1. How many lines of the following program contain compilation errors?

```
package theater;
class Cinema {
   private String name;
   public Cinema(String name) {this.name = name;}
}
public class Movie extends Cinema {
   public Movie(String movie) {}
   public static void main(String[] showing) {
      System.out.print(new Movie("Another Trilogy").name);
   }
}
```

   A. None
   B. One
   C. Two
   D. Three

2. Which modifier can be applied to an `abstract` interface method?
   A. `protected`
   B. `static`
   C. `final`
   D. `public`

3. What is the output of the following application?

```
package radio;
public class Song {
   public void playMusic() {
      System.out.print("Play!");
   }
   private static int playMusic() {
      System.out.print("Music!");
   }
   public static void main(String[] tracks) {
      new Song().playMusic();
   }
}
```

   A. `Play!`
   B. `Music!`
   C. The code does not compile.
   D. The code compiles but the answer cannot be determined until runtime.

4. Which of the following statements about inheritance is true?

   A. Inheritance allows objects to access commonly used attributes and methods.

   B. Inheritance always leads to simpler code.

   C. All primitives and objects inherit a set of methods.

   D. Inheritance allows you to write methods that reference themselves.

5. Given the class declaration below, which value cannot be inserted into the blank line that would allow the code to compile?

```
package mammal;
interface Pet {}
public class Canine implements Pet {
   public _____ getDoggy() {
      return this;
   }
}
```

   A. `Class`

   B. `Pet`

   C. `Canine`

   D. `Object`

6. Imagine you are working with another team to build an application. You are developing code that uses a class that the other team has not finished writing yet. Which element of Java would best facilitate this development, allowing easy integration once the other team's code is complete?

   A. An abstract class

   B. An interface

   C. `static` methods

   D. An access modifier

7. What is the output of the following application?

```
package vehicles;
class Automobile {
   private final String drive() { return "Driving vehicle"; }
}
class Car extends Automobile {
   protected String drive() { return "Driving car"; }
}
public class ElectricCar extends Car {
   public final String drive() { return "Driving electric car"; }
   public static void main(String[] wheels) {
      final Car car = new ElectricCar();
```

```
      System.out.print(car.drive());
   }
}
```

A. `Driving vehicle`

B. `Driving electric car`

C. `Driving car`

D. The code does not compile.

8. Which of the following statements about inheritance is correct?

A. Java does not support multiple inheritance.

B. Java allows multiple inheritance using abstract classes.

C. Java allows multiple inheritance using non-abstract classes.

D. Java allows multiple inheritance using interfaces.

9. How many changes need to be made to the classes below to properly override the `watch()` method?

```
package entertainment;
class Television {
   protected final void watch() {}
}
public class LCD extends Television {
   Object watch() {}
}
```

A. One

B. Two

C. Three

D. None; the code compiles as is.

10. Which of the following statements about overriding a method is incorrect?

A. The return types must be covariant.

B. The access modifier of the method in the child class must be the same or broader than the method in the superclass.

C. A checked exception thrown by the method in the parent class must be thrown by the method in the child class.

D. A checked exception thrown by a method in the child class must be the same or narrower than the exception thrown by the method in the parent class.

11. What is the output of the following application?

```
package machines;
class Computer {
   protected final int process() { return 5; }
```

```
}
public class Laptop extends Computer {
   public final int process() { return 3; }
   public static void main(String[] chips) {
      System.out.print(new Laptop().process());
   }
}
```

A. 5

B. 3

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

12. Given that `FileNotFoundException` is a subclass of `IOException`, what is the output of the following application?

```
package edu;
import java.io.*;
class School {
   public int getNumberOfStudentsPerClassroom(String... students)
         throws IOException {
      return 3;
   }
   public int getNumberOfStudentsPerClassroom() throws IOException {
      return 9;
   }
}
public class HighSchool extends School {
   public int getNumberOfStudentsPerClassroom() throws FileNotFoundException {
      return 2;
   }
   public static void main(String[] students) throws IOException {
      School school = new HighSchool();
      System.out.print(school.getNumberOfStudentsPerClassroom());
   }
}
```

A. 2

B. 3

C. 9

D. The code does not compile.

13. Which modifier can be applied to an interface method?

    A. `protected`

    B. `static`

    C. `private`

    D. `final`

14. What is the output of the following application?

```
package track;
interface Run {
   default void walk() {
      System.out.print("Walking and running!");
   }
}
interface Jog {
   default void walk() {
      System.out.print("Walking and jogging!");
   }
}
public class Sprint implements Run, Jog {
   public void walk() {
      System.out.print("Sprinting!");
   }
   public static void main() {
      new Sprint().walk();
   }
}
```

    A. `Walking and running!`

    B. `Walking and jogging!`

    C. `Sprinting!`

    D. The code does not compile.

15. Which of the following statements about interfaces is not true?

    A. An interface can extend another interface.

    B. An interface can implement another interface.

    C. A class can implement two interfaces.

    D. A class can extend another class.

16. What is the output of the following application?

```
package transport;

class Ship {
   protected int weight = 3;
   private int height = 5;
   public int getWeight() { return weight; }
   public int getHeight() { return height; }
}

public class Rocket extends Ship {
   public int weight = 2;
   public int height = 4;
   public void printDetails() {
      System.out.print(super.getWeight()+","+super.height);
   }
   public static final void main(String[] fuel) {
      new Rocket().printDetails();
   }
}
```

A. `2,5`

B. `3,4`

C. `3,5`

D. The code does not compile.

17. Fill in the blanks: Excluding `default` and `static` methods, a(n) _____ can contain both abstract and concrete methods, while a(n) _____ contains only abstract methods.

A. concrete class, abstract class

B. concrete class, interface

C. interface, abstract class

D. abstract class, interface

18. Which statement about the following class is correct?

```
package shapes;
abstract class Triangle {
   abstract String getDescription();
}
```

```
class RightTriangle extends Triangle {
   protected String getDescription() { return "rt"; } // g1
}
public abstract class IsoscelesRightTriangle extends RightTriangle { // g2
   public String getDescription() { return "irt"; }
   public static void main(String[] edges) {
      final Triangle shape = new IsoscelesRightTriangle(); // g3
      System.out.print(shape.getDescription());
   }
}
```

A. The code does not compile due to line `g1`.

B. The code does not compile due to line `g2`.

C. The code does not compile due to line `g3`.

D. The code compiles and runs without issue.

19. Given that `Short` and `Integer` extend `Number`, what type can be used to fill in the blank in the class below to allow it to compile?

```
package band;

interface Horn { public Integer play(); }
abstract class Woodwind { public Short play() {return 3;} }
public final class Saxophone extends Woodwind implements Horn {
   public _______ play() {
      return null;
   }
}
```

A. `Integer`

B. `Short`

C. `Number`

D. None of the above

20. Fill in the blanks: A class _____ an interface, while a class _____ an `abstract` class.

A. extends, implements

B. extends, extends

C. implements, extends

D. implements, implements

21. What is the output of the following application?

```
package paper;

abstract class Book {
   protected static String material = "papyrus";
   public Book() {}
   public Book(String material) {this.material = material;}
}
public class Encyclopedia extends Book {
   public static String material = "cellulose";
   public Encyclopedia() {super();}
   public String getMaterial() {return super.material;}
   public static void main(String[] pages) {
      System.out.print(new Encyclopedia().getMaterial());
   }
}
```

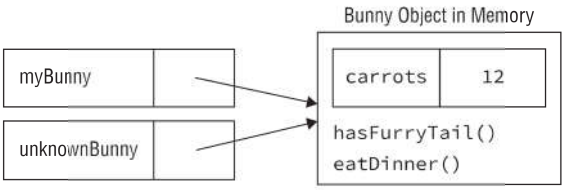
A. `papyrus`

B. `cellulose`

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

22. The following diagram shows two reference variables pointing to the same `Bunny` object in memory. The reference variable `myBunny` is of type `Bunny`, while `unknownBunny` is of an unknown data type. Which statement about the reference variables is not true? For this question, assume the instance methods and variables shown in the diagram are marked `public`.



A. If the `unknownBunny` reference does not have access to the same variables and methods that `myBunny` has access to, it can be explicitly cast to a reference type that does.

B. The data type of `unknownBunny` must be `Bunny` or a subclass of `Bunny`.

C. If the data type of `unknownBunny` is `Bunny`, it has access to all of the same methods and variables as `myBunny`.

D. The data type of `unknownBunny` could be an interface, class, or abstract class.

23. Which of the following modifiers can be applied to an abstract method?

A. `final`

B. `private`

C. `default`

D. `protected`

24. What is the output of the following application?

```
package space;

interface Sphere {
   default String getName() { return "Unknown"; }
}
abstract class Planet {
   abstract String getName();
}
public class Mars extends Sphere implements Planet {
   public Mars() {
      super();
   }
   public String getName() { return "Mars"; }
   public static void main(final String[] probe) {
      System.out.print(((Planet)new Mars()).getName());
   }
}
```

A. `Mars`

B. `Unknown`

C. The code does not compile due to the declaration of `Sphere`.

D. The code does not compile for another reason.

25. Which of the following statements is correct?

A. A reference to a class can be assigned to a subclass reference without an explicit cast.

B. A reference to a class can be assigned to a superclass reference without an explicit cast.

C. A reference to an interface can be assigned to a reference of a class that implements the interface without an explicit cast.

D. A reference to a class that implements an interface can be assigned to an interface reference only with an explicit cast.

26. Of the following four modifiers, choose the one that is not implicitly applied to all interface variables.

A. `final`

B. `abstract`

C. `static`

D. `public`

27. What is the output of the following application?

```
package race;
abstract class Car {
   static { System.out.print("1"); }
   public Car(String name) {
      super();
      System.out.print("2");
   }
   { System.out.print("3"); }
}
public class BlueCar extends Car {
   { System.out.print("4"); }
   public BlueCar() {
      super("blue");
      System.out.print("5");
   }
   public static void main(String[] gears) {
      new BlueCar();
   }
}
```

A. `23451`

B. `12354`

C. `13245`

D. The code does not compile.

28. Fill in the blank: Overloaded and overridden methods always have _____.

A. the same parameter list

B. different return types

C. the same method name

D. covariant return types

29. What is the output of the following application?

```
package sports;
abstract class Ball {
   protected final int size;
   public Ball(int size) {
      this.size = size;
   }
}
```

```
interface Equipment {}
public class SoccerBall extends Ball implements Equipment {
   public SoccerBall() {
      super(5);
   }
   public Ball get() { return this; }
   public static void main(String[] passes) {
      Equipment equipment = (Equipment)(Ball)new SoccerBall().get();
      System.out.print(((SoccerBall)equipment).size);
   }
}
```

A. 5

B. The code does not compile due an invalid cast.

C. The code does not compile for a different reason.

D. The code compiles but throws a `ClassCastException` at runtime.

30. Fill in the blanks: A class that defines an instance variable with the same name as a variable in the parent class is referred to as _____ a variable, while a class that defines a `static` method with the same signature as a `static` method in a parent class is referred to as _____ a method.

A. hiding, overriding

B. overriding, hiding

C. hiding, hiding

D. replacing, overriding

31. Which statement about the following class is correct?

```
package shapes;

abstract class Parallelogram {
   private int getEqualSides() {return 0;}
}
abstract class Rectangle extends Parallelogram {
   public static int getEqualSides() {return 2;} // x1
}
public final class Square extends Rectangle {
   public int getEqualSides() {return 4;} // x2
   public static void main(String[] corners) {
      final Square myFigure = new Square(); // x3
      System.out.print(myFigure.getEqualSides());
   }
}
```

A. The code does not compile due to line `x1`.

B. The code does not compile due to line `x2`.

C. The code does not compile due to line `x3`.

D. The code compiles and runs without issue.

32. What is the output of the following application?

```
package flying;

class Rotorcraft {
   protected final int height = 5;
   abstract int fly();
}
public class Helicopter extends Rotorcraft {
   private int height = 10;
   protected int fly() {
      return super.height;
   }
   public static void main(String[] unused) {
      Helicopter h = (Helicopter)new Rotorcraft();
      System.out.print(h.fly());
   }
}
```

A. `5`

B. `10`

C. The code does not compile.

D. The code compiles but produces a `ClassCastException` at runtime.

33. Fill in the blanks: A class may be assigned to a(n) _____ reference variable automatically but requires an explicit cast when assigned to a(n) _____ reference variable.

A. subclass, outer class

B. superclass, subclass

C. subclass, superclass

D. abstract class, concrete class

34. Fill in the blank: A(n) _____ is the first non-abstract subclass that is required to implement all of the inherited abstract methods.

A. abstract class

B. abstraction

C. concrete class

D. interface

35. How many compiler errors does the following code contain?

```
package animal;
interface CanFly {
   public void fly() {}
}
final class Bird {
   public int fly(int speed) {}
}
public class Eagle extends Bird implements CanFly {
   public void fly() {}
}
```

A. None
B. One
C. Two
D. Three

36. Which of the following is not an attribute common to both abstract classes and interfaces?

A. They both can contain `static` variables.
B. They both can contain `default` methods.
C. They both can contain `static` methods.
D. They both can contain `abstract` methods.

37. What is the output of the following application?

```
package musical;
interface SpeakDialogue { default int talk() { return 7; } }
interface SingMonologue { default int talk() { return 5; } }
public class Performance implements SpeakDialogue, SingMonologue {
   public int talk(String... x) {
      return x.length;
   }
   public static void main(String[] notes) {
      System.out.print(new Performance().talk(notes));
   }
}
```

A. `7`
B. `5`
C. The code does not compile.
D. The code compiles without issue, but the output cannot be determined until runtime.

38. Which of the following is a virtual method?

A. `protected` instance methods

B. `static` methods

C. `private` instance methods

D. `final` instance methods

39. Fill in the blanks: An interface _____ another interface, while a class _____ another class.

A. implements, extends

B. extends, extends

C. implements, implements

D. extends, implements

40. What is the output of the following application?

```
class Math {
   public final double secret = 2;
}
class ComplexMath extends Math {
   public final double secret = 4;
}
public class InfiniteMath extends ComplexMath {
   public final double secret = 8;
   public static void main(String[] numbers) {
      Math math = new InfiniteMath();
      System.out.print(math.secret);
   }
}
```

A. `2`

B. `4`

C. `8`

D. The code does not compile.

41. Given the following method and the fact that `FileNotFoundException` is a subclass of `IOException`, which of the following method signatures is a valid override by a subclass?

```
protected void dance() throws FileNotFoundException {}
```

A. `void dance() throws IOException`

B. `public void dance() throws IOException`

C. `private void dance() throws FileNotFoundException`

D. `public final void dance()`

42. Given the class definitions below, which value, when inserted into the blank line, does not allow the class to compile?

```
public class Canine {}
public class Dog extends Canine {}
public class Wolf extends Canine {}
public final class Husky extends Dog {}
public class Zoologist {
   Canine animal;
   public final void setAnimal(Dog animal) { this.animal = animal; }
   public static void main(String[] furryFriends) {
      new Zoologist().setAnimal(___________);
   }
}
```

A. `new Husky()`

B. `new Dog()`

C. `new Wolf()`

D. `null`

43. Which of the following modifiers cannot be applied to an interface method?

A. `final`

B. `default`

C. `static`

D. `abstract`

44. Which statement about the following application is true?

```
package party;

abstract class House {
   protected abstract Object getSpace();
}
abstract class Room extends House {
   abstract Object getSpace(Object list);
}
abstract public class Ballroom extends House {
   protected abstract Object getSpace();
   public static void main(String[] squareFootage) {
      System.out.print("Let's start the party!");
   }
}
```

A. It compiles and at runtime prints `Let's start the party!`

B. It does not compile for one reason.

C. It does not compile for two reasons.

D. It does not compile for three reasons.

45. Fill in the blanks: _____ methods must have a different list of parameters, while _____ methods must have the exact same return type.

A. Overloaded, overridden

B. Inherited, overridden

C. Overridden, overloaded

D. None of the above

46. Which of the following statements about no-argument constructors is correct?

A. If a parent class does not include a no-argument constructor, a child class cannot declare one.

B. If a parent class does not include a no-argument constructor (nor a default one inserted by the compiler), a child class must contain at least one constructor definition.

C. If a parent class contains a no-argument constructor, a child class must contain a no-argument constructor.

D. If a parent class contains a no-argument constructor, a child class must contain at least one constructor.

47. Fill in the blanks: The _____ determines which attributes exist in memory, while the _____ determines which attributes are accessible by the caller.

A. reference type, signature

B. object type, superclass

C. reference type, object type

D. object type, reference type

48. Given that `Integer` and `Long` are subclasses of `Number`, what type can be used to fill in the blank in the class below to allow it to compile?

```
package orchestra;
interface MusicCreator { public Number play(); }
abstract class StringInstrument { public Long play() {return 3L;} }
public class Violin extends StringInstrument implements MusicCreator {
   public _________ play() {
      return 12;
   }
}
```

A. `Long`

B. `Integer`

C. `Long` or `Integer`

D. `Long` or `Number`

49. Which of the following is the best reason for creating a `default` interface method?

A. Allow interface methods to be inherited.

B. Add backward compatibility to existing interfaces.

C. Give an interface the ability to create concrete methods.

D. Allow an interface to define a method at the class level.

50. Given that `EOFException` is a subclass of `IOException`, what is the output of the following application?

```
package ai;
import java.io.*;
class Machine {
   public boolean turnOn() throws EOFException {return true;}
}
public class Robot extends Machine {
   public boolean turnOn() throws IOException {return false;}
   public static void main(String[] doesNotCompute) throws Exception {
      Machine m = new Robot();
      System.out.print(m.turnOn());
   }
}
```

A. `true`

B. `false`

C. The code does not compile.

D. The code compiles but produces an exception at runtime.