` method.
c) Send the recursive `isPal` method its terminating condition.
d) Recursively call itself.
Answer: b

67) A palindrome is a word or phrase spelled which reads the same forward or backward. Consider the following code snippet:
```
public boolean palindrome(String string)
{
   return isPal(string, 0, string.length() - 1);
}

private boolean isPal(String string, int left, int right)
{
   if (left >= right)
   {
      return true;
   }
   else if (string.charAt(left) == string.charAt(right))
   {
      return isPal(string, left + 1, right - 1);
   }
   else
   {
      return false;
   }
}
```
What does the method `palindrome` return?
a) true
b) false
c) a palindrome not found exception
d) the Boolean value returned from the `isPal` method
Answer: d

68) A palindrome is a word or phrase that reads the same forward or backward. Consider the following code snippet:
```
public boolean palindrome(String string)
{
   return isPal(string, 0, string.length() - 1);
}

private boolean isPal(String string, int left, int right)
{
   if (left >= right)
   {
      return true;
   }
   else if (string.charAt(left) == string.charAt(right))
   {
      return isPal(string, left + 1, right - 1);
   }
   else
   {
      return false;
   }
}
```
What does the condition `left >= right` refer to?

a) An empty or one-character string is considered a palindrome.
b) The string is not a palindrome.
c) It cannot be determined if the string is a palindrome.
d) You have reached the middle of the string.
Answer: a

69) What is the purpose of a recursive helper method?
a) Shield the user of the recursive method from the recursive details.
b) Speed up the execution.
c) Eliminate the recursion.
d) Add another base case.
Answer: a

70) Consider the recursive `square` method shown below. It takes a non-negative `int` argument. Then it recursively adds the number `n` to itself `n` times to produce the square of `n`. Complete the correct code for the `square` helper method.

```
public int square(int n)
{
   _______________________;
}

public int square(int c, int n)
{
   if (c == 1)
   {
      return n;
   }
   else
   {
      return n + square(c - 1, n);
   }
}
```

a) `return square(n, n)`
b) `return square(n)`
c) `return square(n - 1, n)`
d) `return square(n, n - 1)`
Answer: a

71) Consider the recursive `square` method shown below that takes a non-negative `int` argument:

```
public int square(int n)
{
   return square(n, n);
}

public int square(int c, int n)
{
   if (c == 1)
   {
      return n;
   }
   else
   {
      return n + square(c - 1, n);
   }
}
```

Assume that the last `return` statement is changed to this:

```
return square(c - 1, n);
```

What would a call to `square(7)` return?

a) 7   b) 13
c) 14   d) 49
Answer: a

72) Consider the recursive `square` method shown below that takes a non-negative `int` argument.

```
public int square(int n)
{
   return square(n, n);
}

public int square(int c, int n)
{
   if (c == 1)
   {
      return n;
   }
   else
   {
      return n + square(c - 1, n);
   }
}
```
Assume that the last `return` statement is changed to this:
`return n * square(c - 1, n);`
What would a call to `square(4)` return?
a) 4
b) 16
c) 64
d) 256
Answer: d

73) Consider the helper method `reversePrint`, which uses recursion to display in reverse the elements in a section of an array limited by the `firstIndex` and `lastIndex` arguments. What statement should be used to complete the recursive method?
```
public static void reversePrint(int[] array, int firstIndex, int lastIndex)
{
   if (firstIndex < lastIndex)
   {
      ________________________________________
   }
   System.out.println(array[firstIndex]);
}
public static void main(String[] args)
{
   int [] numbers = { 4, 7, 1, 0, 2, 7 };
   reversePrint(numbers, 0, numbers.length - 1);
}
```
a) `reversePrint(array, firstIndex, lastIndex + 1);`
b) `reversePrint(array, firstIndex, lastIndex - 1);`
c) `reversePrint(array, firstIndex + 1, lastIndex - 1);`
d) `reversePrint(array, firstIndex + 1, lastIndex);`
Answer: d

74) Assume that recursive method search returns `true` if argument `value` is one of the elements in the section of the array limited by the `firstIndex` and `lastIndex` arguments. What statement can be used in `main` to determine if the value `7` is one of the elements in array `values`?
```
public static boolean search(int value, int[] array, int firstIndex, int lastIndex)
{
   if (firstIndex <= lastIndex)
   {
      if (array[firstIndex] == value)
      {
         return true;
      }
      else
      {
         return search(value, array, firstIndex + 1, lastIndex);
      }
   }
   return false;
}

public static void main(String[] args)
{
   int [] values = { 4, 7, 1, 0, 2, 7 };
   if ( _________________________________ )
   {
      System.out.println("7 is in the array");
   }
}
```
a) `search(7, values, 0, values.length)`
b) `search(7, values, 1, values.length)`
c) `search(7, values, 0, values.length - 1)`
d) `search(7, values, 1, values.length - 1)`
Answer: c

75) Consider the problem of displaying a pattern of asterisks which form a triangle of height h, as shown below for h = 4:

```
*
**
***
****
***
**
*
```

The problem can be solved with the recursive helper method `shape` below. The method takes two parameters: `low` and `high`. It first prints a row of asterisks corresponding to parameter `low`. If `high` is higher than `low`, it recursively generates a shape in which `low` is incremented by one, and then prints another row of `low` asterisks. Select a statement to complete method `triangle`, so that the helper method `shape` is invoked with the correct arguments in order to display a triangle with parameter `height`.

```
public static void shape(int low, int high)
{
   if (high >= low)
   {
      asterisksRow(low);
      if (high > low)
      {
         shape(low + 1, high);
         asterisksRow(low);
      }
   }
}
public static void asterisksRow(int n)  // Method to display a row of n stars
{
   for (int j = 1; j < n; ++j)
   {
      System.out.print("*");
   }
   System.out.println("*");
}
public static void triangle(int height)
{
   if (height > 1)
   {
      _______________________
   }}
```

a) `shape(0, height);`
b) `shape(0, height - 1);`
c) `shape(1, height);`
d) `shape(0, height - 1);`
Answer: c

76) Why does the best recursive method usually run slightly slower than its iterative counterpart?
a) Testing the terminating condition takes longer.
b) Each recursive method call takes processor time.
c) Multiple recursive cases must be considered.
d) Checking multiple terminating conditions take more processor time.
Answer: b

77) Consider the `fib` method from the textbook shown below:

```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n – 1) + fib(n – 2);
   }}
```

Computing the 7th fibonacci number, `fib(7)`, recursively computes `fib(6)`, `fib(5)`, and `fib(4)` ___ times respectively.
a) 1, 2, and 3
b) 6, 5, and 4
c) 4, 5, and 6
d) 3, 2, and 1
Answer: a

78) Consider the `fib` method from the textbook shown below.

```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n – 1) + fib(n – 2);
   }
}
```

Calling `fib(3)` will trigger ___ recursive call(s) and execute the terminating condition ___ time(s), respectively.

a) 1, 1 b) 2, 2
c) 1, 2 d) 2, 1

Answer: b

79) Consider the `fib` method from the textbook shown below:

```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;  // line #1
   }
   else
   {
      return fib(n – 1) + fib(n – 2);  // line #2
   }
}
```

Assume line #1 is changed to this:

```
if (n <= 2) { return 2; }
```

How will this change affect the result of calling `fib(7)`?

a) It will add 2 to the return from `fib(7)`
b) It will add 4 to the return from `fib(7)`
c) It will add 8 to the return from `fib(7)`
d) It will double the return from `fib(7)`

Answer: d

80) Consider the `fib` method from the textbook shown below:

```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;  // line #1
   }
   else
   {
      return fib(n – 1) + fib(n – 2); // line #2
   }
}
```

Assume line #2 is changed to this:

```
else { return 2 * fib(n – 1) + 2 * fib(n – 2); }
```

What effect will this change have?

a) This will return 5 from `fib(4)`
b) This will return 8 from `fib(4)`
c) This will return 10 from `fib(4)`
d) This will cause an exception on a call `fib(4)`

Answer: c

81) Consider the `fib` method from the textbook shown below:

```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;  // line #1
   }
   else
```

```
    {
       return fib(n - 1) + fib(n - 2);  // line #2
    }
}
```

Assume line #1 is changed to this:
```
if (n <= 2) { return n; }
```

What effect will this change have?
a) This will return 21 from `fib(6)`
b) This will return 13 from `fib(6)`
c) This will return 8 from `fib(6)`
d) This will return 6 from `fib(6)`
Answer: b

82) Consider the recursive version of the `fib` method from the textbook shown below:
```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n - 1) + fib(n - 2);
   }
}
```
How many recursive calls to `fib(2)` will be made from the original call of `fib(6)`?
a) 2 b) 3
c) 4 d) 5
Answer: d

83) Consider the recursive version of the `fib` method from the textbook shown below:
```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n - 1) + fib(n - 2);
   }
}
```
How many total recursive calls (not counting the original call) to `fib` will be made from the original call of `fib(6)`?
a) 6 b) 12 c) 14 d) 20
Answer: c

84) Consider the recursive version of the `fib` method from the textbook shown below:
```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n - 1) + fib(n - 2);
   }
}
```
How many total recursive calls (not counting the original call) to `fib` will be made from the original call of `fib(7)`?
a) 12 b) 24 c) 30 d) 32
Answer: b

85) Consider the recursive version of the `fib` method from the textbook shown below:
```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n - 1) + fib(n - 2);
   }}
```
How many more recursive calls to `fib` will be made from the original call of `fib(7)` than from the original call of `fib(6)` (not counting the original calls)?

a) 1 b) 2
c) 5 d) 10
Answer: d

86) The method below implements the exponentiation operation recursively by taking advantage of the fact that, if the exponent n is even, then $x^n = (x^{n/2})^2$. Select the expression that should be used to complete the method so that it computes the result correctly.
```
public static double power(double base, double exponent)
{
   if (exponent % 2 != 0)   // if exponent is odd
   {
      return base * power(base, exponent - 1);
   }
   else if (exponent > 0)
   {
      double temp = _________________________ ;
      return temp * temp;
   }
   return base; }
```
a) `power(base, exponent / 2)`
b) `power(base, exponent / 2) * power(base, exponent / 2)`
c) `power(base, exponent / 2) + power(base, exponent / 2)`
d) `power(base, exponent) / 2`
Answer: a

87) Consider the iterative version of the `fib` method from the textbook shown below:
```
public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   long fold = 1;
   long fold2 = 1;
   long fnew = 1;
   for (int i = 3; i <= n; i++)
   {
      fnew = fold + fold2;
      fold2 = fold;
      fold = fnew;
   }
   return fnew;
}
```
How many iterations of the `for` loop will there be for the call `fib(6)`?

a) 6 b) 5
c) 4 d) 3
Answer: c

88) Suppose we wrote a new version of `fib` called `newFib`. What does the call `newFib(6)` return?
```
public static long newFib(int n)
{
   if (n <= 3)
   {
      return 1;
   }
   else
   {
      return newFib(n - 1) + newFib(n - 2) + newFib(n - 3);
   }}
```
a) 3 b) 5
c) 7 d) 9
Answer: d

89) Suppose we wrote a new version of method `fib`, called `newFib`. Compare `newFib` to the original `fib` shown below:

```
public static long newFib(int n)
{
   if (n <= 3)
   {
      return 1;
   }
   else
   {
      return newFib(n - 1) + newFib(n - 2) + newFib(n - 3);
   }
}

public static long fib(int n)
{
   if (n <= 2)
   {
      return 1;
   }
   else
   {
      return fib(n - 1) + fib(n - 2);
   }
}
```

For which values of the integer `n` does `newFib(n)` always returns a value greater than `fib(n)`?
a) any positive `n`
b) any value of `n`
c) `n >= 3`
d) `n > 3`
Answer: d

90) Which statement(s) about recursion are true?

I Recursion is faster than iteration
II Recursion is often easier to understand than iteration
III Recursive design has an economy of thought
a) I
b) II
c) II and III
d) I and III
Answer: c

91) In recursion, the recursive call is analogous to a loop ____.
a) call
b) iteration
c) termination
d) return
Answer: b

92) In recursion, the non-recursive case is analogous to a loop ____.
a) call b) iteration
c) termination d) condition
Answer: c

93) In recursion, the terminating condition is analogous to a loop _________.
a) call
b) iteration
c) termination condition
d) initialization condition
Answer: c

94) Complete the following code snippet, which is intended to print out all permutations of the string `generate` by using a permutation generator object.

```
public class PermutationGeneratorTester
{
   public static void main(String[] args)
   {
      PermutationGenerator generator
            = new PermutationGenerator("generate");
      ArrayList<String> permutations
```

```
            = generator.getPermutations();
      for (String s : permutations)
      {
         ____________________
      }
   }}
```
a) `generator.print();`
b) `generator.setPermutations();`
c) `generator.calculate();`
d) `System.out.println(s);`
Answer: d

95) Which of the following strings is a palindrome?
a) "Test" b) "B"
c) "canal" d) "salami"
Answer: b

96) Which of the following statements is correct?
a) The empty string is a palindrome.
b) Strings of length 0 or 1 are not palindromes.
c) The string `"eat"` is a palindrome.
d) The string `"Adam"` is a palindrome.
Answer: a

97) The string `"eat"` has ____ permutations.
a) 2 b) 4
c) 6 d) 8
Answer: c

98) A unique permutation is one that is different from any other generated permutation. How many unique permutations does the string `"bee"` have?
a) 2 b) 3
c) 4 d) 5
Answer: b

99) A unique permutation is one that is different from any other generated permutation. How many unique permutations does the string `"aaa"` have?
a) 0 b) 1
c) 2 d) 3
Answer: b

100) Which of the following statements about palindromes is correct?
a) The empty string is not a palindrome.
b) The string `"I"` is not a palindrome.
c) The string `"rascal"` is a palindrome.
d) All strings of length 0 or 1 are palindromes.
Answer: d

101) Consider the following change to the `PermutationGenerator` class from the textbook. Instead of adding the removed character to the front of the each permutation of the simpler word, we will add it to the end.
```
// Add the removed character to the end of
// each permutation of the simpler word
for (String s : shorterWordPermutations)
{
   result.add(s + word.charAt(i));
}
```
Consider the list produced by the modified `PermutationGenerator`. Which best describes this list?
a) It contains all permutations.
b) It contains reversed permutations.
c) It is an incomplete list of permutations.
d) It contains strings that are not permutations.
Answer: a

102) Which of the following statements about recursion is correct?
a) It is not necessary to have a special terminating case in all recursions.
b) It is not necessary to simplify the argument in the recursive call.
c) A recursive solution will always run faster than an equivalent iterative solution.
d) In many cases, a recursive solution will be easier to understand and to implement than an iterative solution.
Answer: d

103) The method below generates all substrings of a String passed as argument. Assuming that the string contains no duplicate characters, select the statement to complete the method so that it prints all substrings correctly.

```
public static void printSubstrings(String word)
{
   if (word.length() > 0)
   {
      for (int j = 1; j <= word.length(); j++)     // print substrings that begin
      {                                            // with the first character
         System.out.println(word.substring(0, j));
      }
      ___________________________________         // print substrings without
   }                                              // the first character
}
```

a) `printSubstrings(word.substring(0));`
b) `printSubstrings(word.substring(1));`
c) `printSubstrings(word.substring(word.length()));`
d) `printSubstrings(word.substring(word.length() - 1));`
Answer: b

104) Consider the mutually recursive methods below. Select the method call that could be used to generate the output sequence:
`A5 B4 A3 B2 A1`

```
public static void methodA(int value)
{
   if (value > 0)
   {
      System.out.print(" A" + value);
      methodB(value - 1);
   }
}
public static void methodB(int value)
{
   if (value > 0)
   {
      System.out.print(" B" + value);
      methodA(value - 1);
   }
}
```

a) `methodA(5);` b) `methodB(5);`
c) `methodA(4);` d) `methodB(4);`
Answer: a

105) Recursion will take place if any of the following happen:
I method A calls method B, which calls method C
II method A calls method B, which calls method A
III method A calls method A
a) I b) I and II
c) II d) II and III
Answer: d

106) Recursion does NOT take place if any of the following happen:
I method A calls method B, which calls method C, which calls method B
II method A calls method B, which calls method A
III method A calls method B, B returns, and A calls B again
a) I
b) I and II
c) II
d) III
Answer: d

107) Consider the following code snippet:

```
public static boolean isEven(int n)
{
   if (n % 2 == 0)
   {
      return true;
   }
   else
   {
      return (isOdd(n));
   }
}
```

```
public static boolean isOdd(int n)
{
   if (n % 2 == 1)
   {
      return true;
   }
   else
   {
      return (isEven(n));
   }
}
```

For any given value of n, what is the maximum number of function calls that could occur?
a) 0
b) 1
c) 2
d) cannot be determined
Answer: c

108) Complete the following code snippet, which is intended to determine if a value is even or odd using mutual recursion:

```
public static boolean isEven(int n)
{
   if (n == 0)
   {
      return true;
   }
   else
   {
      return isOdd(Math.abs(n) - 1);
   }
}
public static boolean isOdd(int n)
{
   if (n == 0)
   {
      _________
   }
   else
   {
      return isEven(Math.abs(n) - 1);
   }
}
```

a) `return true;`
b) `return false;`
c) `return isOdd(Math.abs(n)-1);`
d) `return isOdd(Math.abs(n));`
Answer: b

109) Backtracking _____.

a) starts from the end of the program and works backward to the beginning.
b) starts with a partial solution and builds it up to get closer to the goal.
c) never abandons a partial solution.
d) explores only one path toward a solution
Answer: b

110) ____ is a problem-solving technique that examines partial solutions, abandons unsuitable ones, and returns to consider other candidate solutions.

a) Debugging
b) Traceback
c) Backtracking
d) Recursion
Answer: c

Chapter14: Sorting and Searching

1) What type of algorithm places elements in order?

a) sorting
b) searching
c) insertion
d) deletion
Answer: a

2) In each iteration, selection sort places which element in the correct location?

a) The smallest in the array.
b) The smallest element not yet placed in prior iterations.
c) The largest element in the array.
d) A random element.
Answer: b

3) After 5 iterations of selection sort working on an array of 10 elements, what must hold true?

a) 5 more iterations are always necessary to complete the sort
b) 4 more iterations are always necessary to complete the sort
c) 5 more iterations may be needed to complete the sort
d) 4 more iterations may be needed to complete the sort
Answer: b

4) After one iteration of selection sort working on an array of 10 elements, what must hold true?

a) The array cannot be sorted.
b) One element must be correctly placed.
c) At least two elements are correctly placed.
d) The largest element is correctly placed.
Answer: b

5) Consider the `sort` method shown below for selection sort:

```
public static void sort(int[] a)
{
   for (int i = 0; i < a.length – 1; i++)
   {
      int minPos = minimumPosition(i);
      swap(minPos, i);
   }
}
```
Suppose we modify the loop condition to read `i < a.length`. What would be the result?

a) An exception would occur.
b) The sort would work, but run one more iteration.
c) The sort would work but with one less iteration.
d) The sort would work exactly the same as before the code modification.
Answer: b

6) Consider the `sort` method shown below for selection sort:
```
public static void sort (int[] a)
{
   for (int i = 0; i < a.length – 1; i++)
   {
      int minPos = minimumPosition(i);
      swap(minPos, i);
   }
}
```
Suppose we modify the call to the `swap` method call to read `swap(i, minPos)`. What would be the result?
a) An exception would occur.
b) The sort would produce incorrect results.
c) The sort would work, but sort backwards.
d) The sort would produce correct results.
Answer: d

7) Consider the `sort` method for selection sort shown below:

```
public static void sort (int[] a)
{
   for (int i = 0; i < a.length – 1; i++)
   {
      int minPos = minimumPosition(i);
      swap(minPos, i);
   }
}
```

Suppose we modify the loop control to read `int i = 1; i < a.length – 1; i++`. What would be the result?

a) An exception would occur
b) The sort would not consider the last array element.
c) The sort would not consider the first array element.
d) The sort would still work correctly.
Answer: c

8) Consider the `minimumPosition` method from the `SelectionSorter` class. Complete the code to write a `maximumPosition` method that returns the index of the largest element in the range from index `from` to the end of the array.

```
private static int minimumPosition(int[] a, int from)
{
   int minPos = from;
   for (int i = from + 1; i < a.length; i++)
   {
      if (a[i] < a[minPos]) { minPos = i; }
   }
     return minPos;
}

private static int maximumPosition(int[] a, int from)
{
   int maxPos = from;
   for (int i = from + 1; i < a.length; i++)
   {
      ________________
   }
   return maxPos;
}
```

a) `if(a[i] > a[maxPos]) { maxPos = i; }`
b) `if(a[i] == a[maxPos]) { maxPos = i; }`
c) `if(a[i] < a[maxPos]) { maxPos = i; }`
d) `if(a[i] <= a[maxPos]) { maxPos = i; }`
Answer: a

9) Consider the `swap` method shown below from the `SelectionSorter` class. If we modified it as shown in the `swap2` method shown below, what would be the effect on the `sort` method?

```
private static void swap(int[] a, int i, int j)
{
   int temp = a[i];
   a[i] = a[j];
   a[j] = temp;
}
private static void swap2(int[] a, int i, int j)
{
   a[i] = a[j];
   a[j] = a[i];
}
```

a) There would be no effect.
b) Some array elements would be overwritten.
c) It would sort the array in reverse order.
d) It would still be correct, but run a little faster.
Answer: b

10) Suppose you wanted to test your sort on an array filled with different elements each time the code is run. What is an efficient technique for creating an array of 1,000 elements for each run?

a) Run the program many times, entering different values for the array elements.
b) Make a file with many sets of values and loop through them, sorting each one.
c) Use the `Random` class to generate array elements, sorting each set in a loop.
d) Create many different arrays with different elements in the program code and sort each array.
Answer: c

11) After 9 iterations of selection sort working on an array of 10 elements, what must hold true?
a) The largest element is correctly placed by default.
b) One more iteration is needed to complete the sort.
c) The smallest element is incorrectly placed.
d) The largest element is incorrectly placed.
Answer: a

12) Which selection sort iteration guarantees the array is sorted for a 10-element array?
a) 9th iteration
b) 10th iteration
c) 1st iteration
d) impossible to tell
Answer: a

13) The `largestPosition` method below returns the index of the largest element in the tail range of an array of integers. Select the expression that would be needed to complete the `selectionSort` method below, so that it sorts the elements in descending order.

```
/**
   Finds the largest element in the tail range of an array.
   @param a the array to be searched
   @param from the first position in a to compare
   @return the position of the largest element in range a[from]..a[a.length – 1]
*/
private static int largestPosition(int[] a, int from)
{
   int maxPos = from;
   for (int j = from + 1; j < a.length; j++)
   {
      if (a[j] > a[maxPos])
      {
         maxPos = j;
      }
   }
   return maxPos;
}

public static void selectionSort(int[] a)
{
   for ____________________________________
   {
      int maxPos = largestPosition(a, i);
      ArrayUtil.swap(a, maxPos, i);
   }
}
```
a) `(int i = 0; i < a.length – 1; i++)`
b) `(int i = 0; i < a.length; i++)`
c) `(int i = a.length; i > 0; i--)`
d) `(int i = a.length – 1; i > 0; i--)`
Answer: a

14) The code segment below is designed to add the elements in an array of integers. Select the expression needed to complete the code segment so that it calculates the running time elapsed.
```
long start = System.currentTimeMillis();
int sum = 0;
for (int k = 0; k < values.length; k++)
{
   sum = sum + values[k];
}
long runningTime = ____________________________;
```

a) `System.currentTimeMillis()`
b) `System.currentTimeMillis() - start`
c) `System.currentTimeMillis() + start`
d) `runningTime + start - System.currentTimeMillis()`
Answer: b

15) The performance of an algorithm is most closely related to what?
a) The total number of element visits
b) The total number of elements
c) The type of elements
d) The number of lines of code in the method
Answer: a

16) Consider an array with n elements. If we visit each element n times, how many total visits will there be?
a) n
b) $2n$
c) n^n
d) n^2
Answer: d

17) Suppose an array has n elements. We visit element #1 one time, element #2 two times, element #3 three times, and so forth. How many total visits will there be?
a) $2n$
b) $n(n+1)/2$
c) n^2
d) n^3
Answer: b

18) Suppose an algorithm requires a total of $3n^3 + 2n^2 - 3n + 4$ visits. In big-Oh notation, the total number of visits is of what order?
a) $n^2 * n^2$
b) n^2
c) n^6
d) n^3
Answer: d

19) In big-Oh notation, when we consider the order of the number of visits an algorithm makes, what do we ignore?
I power of two terms
II the coefficients of the terms
III all lower order terms
a) I
b) II
c) I and II
d) II and III
Answer: d

20) In big-Oh notation, suppose an algorithm requires an order of n^3 element visits. How does doubling the number of elements affect the number of visits?

a) It doubles the number of visits.
b) It quadruples the number of visits.
c) It triples the number of visits.
d) It number of visits goes up by a factor of eight.
Answer: d

21) What is the smallest value of n for which $n^2 > 3n + 4$?
a) 6
b) 5
c) 4
d) 3
Answer: b

22) How large does n need to be so $0.3n^2$ is greater than $2.5n - 3$?
a) 3
b) 5
c) 7
d) 9
Answer: c

23) Which of the following completes the selection sort method `minimumPosition()`?

```
private static int minimumPosition(int[] a, int from)
{
   int minPos = from;
   for (int i = from + 1; i < a.length; i++)
   {
      ________________
   }
   return minPos;
}
```

a) `if (a[i] > a[minPos]) { minPos = i; }`
b) `if (a[i] < a[minPos]) { minPos = i; }`
c) `if (a[i] < a[j]) { minPos = i; }`
d) `if (a[i] < a[minPos]) { i = minPos; }`
Answer: b

24) If you increase the size of a dataset fivefold, how much longer does it take to sort it with the selection sort algorithm?
a) Approximately 5 times longer
b) Approximately 20 times longer
c) Approximately 25 times longer
d) Approximately 100 times longer
Answer: c

25) How many comparisons does selection sort make when sorting an array of length n?
a) n
b) $\log(2n)$
c) $n(n + 1) / 2$
d) n
Answer: c

26) In Big-Oh notation, selection sort is a(n) ____ algorithm.
a) $O(n^2)$
b) $O(1)$
c) $\log n$
d) $O(\log n^2)$
Answer: a

27) When the size of an array increases by a factor of 100, the time required by selection sort increases by a factor of ____.

a) 2,000
b) 5,000
c) 10,000
d) 12,000
Answer: c

28) Which notation, big-Oh, theta, or omega describes the growth rate of a function?

I big-Oh
II theta
III omega

a) I
b) II and III
c) I and II
d) I, II, and III
Answer: d

29) If $f(n) = O(g(n))$ and $g(n) = O(f(n))$, what else must be true?

I $f(n) = \Omega(g(n))$
II $g(n) = \Omega(f(n))$
III $f(n) = \Theta(g(n))$

a) I
b) II
c) I and II
d) I, II, and III
Answer: d

30) In general, the expression ____ means that *f* grows no faster than *g*.

a) $f(n) = \log g$
b) $f(n) = \log g^2$
c) $f(n) = O(g(n))$
d) $g(n) = O(f(n))$
Answer: c

31) Find the simplest order of growth of the following expression: $(n^3 + n + 3)^2$.

a) $\theta(n^5)$
b) $\theta(2^n)$
c) $\theta(n^6)$
d) $\theta(n^9)$
Answer: c

32) Find the simplest order of growth of the following expression: $n^3 + \log(n^5)$.

a) $\theta(n^3)$
b) $\theta(n^3 + \log(n))$
c) $\theta(\log(n^5))$
d) $\theta(n + \log(n))$
Answer: a

33) Choose the order of the following growth rates, from slowest to fastest: $\theta(n^3)$, $\theta(n\log(n))$, $\theta(n^{3/2})$, $\theta(2^n)$.

a) $\theta(n \log(n))$, $\theta(n^{3/2})$, $\theta(n^3)$, $\theta(2^n)$
b) $\theta(n^3)$, $\theta(n \log(n))$, $\theta(n^{3/2})$, $\theta(2^n)$
c) $\theta(n^{3/2})$, $\theta(n \log(n))$, $\theta(n^3)$, $\theta(2^n)$
d) $\theta(2^n)$, $\theta(n^3)$, $\theta(n^{3/2})$, $\theta(n \log(n))$
Answer: a

34) Which function has a faster growth rate: $\theta(n^{1/2})$ or $\theta(\log(n))$?
a) $\theta(\log(n))$
b) $\theta(n^{1/2})$
c) They are the same.
d) They can't be compared.
Answer: b
35) Consider the following code snippet:
```
public static void sort(int[] a)
{
   for (int i = 1; i < a.length; i++)
   {
      int next = a[i];
      int j = i;
      while (j > 0 && a[j - 1] > next)
      {
         a[j] = a[j - 1];
         j--;
      }
      a[j] = next;
   }
}
```

What sort algorithm is used in this code?

a) insertion sort
b) selection sort
c) merge sort
d) quicksort
Answer: a

36) Which sort algorithm starts with an initial sequence of size 1, which is assumed to be sorted, and increases the size of the sorted sequence in the array in each iteration?

a) insertion sort
b) selection sort
c) merge sort
d) quicksort
Answer: a

37) What is the worst-case performance of insertion sort?

a) $O(n)$
b) $O(n^2)$
c) $O(\log(n))$
d) $O(n \log(n))$
Answer: b

38) If the array is already sorted, what is the performance of insertion sort?

a) $O(n)$
b) $O(n^2)$
c) $O(\log n)$
d) $O(n \log n)$
Answer: a

39) Which sort algorithm starts by cutting the array in half and then recursively sorts each half?

a) insertion sort
b) selection sort
c) merge sort
d) quicksort
Answer: c

40) How many times can an array with 4,096 elements be cut into two equal pieces?
a) 16
b) 12
c) 10
d) 8
Answer: b

41) How many times can an array with 729 elements be cut into three equal pieces?
a) 4
b) 5
c) 6
d) 7
Answer: c

42) The merge sort algorithm presented in section 14.4, which sorts an array of integers in ascending order, uses the `merge` method which is partially shown below. Select the condition that would be needed to complete the method so that the elements are sorted in descending order.

```
private static void merge(int[] first, int[] second, int[] a)
{
   int iFirst = 0;
   int iSecond = 0;
   int j = 0;
   while (iFirst < first.length && iSecond < second.length)
   {
      if (_______________________________)
      {
         a[j] = first[iFirst];
         iFirst++;
      }
      else
      {
         a[j] = second[iSecond];
         iSecond++;
      }
      j++;
   }
   // rest of the method follows here
}
```
a) `first[iFirst] < second[iSecond]`
b) `iFirst < iSecond`
c) `first[iFirst] > second[iSecond]`
d) `iFirst > iSecond`
Answer: c

43) In the textbook, we determined that the `merge` method requires a total of $5n$ visits. We found that the number of visits required to sort an array of n elements is $T(n) = T(n / 2) + T(n / 2) + 5n$. What does $T(n / 2)$ describe?

a) the total number of visits to merge sort an array of n elements
b) half the number of visits to merge sort an array of n elements
c) the total number of visits to merge sort an array of $n / 2$ elements
d) half the number of visits to merge sort an array of $n / 2$ elements
Answer: c

44) In the textbook, we found that the number of element visits for merge sort totaled $n + 5n\log_2 n$. Let's consider sorting 1024 elements. How many visits are needed?

a) 1.024	b) 6,144
c) 51,200	d) 52,224

Answer: d

45) In the textbook, we found that the number of element visits for merge sort totaled $n + 5n \log_2 n$. Which of the following is the appropriate big-Oh notation for merge sort?
a) $5n \log_2 n$
b) $n + \log_2 n$
c) $n + 5n$
d) $n \log_2 n$
Answer: d

46) Selection sort has $O(n^2)$ complexity. If a computer can sort 1,000 elements in 4 seconds, approximately how many seconds will it take the computer to sort 1,000 times that many, or 1,000,000 elements?
a) 16
b) 1,000
c) 1,000,000
d) 4,000,000
Answer: d

47) Merge sort has a $O(n \log_2(n))$ complexity. If a computer can sort 1,024 elements in an amount of time x, approximately how much longer will it take the computer to sort 1,024 times that many, or 1,048,576 elements?
a) 1,024 times longer
b) 2,048 times longer
c) 8,192 times longer
d) 1,048,576 times longer
Answer: c
48) Merge sort is a(n) ____ algorithm.
a) $O(n)$
b) $O(n \log(n))$
c) $O(n^2)$
d) $O(\log n)$
Answer: b

49) Assume we are using quicksort to sort an array in ascending order. Into which array location does quicksort's strategy place a pivot element after partitioning?
a) lowest index in array still available
b) highest index in array still available
c) closer to its correct final location
d) its final correct location
Answer: d

50) Assume we are using quicksort to sort an array in ascending order. What can we conclude about the indexes of two pivot elements placed in consecutive recursive calls?
a) They are randomly located.
b) They are in different halves of the array.
c) They are both in the same half of the array.
d) They are adjacent.
Answer: a

51) Assume we are using quicksort to sort an array in ascending order. What can we conclude about the elements to the left of the currently placed pivot element?
a) They are all sorted.
b) They are all less than or equal to the pivot element.
c) They are all greater than or equal to the pivot element.
d) None can equal the pivot element.

Answer: b

52) When does quicksort's worst-case run-time behavior occur?

I when the data is randomly initialized in the array
II when the data is in ascending order
III when the data is in descending order
a) I
b) II
c) III
d) II and III
Answer: d

53) Which sort algorithm starts by partitioning the array and selecting a pivot element?
a) merge sort
b) quicksort
c) selection sort
d) insertion sort
Answer: b

54) A version of which sort algorithm is used in the `sort` method in the Java `Arrays` class?
a) merge sort
b) quicksort
c) selection sort
d) insertion sort
Answer: b

55) Which sort algorithm is used in the `sort` method in the Java `Arrays` class when the array length is less than 7?
a) merge sort
b) quicksort
c) selection sort
d) insertion sort
Answer: d

56) Which of the sorts in the textbook are based on the strategy of divide and conquer?
I quicksort
II mergesort
III insertion sort
a) I b) II
c) III d) I and II
Answer: d

57) Which of the sorts in the textbook can be characterized by the fact that the best case will have a running time of $\theta(n)$ if the data is already sorted?
I quicksort
II selection sort
III insertion sort

a) I
b) II
c) III
d) I and III
Answer: c

58) Which of the sorts in the textbook can be characterized by the fact that even in the worst case the running time will be $O(n \log(n))$?
I quicksort
II selection sort
III merge sort

a) I
b) II
c) III
d) I and III
Answer: c

59) In the worst case, quicksort is a(n) ____ algorithm.
a) $O(n)$
b) $O(\log(n))$
c) $O(n^2)$
d) $O(n \log n)$
Answer: c

60) In the worst case, a linear search locates a value in an array of length *n* in ____ steps.
a) $O(n^2)$
b) $O(\log n)$
c) $O(n)$
d) $O(\log n^2)$
Answer: c

61) The following code is an example of a ___ search.

```
public static int search(int[] a, int v)
{
   for (int i = 0; i < a.length; i++)
   {
      if (a[i] == v) { return i; }
   }
   return -1;
}
```

a) sorted
b) binary
c) linear
d) random
Answer: c

62) Suppose you have a phone number and need to find the address that it corresponds to. If there are 2,000,000 entries, how many do you expect to search in a printed phone directory before finding the address you are looking for?

a) Approximately 50,000 records.
b) Approximately 75,000 records.
c) Approximately 1,000,000 records.
d) Approximately 1,200,000 records.
Answer: c

63) If an element is present in an array of length *n*, how many element visits, in the worst case, are necessary to find it using a linear search?
a) $n / 2$
b) n
c) $2n$
d) n^2
Answer: b
64) If an element is present in an array of length *n*, how many element visits, on average, are necessary to find it using a linear search?
a) $n / 2$
b) n
c) $2n$
d) n^2
Answer: a

65) Another name for linear search is ____ search.
a) sorted
b) sequential
c) random
d) binary
Answer: b

66) If you implement a recursive linear search, its performance will be ____.
a) $O(n)$
b) $O(n^2)$
c) $O(\log (n))$
d) $O(n \log (n))$
Answer: a

67) Binary search is an ____ algorithm.
a) $O(n)$
b) $O(n^2)$
c) $O(\log n)$
d) $O(n \log n)$
Answer: c

68) Given an ordered array with 15 elements, how many elements must be visited in the worst case of binary search?
a) 8
b) 4
c) 3
d) 2
Answer: b

69) Given an ordered array with 31 elements, how many elements must be visited in the worst case of binary search?
a) 16
b) 8
c) 5
d) 4
Answer: c

70) The analysis for the number of visits in a binary search begins with the equation,
$T(n) = T(n / 2) + 1$. What does the number 1 represent in this equation?
a) One element visit before a recursive call on one half of the array.
b) The total number of recursions required.
c) The fact that the number of elements is odd.
d) The fact that we search either the left half or the right half, but not both.
Answer: a

71) Suppose we are using binary search on an array with approximately 1,000,000 elements. How many visits should we expect to make in the worst case?
a) 1 b) 16
c) 20 d) 30
Answer: c

72) A binary search is generally ____ a linear search.
a) slower than
b) equal to
c) less efficient than
d) faster than
Answer: d

73) A search technique where, in each step, you split the size of the search in half is called a____ search.
a) random
b) binary
c) linear
d) merging
Answer: b

74) Can you search the following array using binary search?
```
int[] A = {6, 5, 4, 2, 0, 1, -1, -17};
```
a) Yes. Binary search can be applied to any array.
b) No. Binary search can be applied to a sorted array only.
c) Yes, but the algorithm runs slower because the array is in descending order.
d) No, negative numbers are not allowed because they indicate that a value is not present.
Answer: b

75) The partial linear search method below is designed to search an array of `String` objects. Select the expression that would be needed to complete the method.
```
public static int search(String[] a, String item)
{
   for (int i = 0; i < a.length; i++)
   {
      if ( ___________________________ )
      {
         return i;
      }
      return -1;
   }}
```
a) `a[i] == item`
b) `a[i].compareTo(item)`
c) `a[i].equals(item)`
d) `a[i].indexOf(item)`
Answer: c

76) The partial binary search method below is designed to search an array of `String` objects sorted in ascending order. Select the expression that would be needed to complete the method.

```
public static int search(String[] a, int low, int high, String item)
{
   if (low <= high)
   {
      int mid = (low + high) / 2;
      int result = ____________________________;
      if (result == 0)
      {
         return mid;
      }
      else if (result < 0)
      {
         return search(a, mid + 1, high, item);
      }
      else
      {
         return search(a, low, mid - 1, item);
      }
   }
   return -1;
}
```

a) `a[low].compareTo(item)`
b) `item.equals(a[mid])`
c) `item.compareTo(a[mid])`
d) `a[mid].compareTo(item)`
Answer: d

77) A portion of your program implements a loop in which each step contains a fixed number of actions. What can you conclude about the running time of this section of code?
a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(n \log(n))$.
Answer: a

78) Which of the following statements about running times of algorithms is correct?
a) An algorithm that is $O(1)$ means that only one comparison takes place.
b) When determining the running time, constants are not taken into consideration.
c) When determining the running time, lower-order terms must be taken into consideration.
d) An algorithm that is $O(n)$ means that the number of comparisons does not grow as the size of the array increases.
Answer: b
79) A portion of your program includes the loop shown in the code snippet below to examine the elements of an array `arr`:

```
int count = 0;
int targetVal = 70;
for (int i = 0; i < arr.length; i++)
{
   if (arr[i] >= targetVal)
   {
      count++;
   }
}
```

What can you conclude about the running time of this section of code?
a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(n \log(n))$.
Answer: a

80) The method `findLargest` examines the elements of an array `arr` which contains non-negative values

```
public static int findLargest(int[] arr)
{
   int curLargest = -1;
   for (int i = 0; i < arr.length; i++)
   {
      if (arr[i] >= curLargest)
      {
         curLargest = arr[i];
      }
    }
    return curLargest;
}
```

What can you conclude about the running time of this section of code?
a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(n \log(n))$.
Answer: a

81) The method `checkArray` examines an array `arr`:

```
public static boolean checkArray(int[] arr)
{
     if (arr[0] >= arr[arr.length -1])
    {
        return true;
    }
    return false;
}
```

What can you conclude about the running time of this section of code?
a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(1)$.
Answer: d

82) An algorithm that tests whether the first array element is equal to any of the other array elements would be an ____ algorithm.
a) $O(1)$
b) $O(n)$
c) $O(\log(n))$
d) $O(n \log(n))$
Answer: b

83) A portion of your program includes the loops shown in the code snippet below to examine the elements of two arrays, `arr1` and `arr2`, both of length n:

```
int matches = 0;
for (int i = 0; i < arr1.length; i++)
{
    for (int j = 0; j < arr2.length; j++)
    {
       if (arr1[i].equals(arr2[j]))
       {
          matches++;
       }
    }
}
```

What can you conclude about the running time of this section of code?

a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(n \log(n))$.
Answer: b

84) A portion of your program includes the method shown in the code snippet below to examine the elements of an array `arr`:

```
private int findElement(int[] arr, int newVal)
{
    int pos = Arrays.binarySearch(arr, newVal);
    return pos;
}
```

What can you conclude about the running time of this section of code?
a) Its running time will be $O(n)$.
b) Its running time will be $O(n^2)$.
c) Its running time will be $O(\log(n))$.
d) Its running time will be $O(n \log(n))$.
Answer: c

85) An algorithm that cuts the work in half in each step is an ____ algorithm.
a) *O*(*n*)
b) *O*(n^2)
c) *O*(log (*n*))
d) *O*(*n* log (*n*))
Answer: c

86) The code segment below prints some of the elements in an array with size n. Select an expression to complete the code segment so that the resulting algorithm has *O(log n)* running time.

```
for ______________________________
{
   System.out.println(array[j]);
}
```

a) `(int j = 0; j < array.length; j = j + 2)`
b) `(int j = 1; j < array.length; j = j * 2)`
c) `(int j = 0; j < array.length / 2; j++)`
d) `(int j = 0; j < array.length; j++)`
Answer: b

87) The code segment below displays a pattern of asterisks. Select an expression to complete the code segment so that the resulting algorithm has *O(n)* running time.

```
for (int k = 0; k < n; k++)
{
   for __________________________
   {
      System.out.print("*");
   }
   System.out.println();
}
```

a) `(int j = n; j > 0; j = j / 2)`
b) `(int j = n; j > 0; j--)`
c) `(int j = 1; j < k; j++)`
d) `(int j = 1; j <= 10; j++)`
Answer: d

88) The code segment below displays a table of numbers. Select an expression to complete the code segment, so that the resulting algorithm has *O*(n^2) running time.

```
for (int k = 1; k <= n; k++)
{
   for __________________________
   {
      System.out.print(j + " ");
   }
   System.out.println();
}
```

a) `(int j = n; j > 0; j = j / 2)`
b) `(int j = 1; j < k; j = j * 2)`
c) `(int j = 0; j < k; j++)`
d) `(int j = 1; j <= 200; j++)`
Answer: c

89) Assume that `names` is an array of `String` objects that has been initialized with a large number of elements. Select the statement that would sort the elements in `names` in ascending alphabetic order.
a) `Arrays.sort(names, 0, names.length - 1);`
b) `Arrays.sort(names);`
c) `Collections.sort(names, 0, names.length - 1);`
d) `Collections.sort(names);`
Answer: b

90) Assume that `bands` is an `ArrayList` of `String` objects which contains a number of elements in ascending order. Select a statement to complete the code segment below, which invokes the Java library `binarySearch` method to search for the string `"Beatles"`. If the list does not already contain the string, it should be inserted in an appropriate location so that the list remains sorted.

```
int index = Collections.binarySearch(bands, "Beatles");
if (index < 0)
{
   ______________________________
```

`}`
a) `bands.add(-1 * index + 1, "Beatles");`
b) `bands.add(index + 1, "Beatles");`
c) `bands.add(-1 * index, "Beatles");`
d) `bands.add(-1 - index, "Beatles");`
Answer: d

91) The sort method of the `Arrays` class sorts arrays containing objects of classes that implement the ____ interface.
a) `Sortable`
b) `Ordered`
c) `Array`
d) `Comparable`
Answer: d

92) The ____ class contains a sort method that can sort array lists.
a) `Sorting`
b) `Arrays`
c) `Collections`
d) `Linear`
Answer: c

93) The `binarySearch` method of the `Collections` class returns a value in the form of `-k - 1` when the target item you are searching for was not found in the array. What does `k` represent?
a) It is the position of an existing item, and indicates that your target item should come after this existing item.
b) It is the position of an existing item, and indicates that your target item should come before this existing item.
c) It represents the number of times the array was accessed by the `binarySearch` method, and can be used for analyzing performance.
d) Nothing – it is an arbitrary value.
Answer: b

94) Given the following code snippet for searching an array:
```
int[] arr = {3, 8, 12, 14, 17};
int newVal = 15;
int pos = Arrays.binarySearch(arr, newVal);
```
What value will `pos` have when this code is executed?
a) 5 b) -5
c) 6 d) -6
Answer: b

95) Given the following code snippet for searching an array:
```
int[] arr = {23, 25, 29, 34, 42};
int newVal = 15;
int pos = Arrays.binarySearch(arr, newVal);
```
What value will `pos` have when this code is executed?
a) 0 b) 1
c) -1 d) -2
Answer: c

96) If a call to the `Arrays` static method `binarySearch` returns a value of -10, what can be concluded?
I the element is not in the array
II the element is at index 10
III the element can be inserted at index 9
a) I
b) II
c) III
d) I and III
Answer: d

97) If a call to the `Arrays` static method `binarySearch` returns a value of 7, what can be concluded?
I the element is not in the array
II the element is at index 7
III the element occurs 7 times in the array
a) I
b) II
c) III
d) II and III
Answer: b

98) If the `Arrays` static method `binarySearch` is called on an array of 10 elements and returns a value of 10, what can be concluded?

I the element is not found
II that value cannot be returned from method `binarySearch`
III if added, the element would be the largest element in the array
a) I
b) II
c) III
d) I and III
Answer: b

99) If you want to use the `Comparable` interface, you must implement a single method called ____.
a) `comparable`
b) `compareTo`
c) `comparator`
d) `compare`
Answer: b

100) Which of the following classes implement the `Comparable` interface?

I `Date`
II `Collections`
III `String`
a) I
b) I and II
c) I and III
d) II and III
Answer: c

101) Suppose the call `obj1.compareTo(obj2)` returns 0. What can definitely be concluded from the return value?
I `obj1` and `obj2` are the same object
II `obj1` and `obj2` objects cannot be compared
III the properties of `obj1` and `obj2` that are being compared have identical values
a) I b) II
c) I and III d) III
Answer: d

102) Which of the following arrays can be used in a call to the `Arrays.sort` method?

I Any array with primitive numeric data, such as `int`, `double`, …
II Arrays of `String` or numeric wrapper classes like, `Integer`, `Double`, …
III Any class that implements the `Comparable` interface
a) I
b) II
c) I and II
d) I, II and III
Answer: d

103) Suppose objects `a` and `b` are from a user-defined class that implements the `Comparable` interface. Which condition tests the `compareTo` method's return value to determine that `a` will precede `b` when the `sort` method is called?
a) `a.compareTo(b) == -1`
b) `a.compareTo(b) == 0`
c) `a.compareTo(b) < 0`
d) `a.compareTo(b) > 0`
Answer: c

104) Suppose objects `a` and `b` are from a user-defined class that implements the `Comparable` interface. What must be true about the return value of `a.compareTo(b)` for the `compareTo` method that this class implements?

a) It must return 1 if `a` comes before `b`, 0 if they are the same, and -1 if `a` comes after `b`.
b) It must return -1 if `a` comes before `b`, 0 if they are the same, and 1 if `a` comes after `b`.
c) It must return a positive value if `a` comes before `b`, 0 if they are the same, and a negative value if `a` comes after `b`.
d) It must return a negative value if `a` comes before `b`, 0 if they are the same, and a positive value if `a` comes after `b`.
Answer: d

105) Suppose you wish to implement the `Comparable` interface to allow your `Vehicle` class to compare `Auto` objects only. Which of the following is the correct way to do this?

a) `public class Auto implements Comparable<Vehicle>`
b) `public class Vehicle implements Comparable`
c) `public class Vehicle implements Comparable<Auto>`
d) `public class Vehicle implements Comparable(Auto)`
Answer: c

106) Suppose a developer gets class `XYZ` files and documentation from a subcontractor. This class does not implement the `Comparable` interface. What must be true in order for the developer to sort an array of `XYZ` objects without modifying the `xyz` class?
a) The `XYZ` class must implement the `Sortable` interface.
b) `XYZ` objects must be randomly distributed.
c) `XYZ` objects must be ordered.
d) The developer must supply a comparator object belonging to a class that implements the `Comparator<XYZ>` interface.
Answer: d

107) Suppose you wish to sort an array list of objects, but the object class does not implement the `Comparable` interface. Because you are not allowed to modify this class, you decide to provide a comparator object that implements the `Comparator` interface. Which method must you implement from this interface to achieve your objective?
a) the `sort` method.
b) the `compare` method.
c) the `compareTo` method.
d) the `compareObject` method.
Answer: b

108) Complete the following code for an interface that classes who wish to compare `Auto` objects can implement.

```
public interface Comparator<Auto>
{
   ___________________________
}
```

a) `boolean compare(Auto a, Auto b);`
b) `single compareTo(Auto a, Auto b);`
c) `int compare(Auto a, Auto b);`
d) `single compare(Auto a, Auto b);`
Answer: c

109) When your class implements a comparator object, you must implement the `compare` method. What must be true about the return value from this method when comparing two objects, `a` and `b` with a call to `a.compare(b)`?

a) It must return 1 if `a` comes before `b`, 0 if they are the same, and -1 if `a` comes after `b`.
b) It must return -1 if `a` comes before `b`, 0 if they are the same, and 1 if `a` comes after `b`.
c) It must return a positive value if `a` comes before `b`, 0 if they are the same, and a negative value if `a` comes after `b`.
d) It must return a negative value if `a` comes before `b`, 0 if they are the same, and a positive value if `a` comes after `b`.
Answer: d

110) Complete the following code that is intended to provide a comparator interface that will be used to create an object that implements the `Comparator` interface so that object can compare `Auto` objects.

```
_________________________________
{
   int compare(Auto a, Auto b);
}
```

a) `public class Comparator<Auto>`
b) `public class Comparator<Auto> implements Comparator`
c) `public interface Auto<Comparator>`
d) `public interface Comparator<Auto>`
Answer: d

Chapter 15 :The Java Collections Framework

1) The `ArrayList` class implements the ____.
a) Queue interface.
b) Set interface.
c) List interface.
d) Stack interface.
Answer: c

2) A list is a collection that ____.
a) should be used when you need to remember the order of elements in the collection.
b) allows items to be added at one end and removed at the other end.
c) does not allow elements to be inserted in any position.
d) manages associations between keys and values.
Answer: a

3) A stack is a collection that ____.
a) remembers the order of elements, and allows elements to be added and removed only at one end.
b) does not remember the order of elements but allows elements to be added in any position.
c) remembers the order of elements and allows elements to be inserted in any position.
d) remembers the order of elements and allows elements to be inserted only at one end and removed only at the other end.
Answer: a

4) A queue is a collection that ____.
a) remembers the order of elements, and allows elements to be added and removed only at one end.
b) does not remember the order of elements but allows elements to be added in any position.
c) remembers the order of elements and allows elements to be inserted in any position.
d) remembers the order of elements and allows elements to be inserted only at one end and removed only at the other end.
Answer: d

5) A collection without an intrinsic order is called a ____.
a) list
b) stack
c) set
d) queue
Answer: c

6) A collection that allows items to be added only at one end and removed only at the other end is called a ____.
a) list
b) stack
c) set
d) queue
Answer: d

7) A collection that remembers the order of items, and allows items to be added and removed only at one end is called a ____.

a) list
b) stack
c) set
d) queue
Answer: b

8) A collection that allows speedy insertion and removal of already-located elements in the middle of it is called a ____.
a) linked list
b) stack
c) set
d) queue
Answer: a

9) Which data structure would best be used for keeping track of a growing set of groceries to be purchased at the food market?
a) queue
b) stack
c) list
d) array
Answer: c

10) Select an appropriate expression to complete the following code segment, which is designed to print a message if the string stored in `name` is part of the `players` collection.

```
Collection<String> players = new ArrayList<String>();
// code to add elements to the collection here

if ______________________________________
   System.out.print(name + " is one of the players in the collection.");
```

a) `(players.indexOf(name))`
b) `(players.contains(name))`
c) `(players.search(name))`
d) `(players.equals(name))`
Answer: b

11) What is included in a linked list node?
I   a reference to its neighboring nodes
II   an array reference
III a data element
a) I
b) II
c) II and III
d) I and III
Answer: d

12) Which of the following statements about linked lists is correct?
a) Once you have located the correct position, adding elements in the middle of a linked list is inefficient.
b) Visiting the elements of a linked list in random order is efficient.
c) When a node is removed, all nodes after the removed node must be moved down.
d) Linked lists should be used when you know the correct position and need to insert and remove elements efficiently.
Answer: d

13) We might choose to use a linked list over an array list when we will not require frequent ____.

I   random access
II   inserting new elements
III removing of elements
a) I
b) II
c) III
d) II and III
Answer: a

14) Which nodes need to be updated when we insert a new node to become the fourth node from the beginning of a doubly-linked list?
a) The current third node.
b) The current third and fourth nodes.
c) The current first node.
d) The current fourth and fifth nodes.
Answer: b
15) A binary search requires ____ access.
a) sequential
b) random
c) sorted
d) arbitrary
Answer: b

16) A linear search only requires ____ access.
a) sequential
b) random
c) sorted
d) arbitrary
Answer: a

17) Rather than storing values in an array, a linked list uses a sequence of ____.
a) indexes
b) nodes
c) elements
d) accessors
Answer: b

18) Which of the following algorithms would be efficiently executed using a `LinkedList`?
a) Tracking paths in a maze.
b) Binary search.
c) Remove first *n*/ 2 elements from a list of *n* elements.
d) Read *n* / 2 elements in random order from a list of *n* elements.
Answer: c

19) What type of access does a `LinkedList` provide for its elements?
a) sequential
b) semi-random
c) random
d) sorted
Answer: a

20) Consider the following code snippet:

```
LinkedList<String> words = new LinkedList<String>();
words.addLast("abc");
words.addLast("def");
words.addLast("ghi");
System.out.print(words.removeLast());
System.out.print(words.removeFirst());
System.out.print(words.removeLast());
```

What will this code print when it is executed?

a) abcdefghi
b) ghiabcdef
c) abcghidef
d) defghiabc
Answer: b

21) Consider the following code snippet:

```
LinkedList<String> words = new LinkedList<String>();
words.addFirst("abc");
words.addLast("def");
words.addFirst("ghi");
System.out.print(words.removeLast());
System.out.print(words.removeFirst());
System.out.print(words.removeLast());
```

What will this code print when it is executed?

a) abcdefghi
b) ghiabcdef
c) abcghidef
d) defghiabc
Answer: d

22) The term ____ is used in computer science to describe an access pattern in which the elements are accessed in arbitrary order.
a) sequential access
b) random access
c) sorted access
d) arbitrary access
Answer: b

23) Which Java package contains the `LinkedList` class?
a) `java.lang`
b) `java.util`
c) `java.collections`
d) `java.io`
Answer: b

24) What can a generic class be parameterized for?
a) properties b) iterators
c) type d) methods
Answer: c

25) Assume you have created a linked list named `myList` that currently holds some number of `String` objects. Which of the following statements correctly adds a new element to the beginning of `myList`?
a) `myList.addFirst("Harry");`
b) `myList.add("Harry");`
c) `myList.insert("Harry");`
d) `myList.put("Harry");`
Answer: a

26) Assume you have created a linked list name `myList` that currently holds some number of `String` objects. Which of the following statements correctly removes an element from the end of `myList`?
a) `myList.remove();`
b) `myList.removeLast();`
c) `myList.getLast();`
d) `myList.pop();`
Answer: b

27) A(n) ____ is a data structure used for collecting a sequence of objects that allows efficient addition and removal of already-located elements in the middle of the sequence.
a) stack b) queue
c) linked list d) priority queue
Answer: c

28) What is the meaning of the type parameter `E`, in the `LinkedList<E>` code fragment?
a) The elements of the linked list are of class `E.`
b) The elements of the linked list are of any subclass of class `E.`
c) The elements of the linked list are any type supplied to the constructor.
d) The elements of the linked list are of class `Object`.
Answer: c

29) Which method is NOT part of the `ListIterator` interface?
a) `delete`
b) `add`
c) `next`
d) `previous`
Answer: a

30) Consider the code snippet shown below. Assume that `employeeNames` is an instance of type `LinkedList<String>`.

```
for (String name : employeeNames)
{
   // Do something with name here
}
```

Which element(s) of `employeeNames` does this loop process?
a) no elements
b) all elements
c) elements meeting a condition
d) the most recently added elements
Answer: b

31) Which method is NOT part of the `ListIterator` generic class?
a) `hasNext`
b) `hasMore`
c) `hasPrevious`
d) `add`
Answer: b

32) Select an appropriate expression to complete the following code segment, which is designed to print a message if the string stored in `name` is the first element of the `players` linked list.

```
LinkedList<String> players = new LinkedList<String>();
// code to add elements to the linked list
if _______________________________________
   System.out.print(name + " is the first player on the list.");
```

a) `(players.indexOf(name) == 1)`
b) `(players.contains(name))`
c) `(players.getFirst().equals(name))`
d) `(players[0].equals(name))`
Answer: c

33) Select an appropriate expression to complete the method below. The method should return the number of times that the string stored in `name` appears in `theList`.

```
public static int count(LinkedList<String> theList, String name)
{
   int number = 0;
   Iterator<String> iter = theList.iterator();

   while (______________________)
   {
      if (iter.next().equals(name))
      {
         number++;
      }
   }
   return number;
}
```

a) `iter.hasNext()`
b) `iter.next() != null`
c) `theList.hasNext()`
d) `theList.next() != null`
Answer: a

34) Select an appropriate expression to complete the following method, which is designed to visit the elements in `theList` and replace each occurrence of the string `"hello"` with the string `"goodbye"`.

```
public static void helloGoodbye(LinkedList<String> theList)
{
   ListIterator<String> iterator = theList.listIterator();
   while (iterator.hasNext())
   {
      if (iterator.next().equals("hello"))
      {
         _____________________________
      }
   }
}
```

a) `iterator.replace("hello", "goodbye");`
b) `iterator.next() = "goodbye";`
c) `iterator.previous("goodbye");`
d) `iterator.set("goodbye");`
Answer: d

35) When using a list iterator, on which condition will the `IllegalStateException` be thrown?
a) Calling `remove` after calling `next`.
b) Calling `next` after calling `previous`.
c) Calling `remove` after calling `remove`.
d) Calling `remove` after calling `previous`.
Answer: c

36) When using a list iterator, on which condition will the `IllegalStateException` be thrown?
a) Calling `remove` after calling `next`.
b) Calling `add` after calling `previous`.
c) Calling `remove` after calling `add`.
d) Calling `previous` after calling `previous`.
Answer: c

37) Which of the following statements about the `LinkedList` class is correct?

a) When you use the `add` method, the new element is inserted before the iterator, and the iterator position is advanced by one position.
b) When you use the `add` method, the new element is inserted after the iterator, and the iterator position is advanced by one position.
c) When you use the `add` method, the new element is inserted before the iterator, and the iterator position is not moved
d) When you use the `add` method, the new element is inserted after the iterator, and the iterator position is not moved.
Answer: a

38) When using a list iterator, on which condition will the `NoSuchElementException` be thrown?
a) Calling `next` when you are past the end of the list.
b) Calling `next` when the iterator points to the last element.
c) Calling `remove` after calling `add`.
d) Calling `remove` after calling `previous`.
Answer: a

39) A linked list ____ encapsulates a position anywhere inside the linked list.
a) accessor
b) index
c) operation
d) iterator
Answer: d

40) You use a(n) ____ to access elements inside a linked list.
a) accessor
b) index
c) list iterator
d) queue
Answer: c

41) A linked list allows ____ access, but you need to ask the list for an iterator.
a) sequential
b) random
c) sorted
d) arbitrary
Answer: a

42) The nodes of a(n) ____ linked list class store two links: one to the next element and one to the previous one.
a) array
b) singly
c) doubly
d) randomly
Answer: c

43) Assume you are using a doubly-linked list data structure with many nodes. What is the minimum number of node references that are required to be modified to remove a node from the middle of the list? Consider the neighboring nodes.
a) 1
b) 2
c) 3
d) 4
Answer: b

44) In a linked list data structure, when does the reference to the first node need to be updated?

I inserting into an empty list
II deleting from a list with one node
III deleting an inner node
a) I
b) II
c) I and II
d) III
Answer: c

45) Consider the following code snippet:

```
LinkedList<String> myLList = new LinkedList<String>();
myLList.add("Mary");
myLList.add("John");
myLList.add("Sue");
ListIterator<String> iterator = myLList.listIterator();
iterator.next();
iterator.next();
iterator.add("Robert");
iterator.previous();
iterator.previous();
iterator.remove();
System.out.println(myLList);
```

What will be printed when this code is executed?

a) `[Mary, John, Robert, Sue]`
b) `[Mary, John, Sue]`
c) `[Mary, Robert, Sue]`
d) `[John, Robert, Sue]`
Answer: c

46) Which of the following statements about data structures is correct?

a) Inserting and removing elements that have already been located is faster with a list than with a set.
b) Accessing elements in a linked list in a random fashion is efficient.
c) Adding and removing already-located elements in the middle of a linked list is efficient.
d) A set is an ordered collection of unique elements.
Answer: c

47) Which of the following statements about sets is correct?

a) Inserting and removing elements that have already been located is faster with a list than with a set.
b) A set allows duplicate values.
c) You can add an element to a specific position within a set.
d) A set is a collection of unique elements organized for efficiency.
Answer: d

48) Which of the following statements about hash tables is NOT correct?
a) Elements are grouped into smaller collections that share the same characteristic.
b) You can form hash tables holding objects of type `String`.
c) You can add an element to a specific position within a hash table.
d) The value used to locate an element in a hash table is called a hash code.
Answer: c

49) Which of the following statements about the `TreeSet` class is NOT correct?
a) Elements are stored in sorted order.
b) Elements are arranged in linear fashion.
c) Elements are stored in nodes.
d) To use a `TreeSet`, it must be possible to compare the elements.
Answer: b

50) Select an appropriate declaration to complete the following code segment, which is designed to read strings from standard input and display them in increasing alphabetical order, excluding any duplicates.

```
_____________________________________________
Scanner input = new Scanner(System.in);
while (input.hasNext())
{
   words.add(input.next());
}
System.out.print(words);
```
a) `LinkedList<String> words = new LinkedList<String>();`
b) `Set<String> words = new Set<String>();`
c) `Set<String> words = new HashSet<String>();`
d) `Set<String> words = new TreeSet<String>();`
Answer: d

51) Select an appropriate expression to complete the following method, which is designed to return the number of elements in the parameter array `numbers`. If a value appears more than once, it should be counted exactly once.
```
public static int countElementsOnce(int[] numbers)
{
   Set<Integer> values = new HashSet<Integer>();
   for (int num: numbers)
   {
      values.add(num);
   }
   ________________________
}
```
a) `return numbers.length;`
b) `return values.length();`
c) `return values.size();`
d) `return numbers.length - values.size();`
Answer: c

52) To create a `TreeSet` for a class of objects, the object class must ____.
a) create an iterator.
b) implement the `Comparable` interface.
c) implement the `Set` interface.
d) create a `Comparator` object.
Answer: b

53) Which of the following statements about manipulating objects in a set is correct?
a) If you try to add an element that already exists, an exception will occur.
b) If you try to remove an element that does not exist, an exception will occur.
c) You can add an element at the position indicated by an iterator.
d) A set iterator visits elements in the order in which the set implementation keeps them.
Answer: d

54) Which of the following statements about manipulating objects in a set is correct?
a) If you try to add an element that already exists, an exception will occur.
b) A set iterator visits elements in the order in which they were added to the set.
c) You can add an element at the position indicated by an iterator.
d) You can remove an element at the position indicated by an iterator.
Answer: d

55) Assume that you have declared a set named `mySet` to hold `String` elements. Which of the following statements will correctly insert an element into `mySet`?
a) `mySet.insert("apple");`
b) `mySet.put(apple");`
c) `mySet.push("apple");`
d) `mySet.add("apple");`
Answer: d

56) Assume that you have declared a set named `mySet` to hold `String` elements. Which of the following statements will correctly remove an element from `mySet`?
a) `mySet.get("apple");`
b) `mySet.remove("apple");`
c) `mySet.pop("apple");`
d) `mySet.delete("apple");`
Answer: b
57) Complete the following code snippet, which is intended to determine if a specific value in a variable named `targetWord` appears in a set of `String` values named `mySet`:

```
for (String aWord : mySet)
{
   _______________________
   {
      System.out.println ("The word " + targetWord + " was found.");
   }
)
```
a) `if (mySet.equalsIgnoreCase(targetWord))`
b) `if (mySet == targetWord)`
c) `if (mySet.contains(targetWord))`
d) `if (mySet.get(targetWord))`
Answer: c

58) Which of the following statements about manipulating objects in a map is NOT correct?
a) Use the `add` method to add a new element to the map.
b) Use the `get` method to retrieve a value from the map.
c) Use the `keyset` method to get the set of keys for the map.
d) Use the `remove` method to remove a value from the map.
Answer: a

59) Complete the following code, which is intended to print out all key/value pairs in a map named `myMap` that contains `String` data for student IDs and names:
```
Map<String, String> myMap = new HashMap<String, String>();
. . .
_______________________________
for (String aKey : mapKeySet)
{
   String name = myMap.get(aKey);
   System.out.println("ID: " + aKey + "->" + name);
}
```

a) `Map<String, String> mapKeySet = myMap.keySet();`
b) `Set<String, String> mapKeySet = myMap.keySet();`
c) `Set<String> mapKeySet = myMap.getKeySet();`
d) `Set<String> mapKeySet = myMap.keySet();`
Answer: d

60) Complete the following code, which is intended to print out all key/value pairs in a map named `myMap` that contains `String` data for student IDs and names:

```
Map<String, String> myMap = new HashMap<String, String>();
. . .

Set<String> mapKeySet = myMap.keySet();
for (String aKey : mapKeySet)
{
   ___________________________;
   System.out.println("ID: " + aKey + "->" + name);
}
```

a) `String name = myMap.get(aKey);`
b) `String name = myMap.next(aKey);`
c) `String name = MapKeySet.get(aKey);`
d) `String name = MapKeySet.next(aKey);`
Answer: a

61) Assume that you have declared a map named `myMap` to hold `String` elements with `Integer` keys. Which of the following statements will correctly insert an element into `myMap`?
a) `myMap.insert(3, "apple");`
b) `myMap.put(3, "apple");`
c) `myMap.push(3, "apple");`
d) `myMap.add(3, "apple");`
Answer: b

62) Assume that you have declared a map named `myMap` to hold `String` elements with `Integer` keys. Which of the following statements will correctly remove an element from `myMap`?
a) `myMap.get(3);`
b) `myMap.remove(3);`
c) `myMap.pop(3);`
d) `myMap.delete(3);`
Answer: b

63) Assume that you have declared a map named `myMap` to hold `String` elements with `Integer` keys. Which of the following statements will correctly retrieve the value of an element from `myMap` by using its key?
a) `myMap.get("apple");`
b) `myMap.peek("apple");`
c) `myMap.get(3);`
d) `myMap.peek(3);`
Answer: c

64) Your program uses a `Map` structure to store a number of user ids and corresponding email addresses. Although each user must have a unique id, two or more users can share the same email address. Select an appropriate expression to complete the method below, which adds a new id and email address to the map only if the id is not already in use. If the id is already in use, an error message is printed.

```
public static void addUserID(Map<String, String> users, String id, String email)
{
   String currentEmail = users.get(id);
   if (____________________)
   {
      users.put(id, email);
   }
   else
   {
      System.out.println(id + " is already in use.");
   }
}
```

a) `currentEmail.equals(email)`
b) `!currentEmail.equals(email)`
c) `currentEmail == null`

d) `currentEmail != null`
Answer: c

65) Which of the following statements about manipulating objects in a map is NOT correct?
a) If you attempt to retrieve a value with a key that is not associated with any value, you will receive a null result.
b) You cannot change the value of an existing association in the map; you must delete it and re-add it with the new values.
c) Use the `get` method to retrieve a value associated with a key in the map.
d) Use the `put` method to add an element to the map.
Answer: b

66) Consider the following code snippet:

```
Map<String, Integer> scores;
```

If you need to visit the keys in sorted order, which of the following statements will create a structure to support this?
a) `scores = new HashMap<String, Integer>;`
b) `scores = new TreeMap<String, Integer>;`
c) `scores = new Map<String, Integer>;`
d) `scores = new HashTable<String, Integer>;`
Answer: b

67) Consider the following code snippet:

```
Map<String, Integer> scores;
```
You expect to retrieve elements randomly by key, and want fastest retrieval times. Which of the following statements will create a structure to support this?
a) `scores = new HashMap<String, Integer>;`
b) `scores = new TreeMap<String, Integer>;`
c) `scores = new Map<String, Integer>;`
d) `scores = new TreeSet<String, Integer>;`
Answer: a

68) You want to enumerate all of the keys in a map named `myMap` whose keys are type `String`. Which of the following statements will allow you to do this?
a)
```
Set<String> keySet = myMap.keySet();
for (String key : keySet) {. . . }
```
b)
```
Set<String> keySet = myMap.getKeys();
for (String key : keySet) {. . . }
```
c)
```
Set<String> keySet = myMap.keys();
for (String key : keySet) {. . . }
```
d)
```
Set<String> keySet = myMap.getKeySet();
for (String key : keySet) {. . . }
```
Answer: a

69) You need to access values by an integer position. Which collection type should you use?
a) Map
b) Hashtable
c) ArrayList
d) Queue
Answer: c

70) You need to access values in objects by a key that is not part of the object. Which collection type should you use?
a) Map
b) Hashtable
c) ArrayList
d) Queue
Answer: a

71) You need to access values in the order in which they were added (first in, first out), and not randomly. Which collection type should you use?
a) Map
b) Hashtable
c) Stack
d) Queue
Answer: d

72) You need to access values in the opposite order in which they were added (last in, first out), and not randomly. Which collection type should you use?
a) Map
b) Hashtable
c) Stack
d) Queue
Answer: c

73) You need to access values using a key, and the keys must be sorted. Which collection type should you use?
a) TreeMap
b) ArrayList
c) HashMap
d) Queue
Answer: a

74) You need to access values by their position. Which collection type should you use?
a) TreeSet
b) ArrayList
c) Stack
d) Queue
Answer: b

74) You have decided to store objects of a class in a `TreeSet` structure. Which of the following statements is correct?
a) If the object class implements the `Comparable` interface, and the sort order in the `compare` method is acceptable, you do not have to do anything else.
b) If the object class implements the `Comparable` interface, and the sort order in the `compareTo` method is acceptable, you do not have to do anything else.
c) If the object class implements the `Comparable` interface, and the sort order in the `compare` method is acceptable, you must create a comparator object.
d) If the object class implements the `Comparable` interface, and the sort order in the `compareTo` method is acceptable, you must create a comparator object.
Answer: b

75) Which data structure would best be used for storing a set of numbers and sorting them in ascending order?
a) queue
b) stack
c) list
d) array
Answer: d

76) Which of the following algorithms would be efficiently executed on an `ArrayList`?
a) add 1 element to the middle of a list with *n* elements
b) add *n* / 2 elements to a list with *n* / 2 elements
c) remove first *n* / 2 elements from a list of *n* elements
d) read *n* / 2 elements in random order from a list of *n* elements
Answer: d

77) What operation is least efficient in a `LinkedList`?
a) Adding an element in a position that has already been located.
b) Linear traversal step.
c) Removing an element when the element's position has already been located.
d) Random access of an element.
Answer: d

78) You need to write a program to simulate the effect of adding an additional cashier in a supermarket to reduce the length of time customers must wait to check out. Which data structure would be most appropriate to simulate the waiting customers?
a) map
b) stack
c) queue
d) linked list
Answer: c

79) You need to write a program to build and maintain an address book. Since the program must support the possibility of having duplicate names, which data structure would be most appropriate to model this situation?
a) map
b) stack
c) queue
d) linked list
Answer: d

80) You need to write a program to build and maintain a catalog of college courses that are associated with specific majors. Which data structure would be most appropriate to model this situation?
a) map
b) stack
c) queue
d) linked list
Answer: a

81) You need to write a program to manage a waiting list of patrons at a restaurant. Which data structure would be most appropriate to model this situation?
a) map
b) stack
c) queue
d) linked list
Answer: c

82) You intend to use a hash set with your own object class. Which of the following statements is NOT correct?
a) You do not have to do anything additional. You can use the `hashCode` function of the `Object` class.
b) You can create your own function to compute a `hashCode` value.
c) You can override the `hashCode` method in the `Object` class to provide your own `hashCode` method.
d) Your class's `hashCode` method does not need to be compatible with its `equals` method.
Answer: d

83) Which of the following statements about hash functions is NOT correct?
a) A hash function produces a unique integer-valued hash code value for each distinct object.
b) A good hash function will minimize the number of objects that are assigned the same hash code.
c) Using a prime number as a hash multiplier will produce better hash codes.
d) If you supply your own `hashCode` method for a class, it must be compatible with that class's `equals` method.
Answer: a

84) Which of the following statements about stacks is correct?
a) A stack implements first-in, first-out retrieval.
b) A stack implements random retrieval.
c) A stack implements last-in, first-out retrieval.
d) A stack stores elements in sorted order.
Answer: c

85) An Undo feature in a word processor program that allows you to reverse a previously completed command is probably implemented using which structure type?
a) queue
b) linked list
c) stack
d) hash table
Answer: c

86) Which of the following correctly declares a stack that will hold `String` elements?
a) `Stack<String> s = new Stack<String>();`
b) `Stack s = new Stack<String>();`
c) `String s = new Stack<String>();`
d) `String s = new Stack();`
Answer: a

87) Assume that you have declared a stack named `myStack` to hold `String` elements. Which of the following statements will correctly add an element to `myStack`?
a) `myStack.put("apple");`
b) `myStack.addItem("apple");`
c) `myStack.push("apple");`
d) `myStack.insert("apple");`
Answer: c

88) Assume that you have declared a stack named `myStack` to hold `String` elements. Which of the following statements will correctly remove an element from `myStack`?
a) `myStack.remove();`
b) `myStack.get();`
c) `myStack.delete();`
d) `myStack.pop();`
Answer: d

89) Assume that you have declared a stack named `myStack` to hold `String` elements. Which of the following statements will correctly retrieve the top element from `myStack` without removing it?
a) `myStack.peek();`
b) `myStack.get();`
c) `myStack.next();`
d) `myStack.pop();`
Answer: a

90) Consider the code snippet shown below:
```
Stack<String> words1 = new Stack<String>();
Stack<String> words2 = new Stack<String>();
words1.push("abc");
words1.push("def");
words1.push("ghi");
while (!words1.empty())
{
   words2.push(words1.pop());
}
while (!words2.empty())
{
   System.out.print(words2.pop());
}
```
What will be printed when this code is executed?
a) abcdefghi
b) ghiabcdef
c) abcghidef
d) defghiabc
Answer: a

91) Consider the following code snippet:
```
Stack<String> words1 = new Stack<String>();
ArrayList<String> words3 = new ArrayList<String>();
words1.push("abc");
words1.push("def");
words1.push("ghi");
while (!words1.empty())
{
   words3.add(words1.pop());
}
int i = words3.size() - 1;
while (i >= 0)
{
   System.out.print(words3.remove(i));
   i--;
}
```
What will this code print when it is executed?
a) abcdefghi
b) ghiabcdef
c) abcghidef
d) defghiabc
Answer: a

92) Which data structure would best be used for finding a path out of a maze?
a) queue
b) stack
c) list
d) array
Answer: b

93) Which operations from the list data structure could be used to implement the push and pop operations of a stack data structure?
I `addLast`
II `addFirst`
III `removeFirst`
a) I
b) II
c) I and II
d) II and III
Answer: d

94) Consider the following code snippet:

```
Stack<String> stringStack = new Stack<String>();
stringStack.push("ab");
stringStack.push("abc");
stringStack.push("a");
while (stringStack.size() > 0)
{
   System.out.print(stringStack.pop() + ",");
}
```

What output will be produced when this code is executed?

a) ab,abc,a,
b) a,abc,ab,
c) a,ab,abc,
d) abc,ab,a,
Answer: b

95) Print jobs submitted to a printer would probably be stored in which type of data structure?
a) queue
b) linked list
c) stack
d) hash table
Answer: a

96) Assume that you have declared a queue named `myQueue` to hold `String` elements. Which of the following statements will correctly remove an element from `myQueue`?

a) `myQueue.remove();`
b) `myQueue.get();`
c) `myQueue.delete();`
d) `myQueue.pop();`
Answer: a

97) Assume that you have declared a queue named `myQueue` to hold `String` elements. Which of the following statements will correctly insert an element into `myQueue`?

a) `myQueue.insert("apple");`
b) `myQueue.put("apple");`
c) `myQueue.push("apple");`
d) `myQueue.add("apple");`
Answer: d

98) Select an appropriate expression to complete the method below, which is designed to print the element at the bottom of a `Stack` collection. The contents of the original stack are restored before the method terminates. It is safe to assume that the original stack contains at least one element.

```
public static void printBottom(Stack<String> theStack)
{
   Stack<String> anotherStack = new Stack<String>();
   while (theStack.size() > 0)
   {
      anotherStack.push(theStack.pop());
   }
   ____________________________
   while (anotherStack.size() > 0)
   {
      theStack.push(anotherStack.pop());
   }
}
```

a) `System.out.println(theStack.pop());`
b) `System.out.println(theStack.peek());`
c) `System.out.println(anotherStack.peek());`
d) `System.out.println(anotherStack.pop());`
Answer: c

99) Assuming that `names` is a `Queue` of `String` objects, select a statement to complete the code segment below. The code is designed to remove the last element from the queue, which is guaranteed to have at least one element.

```
Queue<String> aQueue = new LinkedList<String>();
while (names.size() > 1)
{
   aQueue.add(names.remove());
}
names.remove();
while (aQueue.size() > 0)
{
   ____________________________
}
```

a) `names.add(aQueue.remove());`
b) `names.add(aQueue.peek());`
c) `aQueue.add(names.remove());`
d) `aQueue.add(names.peek());`
Answer: a

100) Select an appropriate expression to complete the following method, which is designed to return the sum of the two smallest values in the parameter array `numbers`.

```
public static int sumTwoLowestElements(int[] numbers)
{
   PriorityQueue<Integer> values = new PriorityQueue<Integer>();
   for (int num: numbers)
   {
      values.add(num);
   }
   ______________________
}
```

a) `return values.peek() + values.peek();`
b) `return values.peek() + values.remove();`
c) `return values.remove() + values.remove();`
d) `return values.remove() * 2;`
Answer: c

101) Suppose you push integer elements 1,2,3,4 onto a stack in that order. Then pop an element off the stack and add that element to a queue. You repeat that process three more times. In what order will you remove the elements from the queue?
a) 1,2,3,4
b) 1,2,4,3
c) 4,3,2,1
d) 4,3,1,2
Answer: c

102) Which data structure would best be used for keeping track of bank customers waiting for a teller?
a) queue
b) stack
c) list
d) array
Answer: a

103) Suppose we have two `String` objects and treat the characters in each string from beginning to end in the following way: With one string, we push each character on a stack. With the other string, we add each character to a queue. After processing both strings, we then pop one character from the stack and remove one character from the queue, and compare the pair of characters to each other. We do this until the stack and the queue are both empty. What does it mean if all the character pairs match?

a) The strings are the identical.
b) The strings are different.
c) One string is the reverse of the other.
d) We can only conclude the strings are of the same length
Answer: c

104) Consider the following code snippet:
```
Queue<String> stringQueue = new LinkedList<String>();
stringQueue.add("ab");
stringQueue.add("abc");
stringQueue.add("a");
while (stringQueue.size() > 0)
{
   System.out.print(stringQueue.remove() + ",");
}
```
What output will be produced when this code is executed?
a) ab,abc,a,
b) a,abc,ab,
c) a,ab,abc,
d) abc,ab,a,
Answer: a

105) Suppose we create a deque (double-ended queue) data structure. It is basically a queue, with its `addLast` and `removeFirst` operations, but we also add the `addFirst` and `removeLast` operations. Which of the following is best modeled by the deque data structure?
a) A toll booth on a highway.
b) A cross country race.
c) A computer keyboard typing buffer.
d) A Memorial Day parade.
Answer: c

106) Suppose we create a deque (double-ended queue) data structure. It is basically a queue, with its `addLast` and `removeFirst` operations, but we also add the `addFirst` and `removeLast` operations. If a line at the grocery store resembles a deque more than a queue, what is the most likely reason?

I   the cashier is very fast
II the line is very long
III the cashier is very slow

a) I
b) II
c) I and II
d) II and III
Answer: d

107) Which of the following statements about a priority queue structure is correct?
a) It uses a FIFO discipline.
b) New items must be inserted at the end of the queue.
c) Elements must be removed in priority order.
d) It uses a LIFO discipline.
Answer: c

108) Which of the following statements about a priority queue structure is NOT correct?
a) Elements added to a priority queue must belong to a class that implements the `Comparable` interface.
b) New elements can be inserted in any order.
c) The `remove` method is used to remove an element from the priority queue.
d) The `insert` method is used to add a new element to the priority queue.
Answer: d

109) Consider the following code snippet:
```
PriorityQueue<String> stringQueue = new PriorityQueue<String>();
stringQueue.add("ab");
stringQueue.add("abc");
stringQueue.add("a");
while (stringQueue.size() > 0)
{
   System.out.print(stringQueue.remove() + ",");
}
```
What output will be produced when this code is executed?

a) ab,abc,a,
b) a,abc,ab,
c) a,ab,abc,
d) abc,ab,a,
Answer: c

Chapter16: Basic Data Structures

1) What is included in a linked list node?
I   a reference to the next node
II   an array reference
III a data element
a) I
b) II
c) II and III
d) I and III
Answer: d

2) In the textbook implementation, the `Node` class is a private inner class of the `LinkedList` class. Which of the following statements regarding this implementation is NOT correct?
a) The methods of the `LinkedList` class can access the public features of the `Node` class.
b) The methods of the `Node` class can access the public features of the `LinkedList` class.
c) The methods of the `Node` class can be directly accessed by other classes.
d) The `Node` class's instance variables that represent the node element and its next node reference are declared as public.
Answer: c

3) Which Java package contains the `LinkedList` class?
a) `java.lang`
b) `java.util`
c) `java.collections`
d) `java.io`
Answer: b

4) Insert the missing code in the following code fragment. This fragment is intended to add a new node to the head of a linked list:

```
public class LinkedList
{
   . . .
   public void addFirst(Object element)
   {
      Node newNode = new Node(); 1
      newNode.data = element;
      _________ 2
      _________ 3
   }
   . . .
}
```
a)
```
first = newNode;
newNode.next = first;
```
b)
```
newNode.next = first;
first = newNode;
```
c)
```
first = newNode.next;
newNode.next = first;
```
d)
```
first = newNode.next;
newNode = first;
```
Answer: b

5) Insert the missing code in the following code fragment. This fragment is intended to remove a node from the head of a linked list:
```
public class LinkedList
{
   . . .
   public Object removeFirst()
   {
      if (first == null) { throw new NoSuchElementException(); }
      Object element = first.data;
      ________________
      ________________
   }
   . . .
}
```
a)
```
first = first.next;
return element;
```
b)
```
first.next = first;
return element;
```
c)
```
first = element.next;
return element;
```
d)
```
first = element.next;
return null;
```
Answer: a

6) Insert the missing code in the following code fragment. This fragment is intended to remove a node from the head of a linked list:
```
public class LinkedList
{
   . . .
   public Object removeFirst()
   {
      if (first == null) { ________________ }
      Object element = first.data;
      first = first.next;
      return element;
   }
   . . .
}
```
a) `throw new NoSuchElementException();`
b) `throw new IllegalStateException();`
c) `throw new NullPointerException();`
d) `throw new IllegalArgumentException();`
Answer: a

7) Which of the following statements about removing a node from a linked list is correct?

a) A node's data is discarded when it is removed from the linked list, and its memory space is immediately reclaimed.
b) A node's data is returned to the program when the node is removed from the linked list, and its memory space is immediately reclaimed.
c) A node's data is discarded when it is removed from the linked list, and its memory space is reclaimed later by the garbage collector.
d) A node's data is returned to the program when the node is removed from the linked list, and its memory space is reclaimed later by the garbage collector.
Answer: d

8) In the textbook implementation, the `LinkedListIterator` class is a private inner class of the `LinkedList` class. Which of the following statements regarding this implementation is NOT correct?
a) The methods of the `LinkedList` class can access the public features of the `LinkedListIterator` class.
b) The methods of the `LinkedListIterator` class can access the public features of the `LinkedList` class.
c) The methods of the `LinkedListIterator` class can be directly accessed by other classes.
d) Clients of the `LinkedListClass` do not know the name of the `LinkedListIterator` class.
Answer: c

9) The linked list iterator described in the textbook maintains a reference to the last visited node, called `position`, and a reference to the last node before that, called `previous`. Which of the following statements is NOT correct regarding advancing this iterator?

a) If another node exists, when the iterator is to be advanced, the `position` reference must be updated to `position.next`.
b) If the iterator currently points before the first element of the list, when the iterator is to be advanced, `position` must be set to point to the first node in the linked list.
c) If another node exists, when the iterator is to be advanced, the `previous` reference must be updated to point to the current location of the iterator
d) If the iterator currently points before the first element of the list, when the iterator is to be advanced, the `position` reference must be set to `position.next` and the `previous` reference must be set to point to the first node in the linked list.
Answer: d

10) Which of the following statements about a linked list iterator is NOT correct?

a) The iterator is at the end of the list if the linked list's `first` node reference is null.
b) The iterator is at the end of the list if the `position.next` reference is null.
c) The iterator is at the end of the list if the `position` reference is null.
d) The list is empty if the linked list's `first` node reference is null.
Answer: c

11) Which of the following statements about a linked list and its iterator is NOT correct?
a) The list is empty if the linked list's `first` node reference is null.
b) The iterator is at the end of the list if the `position.next` reference is null.
c) The iterator is at the beginning of the list if the `previous` reference is null.
d) The iterator is at the first node of the list if its `position` reference is null.
Answer: d
12) Using the textbook's implementation of a singly linked list and linked list iterator, the following steps are required to remove a node from the middle of a linked list. Place these steps into the order in which they should be performed.

I The preceding node's next reference must be updated to skip the removed node.
II The iterator's position reference must be set to the previous reference.
III The previous reference must be checked to see if it is equal to the position reference.
a) III, I, II
b) I, III, II
c) II, I, III
d) III, II, I
Answer: a

13) When using the textbook’s implementation of a singly linked list to remove an element in the middle of the list, why it is necessary to check whether the `previous` reference equals the `position` reference?

a) If `previous` equals `position`, the action does not follow a call to `next`.
b) If `previous` equals `position` and an attempt is made to remove the node, the iterator would have to start at the beginning of the list to rebuild the links.
c) If `previous` equals `position`, the iterator is at the beginning of the list and does not point to a valid node.
d) If `previous` equals `position`, the iterator is at the end of the list and does not point to a valid node.
Answer: a

14) Using the textbook's implementation of a linked list, which of the following statements about adding a node to the middle of a linked list is correct?
a) The new node will be added before the last visited node.
b) The `position.next` reference will be updated to point to the new node.
c) The `remove` method can be called immediately before or after adding the new node.
d) The `previous` reference must be updated when adding the new node.
Answer: b

15) Consider the following code snippet:
```
LinkedList<String> words = new LinkedList<String>();
words.addFirst("123");
words.addLast("456");
words.addFirst("789");
System.out.print(words.removeLast());
System.out.print(words.removeFirst());
System.out.print(words.removeLast());
```

What does this code print?
a) `123456789`
b) `789123456`
c) `123789456`
d) `456789123`
Answer: d

16) Consider the following code snippet:
```
LinkedList<String> words = new LinkedList<String>();
words.addFirst("xyz");
words.addLast("jkl");
words.addLast("def");
System.out.print(words.removeFirst());
System.out.print(words.removeLast());
System.out.print(words.removeLast());
```

What does this code print?
a) `xyzjkldef`
b) `defxyzjkl`
c) `xyzdefjkl`
d) `defjklxyz`
Answer: c

17) Using the textbook's implementation of a linked list, which of the following statements about changing the data stored in a previously visited element is correct?
a) The node that will be updated is the most recently visited node.
b) The `position.next` reference must be updated when the data is updated.
c) The `set` method can be called again immediately after calling the `add` method.
d) The `previous` reference must be updated when the node is updated.
Answer: a

18) Using the textbook's implementation of a linked list, which of the following statements about managing nodes within a linked list using an iterator is correct?
a) The node that will be removed is the node pointed to by the `position.next` reference.
b) The `set` method can be called immediately after adding a new node using the `add` method.
c) The `set` method can be called immediately after removing an existing node using the `remove` method.
d) The `position` reference must be updated when a new node is added.
Answer: d

19) Assume that the linked list implementation includes a reference to the last node as well as to the first node. Which of the following statements about the efficiency of the linked list is correct?

a) Adding an element to the middle of the linked list at the current position of the iterator is $O(n)$.
b) Removing an element other than the last element from the linked list at the current position of the iterator is $O(n)$.
c) Accessing an element in the linked list using an iterator is $O(n)$.
d) Adding an element to the end of the linked list is $O(1)$.
Answer: d

20) Assume that the linked list implementation includes a reference to the last node as well as to the first node. Which of the following statements about the efficiency of a singly linked list is NOT correct?
a) Adding an element to the middle of a linked list using an iterator is $O(1)$.
b) Removing the last element from a linked list using an iterator is $O(1)$.
c) Accessing an element in a linked list using an iterator is $O(n)$.
d) Adding an element to the end of a linked list is $O(1)$.
Answer: b

21) Which of the following operations is least efficient in a `LinkedList`?
a) adding an element in a position that has already been located
b) linear traversal step
c) removing an element when the element's position has already been located
d) random access of an element
Answer: d

22) Which of the following algorithms would be efficiently executed on a `LinkedList`?
a) tracking paths in a maze
b) binary search
c) remove first n / 2 elements from a list of n elements
d) read n / 2 elements in random order from a list of n elements
Answer: c

23) What type of access does the use of an iterator with a `LinkedList` provide for its elements?
a) sequential
b) semi-random
c) random
d) sorted
Answer: a

24) Adding or removing an element at an arbitrary iterator position in a singly linked list of length n takes ____ time.
a) $O(n)$
b) $O(\log n)$
c) $O(1)$
d) $O(n^2)$
Answer: c

25) If we want a create a doubly-linked list data structure so that we can move from a node to the next node as well as to a previous node, we need to add a `previous` reference to the `Node` class. Each such `DNode` (doubly-linked Node) will have a data portion and two `DNode` references, `next` and `previous`. How many references need to be updated when we remove a node from the middle of such a list? Consider the neighboring nodes.
a) 1 b) 2
c) 3 d) 4
Answer: b

26) If we want a create a doubly-linked list data structure so that we can move from a node to the next node as well as to a previous node we need to add a `previous` reference to the `Node` class. Each such `DNode` (doubly-linked Node) will have a data portion and two `DNode` references, `next` and `previous`. How many references need to be updated when we remove a node from the beginning of a list with many nodes? Consider the `first` reference and neighboring node(s).
a) 1 b) 2
c) 3 d) 4
Answer: b

27) In a linked list data structure, when does the reference to the first node need to be updated?
I inserting into an empty list
II deleting from a list with one node
III deleting an inner node
a) I
b) II
c) I and II
d) III
Answer: c

28) Suppose we maintain a linked list of length n in sorted order. What would be the big-Oh notation for the `add` operation?
a) $O(1)$
b) $O(n)$
c) $O(n \log_2 n)$
d) $O(n^2)$
Answer: b

29) Suppose we maintain two linked lists of length n in sorted order. What would be the big-Oh notation for the creating a third list, which included only elements common to both lists?
a) $O(1)$
b) $O(n)$
c) $O(n \log_2 n)$
d) $O(n^2)$
Answer: b

30) Suppose we maintain two linked lists of length n in random element order. What would be the big-Oh notation for the creating a third list that includes only elements common to both lists, without sorting the first two lists?
a) $O(1)$
b) $O(n)$
c) $O(n \log_2 n)$
d) $O(n^2)$
Answer: d

31) Suppose we maintain a linked list of length n in random element order. What would be the big-Oh notation for an algorithm that prints each list element and the number of times it occurs in the list (without sorting the list)?
a) $O(1)$
b) $O(n)$
c) $O(n \log_2 n)$
d) $O(n^2)$
Answer: d

32) Suppose we maintain a linked list of length n in random element order. What would be the big-Oh notation for printing out those elements which occur exactly once in the list (without sorting the list)?
a) $O(1)$
b) $O(n)$
c) $O(n\log_2 n)$
d) $O(n^2)$
Answer: d

33) Suppose we maintain a linked list of length n in sorted order. What would be the big-Oh notation for printing out those elements that occur exactly once in the list?
a) $O(1)$
b) $O(n)$
c) $O(n \log_2 n)$

d) $O(n^2)$
Answer: b

34) A doubly-linked list requires that each node maintain two references, one to the next node and one to the previous node. Which of the following statements about a doubly-linked list is NOT correct?
a) If a node's next reference is null, it is at the end of the list.
b) To remove a node in the middle of the list, the previous node's next reference must be updated.
c) To add a node in the middle of the list, you must update the next reference of the node after which the new node will be added.
d) To remove a node in the middle of the list, the previous node's previous reference must be updated.
Answer: d

35) Which of the following actions must be taken to remove a node X from the middle of a doubly-linked list?
I Update the next reference in the node before X
II Update the previous reference in the node after X
III Update the list's first reference
a) I
b) II
c) I and II
d) II and III
Answer: c

36) Which of the following actions must be taken to add a node X into the middle of a doubly-linked list?

I Update the next reference in the node before the position where X will be placed
II Update the previous reference in the node after the position where X will be placed
III Update the list's first reference
a) I
b) II
c) I and II
d) II and III
Answer: c

37) Which of the following actions must be taken to add a node X at the beginning of a doubly-linked list?

I Update the next reference in the node before the position where X will be placed
II Update the previous reference in the node after the position where X will be placed
III Update the list's first reference
a) I
b) II
c) I and II
d) II and III
Answer: d

38) Which of the following actions must be taken to add a node X at the end of a doubly-linked list?

I Update the next reference in the node before the position where X will be placed
II Update the previous reference in the node before the position where X will be placed
III Update the list's last reference
a) I
b) II
c) I and II
d) I and III
Answer: d

39) Using the textbook's implementation of a linked list, what is the purpose of declaring the `Node` class to be a `static` inner class?
a) To create an outer-class reference.
b) To create a `this` reference to itself.
c) To prevent storage of an outer-class reference that is not needed.
d) To create an outer-class reference that is needed.
Answer: c

40) Using the textbook's implementation of a linked list, why is the `LinkedListIterator` inner class NOT declared as an inner `static` class?
a) Because the `LinkedList` class must have access to the instance variables of the `LinkedListIterator` class.
b) Because the `Node` class must have access to the instance variables of the `LinkedListIterator` class.
c) Because the `LinkedListIterator` class must have access to `Node` class instance variables.
d) Because the `LinkedListIterator` class must have access to `LinkedList` class instance variables.
Answer: d

41) What is never present in a `static` inner class?
a) an outer-class reference.
b) the `this` reference to itself.
c) `static` properties.
d) `static` methods.
Answer: a

42) Given the partial `LinkedList` class declaration below, select a statement to complete the `printFirst` method, which is designed to display the contents of the first list element.

```
public class LinkedList
{
   class Node
   {
      public Object data;
      public Node next;
   }

   private Node first;
   . . .
   public void printFirst()
   {
      _____________________________
   }
}
```

a) `System.out.println(first);`
b) `System.out.println(first.data);`
c) `System.out.println(first.next);`
d) `System.out.println(Node.data);`
Answer: b

43) Given the partial `LinkedList` class declaration below, select an expression to complete the `empty` method, which is designed to return true if the list contains no elements.

```
public class LinkedList
{
   class Node
   {
      public Object data;
      public Node next;
   }

   private Node first;
   . . .
   public boolean empty()
   {
      return ________________________ ;
   }
}
```

a) `first != null`
b) `first == null`
c) `first.data == null`
d) `first.next == null`
Answer: b

44) Given the partial `LinkedList` class declaration below, select a statement to complete the `size` method, which is designed to return the number of list elements.

```
public class LinkedList
{
   class Node
   {
      public Object data;
      public Node next;
   }

   private Node first;
   . . .
   public int size()
   {
      int count = 0;
      Node temp = first;
      while (temp != null)
      {
         count++;
         ______________________
      }
      return count;
   }
}
```

a) `temp = temp.next;`
b) `temp = first.next;`
c) `first = temp.next;`
d) `first = first.next;`
Answer: a

45) Given the partial `LinkedList` and `LinkedListIterator` class declarations below, select an expression to complete the `LinkedList get(index)` method, which returns the element at the position indicated by `index`.

```
public class LinkedList
{
   . . .
   public ListIterator listIterator()
   {
      return new LinkedListIterator();
   }

   class LinkedListIterator implements ListIterator
   {
      private Node position;
      private Node previous;
      private boolean isAfterNext;

      public LinkedListIterator()
      {
         . . .
      }

      public Object next()
      {
         . . .
      }

      public boolean hasNext()
      {
         . . .
      }
   }

   public Object get(int index)
   {
      ListIterator it = listIterator();
      for (int i = 0; i < index; ++i)
      {
         it.next();
      }
      return _________________________ ;
   }
}
```

a) `it.next()`
b) `it.previous`
c) `it.position`
d) `it.next().data`

Answer: a

46) Given the partial `ArrayList` class declaration below, select an expression to complete the `empty` method, which is designed to return true if the list contains no elements.

```
public class ArrayList
{
   private Object[] elements;
   private int currentSize;

   public ArrayList()
   {
      final int INITIAL_SIZE = 10;
      elements = new Object[INITIAL_SIZE];
      currentSize = 0;
   }

   public boolean empty()
   {
      return ________________________ ;
   }
}
```

a) `elements.length == 0`
b) `elements.currentSize == 0`
c) `elements[0] == null`
d) `currentSize == 0`
Answer: d

47) Given the partial `ArrayList` class declaration below, select an expression to complete the contains method.

```
public class ArrayList
{
   private Object[] elements;
   private int currentSize;

   public ArrayList()
   {
      final int INITIAL_SIZE = 10;
      elements = new Object[INITIAL_SIZE];
      currentSize = 0;
   }

   public boolean contains(Object item)
   {
      for (int i = 0; ________________ ; i++)
      {
         if (elements[i].equals(item))
         {
            return true;
         }
      }
      return false;
   }
   ...
}
```

a) `i < currentSize`
b) `i <= currentSize`
c) `i < elements.length`
d) `i <= elements.length`
Answer: a

48) An array list maintains a reference to an array of elements called a ____.

a) buffer
b) tree map
c) hash table
d) bucket
Answer: a

49) Reading or writing an array list element at an arbitrary index takes ____ time.

a) $O(\log(n))$
b) $O(n)$
c) $O(n^2)$
d) $O(1)$
Answer: d

50) Suppose we maintain an array `A` of n `int` values as follows:
`A[0] < A[1] < . . . < A[i] > A[i + 1] > A[i + 2] > . . . > A[n - 1]`

The ith element is the maximum in the array. What would be the lowest big-Oh notation for finding that element? Consider a variation of the binary search.

a) $O(1)$
b) $O(\log_2 n)$
c) $O(n)$
d) $O(n^2)$
Answer: b

51) What feature of the `ArrayList` class makes it much better for a binary search than the `LinkedList` class?
a) indexing
b) has no link references
c) it is a generic class
d) it is an abstract class
Answer: a

52) Array lists and linked lists both have the same ____.
a) add/remove efficiency
b) concrete implementations
c) random element access efficiency
d) linear traversal step efficiency
Answer: d

53) Adding or removing an arbitrary element in the middle of an array list takes ____ time.

a) $O(n)$
b) $O(1)$
c) $O(\log(n))$
d) $O(n^2)$
Answer: a

54) On average, how many elements of an array list of size n need to be moved when an element is removed?
a) n
b) n^2
c) $2n$
d) $n / 2$
Answer: d

55) On average, how many elements of an array list of size n need to be moved when an element is added?
a) n
b) n^2
c) $2n$
d) $n / 2$
Answer: d

56) If the current size of an array list is less than the length of the buffer, adding an element at the end of an array list takes ____ time.
a) $O(n)$
b) $O(1)$
c) $O(\log(n))$
d) $O(n^2)$
Answer: b

57) When the buffer for an array list must be grown, a single reallocation operation takes ____ time.
a) $O(n)$
b) $O(1)$
c) $O(\log(n))$
d) $O(n^2)$
Answer: a

58) When considering the reallocation operation for a list whose buffer is full, on average it will take ____ time.
a) $O(n)$
b) $O(1)$
c) $O(1)+$
d) $O(n^2)$
Answer: c

59) Which of the following statements about array list and doubly-linked list operations is correct?
a) It is more efficient to add an element in the middle of an array list than a doubly-linked list.
b) It is more efficient to add an element to the beginning of an array list than a doubly-linked list.
c) It is more efficient to remove an element in the middle of an array list than a doubly-linked list.
d) It is more efficient to retrieve an element in the middle of an array list than a doubly-linked list.
Answer: d

60) Array list operations that were studied included adding/removing an element at the end or in the middle, and retrieving the *k*th element. Which of the following statements about these array list operations is correct?

a) The most expensive operation of an array list is to add an element at the end.
b) The most expensive operation of an array list is to remove an element at the end.
c) The most expensive operation of an array list is to add an element in the middle.
d) The most expensive operation of an array list is to retrieve an arbitrary element.
Answer: c

61) Array list operations that were studied included adding/removing an element at the end or in the middle, and retrieving the *k*th element. Which of the following statements about these array list operations is correct?

a) The least expensive operation of an array list is to add an element at the end.
b) The least expensive operation of an array list is to remove an element at the end.
c) The least expensive operation of an array list is to add an element in the middle.
d) The least expensive operation of an array list is to retrieve an arbitrary element.
Answer: d

62) Linked list operations that were studied included adding/removing an element at the end or in the middle, and retrieving the *k*th element. If the iterator is currently pointing to the correct location for insertion or removal, which of the following statements about these doubly-linked list operations is correct?

a) The most expensive operation of a doubly-linked list is to add an element at the end.
b) The most expensive operation of a doubly-linked list is to remove an element at the end.
c) The most expensive operation of a doubly-linked list is to add an element in the middle.
d) The most expensive operation of a doubly-linked list is to retrieve an arbitrary element.
Answer: d

63) Linked list operations that were studied included adding/removing an element at the end or in the middle, and retrieving the *k*th element. If the iterator is currently pointing to the correct location for insertion or removal, which of the following statements about these doubly-linked list operations is correct?

a) The least expensive operation of a doubly-linked list is to add an element at the end.
b) The least expensive operation of a doubly-linked list is to retrieve an arbitrary element.
c) The least expensive operation of a doubly-linked list is to add an element in the middle.
d) All of these operations have the same time cost.
Answer: c

64) Which operations from the array list data structure could be used in the implementation of the push and pop operations of a stack data structure?
I `addLast`
II `addFirst`
III `removeFirst`
a) I
b) II
c) I and II
d) II and III
Answer: d
65) Which of the following operations from the array list data structure could be used in the implementation of the push and pop operations of a stack data structure?
I `addLast`
II `addFirst`
III `removeLast`
a) I b) II
c) I and III d) II and III
Answer: b

66) Complete the following code, which is intended to add an element to the top of a stack implemented as a linked list.

```
Node newNode = new Node();
newNode.data = element;
_________________
_________________
```

a)
```
first = newNode;
newNode.next = first;
```
b)
```
newNode.next = first;
first = newNode;
```
c)
```
newNode.previous = first;
first.next = newNode;
```
d)
```
first = newNode;
newNode.previous = first;
```
Answer: b

67) A stack can be implemented as a sequence of nodes in a linked list or an array list. Which of the following statements about this is correct?
a) If implementing the stack as a linked list, the least expensive approach is to add and remove elements at the end.
b) If implementing the stack as an array list, the least expensive approach is to add and remove elements at the end.
c) If implementing the stack as an array list, there is no cost difference whether adding and removing elements at the beginning or the end.
d) If implementing the stack as a linked list, there is no cost difference whether adding and removing elements at the beginning or the end.
Answer: b

68) A stack can be implemented as a sequence of nodes in a linked list or an array list. Which of the following statements about this is correct?

a) If implementing the stack as a linked list, it is more expensive to add and remove elements at the end than at the beginning.
b) If implementing the stack as an array list, it is more expensive to add and remove elements at the end than at the beginning.
c) If implementing the stack as an array list, there is no cost difference whether adding and removing elements at the beginning or the end.
d) If implementing the stack as a linked list, there is no cost difference whether adding and removing elements at the beginning or the end.
Answer: a

69) When implementing a queue as a singly-linked list, which of these statements is correct?

a) For better efficiency, nodes should be added at the back and removed at the front.
b) For better efficiency, nodes should be added at the front and removed at the back.
c) There is no difference in efficiency whether nodes are added at the front and removed at the back, or added at the back and removed at the front.
d) You cannot effectively implement a queue as a singly-linked list.
Answer: a

70) You have implemented a queue as a singly-linked list, adding elements at the end and removing elements at the front. What is the cost of the `add` operation?

a) $O(\log n)$
b) $O(n)$
c) $O(n^2)$
d) $O(1)$
Answer: d

71) You have implemented a queue as a singly-linked list, adding elements at the end and removing elements at the front. What is the cost of the `remove` operation?

a) $O(\log(n))$
b) $O(n)$
c) $O(n^2)$
d) $O(1)$
Answer: d

72) Given the `ArrayStack` class implementation discussed in section 16.3 (partially shown below), select the statements needed to complete the `push` method.

```
public class ArrayStack
{
   private Object[] elements;
   private int currentSize;

   public ArrayStack()
   {
      final int INITIAL_SIZE = 10;
      elements = new Object[INITIAL_SIZE];
      currentSize = 0;
   }

   public void push(Object element)
   {
      growIfNecessary();
      ________________
      ________________
   }
}
```

a)
```
elements[currentSize] = element;
currentSize++;
```
b)
```
currentSize++;
elements[currentSize] = element;
```
c)
```
elements[currentSize - 1] = element;
currentSize++;
```
d)
```
elements[currentSize + 1] = element;
currentSize++;
```
Answer: a

73) Given the `LinkedListStack` class implementation discussed in section 16.3 (partially shown below), select the statement(s) to complete the `peek` method.

```
public class LinkedListStack
{
   private Node first;

   public LinkedListStack()
   {
      first = null;
   }

   public Object peek()
   {
      if (empty())
      {
         throw new NoSuchElementException();
      }
      _____________________________
   }
   ...
}
```
a)
```
Object value = first.data;
first = first.next;
return value;
```
b)
```
first = first.next;
return first.data;
```
c)
```
return first;
```
d)
```
return first.data;
```
Answer: d

74) Given the `LinkedListQueue` class implementation discussed in section 16.3 (partially shown below), select the appropriate statements to complete the `add` method.

```
public class LinkedListQueue
{
   private Node first;
   private Node last;

   public LinkedListQueue()
   {
      first = null;
      last = null;
   }

   public void add(Object newElement)
   {
      Node newNode = new Node();
      newNode.data = newElement;
      newNode.next = null;
      if (last == null)
      {
         first = newNode;
         last = newNode;
      }
      else
      {
         ______________________
         ______________________
      }
   }
   . . .
}
```

a)
```
last.next = newNode.next;
last = newNode;
```
b)
```
last.next = newNode;
last = newNode;
```
c)
```
last = newNode;
last.next = newNode;
```
d)
```
last = newNode;
last.next = newNode.next;
```
Answer: b

75) Given the `HashSet` class implementation discussed in section 16.4 (partially shown below), select the statement needed to complete the `clear` method, which is designed to remove all elements from the set.

```
public class HashSet
{
   private Node[] buckets;
   private int currentSize;

   public HashSet(int bucketsLength)
   {
      buckets = new Node[bucketsLength];
      currentSize = 0;
   }

   public void clear()
   {
      for (int j = 0; j < buckets.length; ++j)
      {
         ___________________________
      }
      currentSize = 0;
   }
   . . .
}
```

a) `buckets[j] = 0;`
b) `buckets[j] = new Node();`

c) `buckets[j] = null;`
d) `buckets[j] = new LinkedList();`
Answer: c

76) Elements in a hash table are said to ____ when they have the same hash code value.
a) be equivalent b) compress
c) collide d) buffer well
Answer: c

77) A hash function is considered good if it ____.
a) does not require compression.
b) detects duplicate elements.
c) results in low integer values.
d) minimizes collisions.
Answer: d

78) Which of the following statements about hash tables is correct?
a) The hash code is used to determine where to store each element.
b) Elements in the hash table are sorted in hash code order.
c) A hash table allows duplicate elements.
d) No two elements of a hash table can have the same hash code.
Answer: a

79) Which of the following statements about hash tables is NOT correct?
a) Each entry in a hash table points to a sequence of nodes whose elements have the same compressed hash code.
b) Elements with the same hash code are stored as nodes in a bucket associated with that hash code.
c) A compressed hash code is used as an array index into the hash table.
d) All elements of a hash table must be searched sequentially to determine if an element already exists.
Answer: d

80) Assume that you have a hash table in which there are few or no collisions. What is the time required to locate an element in this hash table?
a) $O(\log n)$ b) $O(n)$
c) $O(n^2)$ d) $O(1)$
Answer: d

81) In the separate chaining technique for handling collisions in a hash table, _____.
a) colliding elements are stored in a nested hash table.
b) colliding elements are placed in empty locations in the hash table.
c) colliding elements are stored in linked lists associated with the hash code.
d) the hash code is compressed to obtain unique hash codes.
Answer: c

82) In the open addressing technique for handling collisions in a hash table, _____.
a) colliding elements are stored in a nested hash table.
b) colliding elements are placed in empty locations in the hash table.
c) colliding elements are stored in linked lists associated with the hash code.
d) the hash code is compressed to obtain unique hash codes.
Answer: b

83) Complete the following code snippet, which is intended to compress a hash code to become a valid array index:
```
int h = x.hashCode();
if (h < 0) { h = -h; }
_________________
```
a) `position = arrayLength % h;`
b) `position = arrayLength / h;`
c) `position = h / arrayLength;`
d) `position = h % arrayLength;`
Answer: d
84) Complete the following code snippet, which is intended to compress a hash code to become a valid array index:
```
________________________
if (h < 0) { h = -h; }
position = h % arrayLength;
```
a) `double h = x.hashCode();`
b) `double h = x.getHashCode();`
c) `int h = x.hashCode();`
d) `int h = x.getHashCode();`
Answer: c

85) Why is it not typical to use the `hashCode` method result directly as an index for array storage?
I because the `hashcode` method returns a `double`
II the values are potentially very large
III the values are not type `int`
a) I
b) I and II
c) I and III
d) II
Answer: d

86) Consider the following code snippet, which computes `h`, the array index for storing `x` in a hash table.

```
int h = x.hashCode();
if (h < 0) { h = -h; }
h = h % size;
```

What does `size` correspond to?
a) The size of the hash table.
b) The number of elements to be stored.
c) The number of collisions.
d) The index of an empty hash table slot.
Answer: a

87) What technique is used to store elements that hash to the same location?
a) colliding
b) bucketing
c) deletion
d) sharing
Answer: b

88) Assume that you have a hash table in which there are few or no collisions. What is the time required to add a new element to this hash table?
a) $O(\log(n))$
b) $O(n)$
c) $O(n^2)$
d) $O(1)$
Answer: d

89) Assume that you have a hash table in which there are few or no collisions. What is the time required to remove an element from this hash table?
a) $O(n)$
b) $O(n^2)$
c) $O(1)$
d) $O(\log (n))$
Answer: c

90) Which of the following statements about adding an element to a hash table is NOT correct?
a) Add the new element at the beginning of the node sequence in the bucket referenced by the hash code.
b) Check the elements in the bucket to determine if the new element already exists.
c) If the element matches another element in the bucket referenced by the hash code, add the new element to that bucket.
d) To add an element, its compressed hash code must be computed.
Answer: c

91) If your `hashCode` function returns a number anywhere in the hash table with equal probability, what is the likely result?
a) Some objects will be impossible to find.
b) The number of collisions will be high.
c) The `get` method will run at $O(n)$ complexity.
d) The `get` method will run at $O(1)$ complexity.
Answer: d

92) Which hash table method(s) will make use of the `equals` method?
I `put`
II `get`
III `contains`
a) I
b) I and II
c) III
d) I, II and III
Answer: d

93) Which of the following statements about using iterators with hash tables is NOT correct?
a) The iterator must track its position within a node chain in a bucket.
b) The iterator must skip over empty buckets.
c) Two iterators are required: one to traverse the buckets, and another to traverse the nodes within a bucket.
d) The iterator must track the bucket number.
Answer: c

94) What is the time required to iterate over all elements in a hash table of size n?
a) $O(\log(n))$
b) $O(n)$
c) $O(n^2)$
d) $O(1)$
Answer: b

95) Assume that you have a hash table in which there are an average number of collisions. What is the time required to remove an element from this hash table?
a) $O(n)$
b) $O(n^2)$
c) $O(1)$
d) $O(1)+$
Answer: d

96) Assume that you have a hash table in which there are an average number of collisions. What is the time required to find an element in this hash table?
a) $O(n)$
b) $O(n^2)$
c) $O(1)$
d) $O(1)+$
Answer: c

97) Assume that you have a hash table in which there are an average number of collisions. What is the time required to add an element to this hash table?

a) $O(n)$
b) $O(n^2)$
c) $O(1)$
d) $O(1)+$
Answer: d

98) Complete the following code, which is intended to add an element to a hash table. Assume that the computed and compressed hash code is stored in the variable `h`.

```
Node newNode = new Node();
newNode.data = x;

_____________________
_____________________
```

a)
```
newNode.next = buckets[h + 1];
buckets[h] = newNode;
```
b)
```
newNode.next = buckets[h];
buckets[h + 1] = newNode;
```
c)
```
newNode.next = buckets[h];
buckets[h - 1] = newNode;
```
d)
```
newNode.next = buckets[h];
buckets[h] = newNode;
```
Answer: d

99) What type of access does the use of an iterator provide for the elements of a bucket in a hash table?

a) sequential
b) semi-random
c) random
d) sorted
Answer: a

100) The advantage of using the open addressing technique over the separate chaining technique for handling collisions in a hash table is that open addressing ____.
a) allows for faster addition of elements
b) allows for faster retrieval of elements
c) is simpler in implementation
d) requires less memory storage for the hash table
Answer: d

101) The ___ technique for handling collisions in a hash table with open addressing attempts to place colliding elements at the next available location in the hash table.
a) sequential placement
b) sequential probing
c) linear placement
d) linear probing
Answer: d

102) Which statement about handling collisions in a hash table using the open addressing technique is correct?
a) A colliding element will always be contiguous to the location in which it would normally be placed.
b) To find an element, you must search from the hash code location until a match or an element with a different hash code is found.
c) To remove an element, you simply empty the slot at which you find it.
d) There may be some elements with different hash codes that lie on the same probing sequence.
Answer: d

103) Which of the following statements about handling collisions in a hash table using the sequential chaining and open addressing techniques is NOT correct?
a) The implementation of sequential chaining technique is simpler than the implementation of the open addressing technique.
b) The sequential chaining technique is simpler to understand than the open addressing technique.
c) The sequential chaining technique requires the storage of links while open addressing does not.
d) The sequential chaining technique requires less memory use than open addressing.
Answer: d

104) Which statement about handling collisions in a hash table using the open addressing technique is NOT correct?
a) A colliding element may not be contiguous to the location in which it would normally be placed.
b) To find an element, you must search from the hash code location until a match or an empty slot is found.
c) To remove an element, you simply empty the slot at which you find it.
d) There may be some elements with different hash codes that lie on the same probing sequence.
Answer: c