Review Questions

1. What modifiers are implicitly applied to all interface methods? (Choose all that apply)
   A. `protected`
   B. `public`
   C. `static`
   D. `void`
   E. `abstract`
   F. `default`

2. What is the output of the following code?

```
1: class Mammal {
2:   public Mammal(int age) {
3:     System.out.print("Mammal");
4:   }
5: }
6: public class Platypus extends Mammal {
7:   public Platypus() {
8:     System.out.print("Platypus");
9:   }
10:  public static void main(String[] args) {
11:    new Mammal(5);
12:  }
13: }
```

   A. `Platypus`
   B. `Mammal`
   C. `PlatypusMammal`
   D. `MammalPlatypus`
   E. The code will not compile because of line 8.
   F. The code will not compile because of line 11.

3. Which of the following statements can be inserted in the blank line so that the code will compile successfully? (Choose all that apply)

```
public interface CanHop {}
public class Frog implements CanHop {
   public static void main(String[] args) {
     _____ frog = new TurtleFrog();
   }
}
```

```
public class BrazilianHornedFrog extends Frog {}
public class TurtleFrog extends Frog {}
```

A. `Frog`

B. `TurtleFrog`

C. `BrazilianHornedFrog`

D. `CanHop`

E. `Object`

F. `Long`

4. Which statement(s) are correct about the following code? (Choose all that apply)

```
public class Rodent {
  protected static Integer chew() throws Exception {
    System.out.println("Rodent is chewing");
    return 1;
  }
}
public class Beaver extends Rodent {
  public Number chew() throws RuntimeException {
    System.out.println("Beaver is chewing on wood");
    return 2;
  }
}
```

A. It will compile without issue.

B. It fails to compile because the type of the exception the method throws is a subclass of the type of exception the parent method throws.

C. It fails to compile because the return types are not covariant.

D. It fails to compile because the method is `protected` in the parent class and `public` in the subclass.

E. It fails to compile because of a `static` modifier mismatch between the two methods.

5. Which of the following may only be hidden and not overridden? (Choose all that apply)

A. `private` instance methods

B. `protected` instance methods

C. `public` instance methods

D. `static` methods

E. `public` variables

F. `private` variables

6. Choose the correct statement about the following code:

```
1: interface HasExoskeleton {
2:   abstract int getNumberOfSections();
3: }
4: abstract class Insect implements HasExoskeleton {
5:   abstract int getNumberOfLegs();
6: }
7: public class Beetle extends Insect {
8:   int getNumberOfLegs() { return 6; }
9: }
```

   A. It compiles and runs without issue.

   B. The code will not compile because of line 2.

   C. The code will not compile because of line 4.

   D. The code will not compile because of line 7.

   E. It compiles but throws an exception at runtime.

7. Which of the following statements about polymorphism are true? (Choose all that apply)

   A. A reference to an object may be cast to a subclass of the object without an explicit cast.

   B. If a method takes a superclass of three objects, then any of those classes may be passed as a parameter to the method.

   C. A method that takes a parameter with type `java.lang.Object` will take any reference.

   D. All cast exceptions can be detected at compile-time.

   E. By defining a public instance method in the superclass, you guarantee that the specific method will be called in the parent class at runtime.

8. Choose the correct statement about the following code:

```
1: public interface Herbivore {
2:   int amount = 10;
3:   public static void eatGrass();
4:   public int chew() {
5:     return 13;
6:   }
7: }
```

   A. It compiles and runs without issue.

   B. The code will not compile because of line 2.

   C. The code will not compile because of line 3.

   D. The code will not compile because of line 4.

   E. The code will not compile because of lines 2 and 3.

   F. The code will not compile because of lines 3 and 4.

9. Choose the correct statement about the following code:

```
1: public interface CanFly {
2:   void fly();
3: }
4: interface HasWings {
5:   public abstract Object getWindSpan();
6: }
7: abstract class Falcon implements CanFly, HasWings {
8: }
```

   A. It compiles without issue.
   B. The code will not compile because of line 2.
   C. The code will not compile because of line 4.
   D. The code will not compile because of line 5.
   E. The code will not compile because of lines 2 and 5.
   F. The code will not compile because the class `Falcon` doesn't implement the interface methods.

10. Which statements are true for both abstract classes and interfaces? (Choose all that apply)
   A. All methods within them are assumed to be abstract.
   B. Both can contain `public static final` variables.
   C. Both can be extended using the `extend` keyword.
   D. Both can contain default methods.
   E. Both can contain static methods.
   F. Neither can be instantiated directly.
   G. Both inherit `java.lang.Object`.

11. What modifiers are assumed for all interface variables? (Choose all that apply)
   A. `public`
   B. `protected`
   C. `private`
   D. `static`
   E. `final`
   F. `abstract`

12. What is the output of the following code?

```
1: interface Nocturnal {
2:   default boolean isBlind() { return true; }
3: }
4: public class Owl implements Nocturnal {
```

```
5:   public boolean isBlind() { return false; }
6:     public static void main(String[] args) {
7:     Nocturnal nocturnal = (Nocturnal)new Owl();
8:     System.out.println(nocturnal.isBlind());
9:     }
10: }
```

A. `true`

B. `false`

C. The code will not compile because of line 2.

D. The code will not compile because of line 5.

E. The code will not compile because of line 7.

F. The code will not compile because of line 8.

13. What is the output of the following code?

```
1: class Arthropod
2:   public void printName(double input) { System.out
  .print("Arthropod"); }
3: }
4: public class Spider extends Arthropod {
5:   public void printName(int input) { System.out.print("Spider"); }
6:   public static void main(String[] args) {
7:     Spider spider = new Spider();
8:     spider.printName(4);
9:     spider.printName(9.0);
10:   }
11: }
```

A. `SpiderArthropod`

B. `ArthropodSpider`

C. `SpiderSpider`

D. `ArthropodArthropod`

E. The code will not compile because of line 5.

F. The code will not compile because of line 9.

14. Which statements are true about the following code? (Choose all that apply)

```
1: interface HasVocalCords {
2:   public abstract void makeSound();
3: }
4: public interface CanBark extends HasVocalCords {
5:   public void bark();
6: }
```

A. The `CanBark` interface doesn't compile.

B. A class that implements `HasVocalCords` must override the `makeSound()` method.

C. A class that implements `CanBark` inherits both the `makeSound()` and `bark()` methods.

D. A class that implements `CanBark` only inherits the `bark()` method.

E. An interface cannot extend another interface.

15. Which of the following is true about a concrete subclass? (Choose all that apply)

A. A concrete subclass can be declared as `abstract`.

B. A concrete subclass must implement all inherited abstract methods.

C. A concrete subclass must implement all methods defined in an inherited interface.

D. A concrete subclass cannot be marked as `final`.

E. Abstract methods cannot be overridden by a concrete subclass.

16. What is the output of the following code?

```
1: abstract class Reptile {
2:   public final void layEggs() { System.out.println("Reptile laying eggs");
   }
3:     public static void main(String[] args) {
4:     Reptile reptile = new Lizard();
5:     reptile.layEggs();
6:     }
7: }
8: public class Lizard extends Reptile {
9:   public void layEggs() { System.out.println("Lizard laying eggs"); }
10: }
```

A. `Reptile laying eggs`

B. `Lizard laying eggs`

C. The code will not compile because of line 4.

D. The code will not compile because of line 5.

E. The code will not compile because of line 9.

17. What is the output of the following code?

```
1: public abstract class Whale {
2:   public abstract void dive() {};
3:   public static void main(String[] args) {
4:     Whale whale = new Orca();
5:     whale.dive();
6:   }
7: }
```

```
8: class Orca extends Whale {
9:   public void dive(int depth) { System.out.println("Orca diving"); }
10: }
```

A. `Orca diving`

B. The code will not compile because of line 2.

C. The code will not compile because of line 8.

D. The code will not compile because of line 9.

E. The output cannot be determined from the code provided.

18. What is the output of the following code? (Choose all that apply)

```
1: interface Aquatic {
2:   public default int getNumberOfGills(int input) { return 2; }
3: }
4: public class ClownFish implements Aquatic {
5:   public String getNumberOfGills() { return "4"; }
6:   public String getNumberOfGills(int input) { return "6"; }
7:   public static void main(String[] args) {
8:     System.out.println(new ClownFish().getNumberOfGills(-1));
9:   }
10: }
```

A. 2

B. 4

C. 6

D. The code will not compile because of line 5.

E. The code will not compile because of line 6.

F. The code will not compile because of line 8.

19. Which of the following statements can be inserted in the blank so that the code will compile successfully? (Choose all that apply)

```
public class Snake {}
public class Cobra extends Snake {}
public class GardenSnake {}
public class SnakeHandler {
   private Snake snake;
   public void setSnake(Snake snake) { this.snake = snake; }
   public static void main(String[] args) {
     new SnakeHandler().setSnake(__________);
   }
}
```

A. `new Cobra()`

B. `new GardenSnake()`

C. `new Snake()`

D. `new Object()`

E. `new String("Snake")`

F. `null`

20. What is the result of the following code?

```
1: public abstract class Bird {
2:   private void fly() { System.out.println("Bird is flying"); }
3:   public static void main(String[] args) {
4:     Bird bird = new Pelican();
5:     bird.fly();
6:   }
7: }
8: class Pelican extends Bird {
9:   protected void fly() { System.out.println("Pelican is flying"); }
10: }
```

A. `Bird is flying`

B. `Pelican is flying`

C. The code will not compile because of line 4.

D. The code will not compile because of line 5.

E. The code will not compile because of line 9.