Exercise 11 Answers

Inheritance

---

1. (a)

   (b) Use the compiler to see what statements are legal.

   (c) Use the compiler to see what is output.

   (d) When an `Eagle` object is constructed, the constructors of all subclasses of `Eagle` are called, not just the `Eagle` constructor. The `Animal` constructor is called first which sets the `numberOfLegs` to 4, followed by the `Bird` constructor, which resets this value to 2, followed by the `Eagle` constructor, which leaves this value unchanged.

   The reason that the superclass's constructors in the process of creating a `new Eagle(...)` is so that the `Eagle` object will have all its properties properly initialised.

   (e) The statement `a.fly()` does not work because there is no method called `fly` in the `Animal` class. Notice that the first statement `a = b` assigns a `Bird` reference to `a`, so that at run time the object referenced by `a` will be a `Bird` and therefore have a `fly` method.

   In spite of this, the compiler must not allow `Animal` references to call the `fly` method of the `Bird` class, because the compiler is not able to do a run-time analysis of what type of object each reference is referring to. Doing such a run-time analysis of the types is beyond the scope of what the compiler can do.

   (f) The statement `b = a` is not allowed by the compiler because `Animal` objects are not able to call the `fly` method. If we allowed this statement, then this would cause the program to try and call the `fly` method on an `Animal` object, which doesn't make sense.

   This can be stated more clearly using commonsense reasoning about birds and animals. The inheritance of `Bird` from `Animal` expresses the notion that "every bird is an animal". The fact that the compiler allows the assignment `a = b` is in accordance with this. However the assignment `b = a` is in accordance with the idea that "every animal is a bird", which is of course nonsense and must be rejected.

2. Here is the `StarWars.java` source file modified to use inheritance. No changes were needed for the `StarWars` class so it is not shown in the listing below.

```
class SpaceShip {

   // Properties of the class...
   protected int     shields;
   protected int     weapon;
   protected boolean dead;

   // Methods of the class...
   public int getWeapon() {
      return weapon;
   }
   public boolean isDead() {
```

```
      return dead;
   }
   public void hit(int damage) {
      shields = shields - damage;
      if (shields < 0) {
         System.out.println("BOOM!!!");
         dead = true;
      }
   }
}


class XWing extends SpaceShip {

   // Constructor of the class...
   public XWing() {
      shields = 1000;
      weapon  = 10;
   }

}

class Tie extends SpaceShip {

   // Constructor of the class...
   public Tie() {
      shields = 500;
      weapon  = 20;
   }

}
```