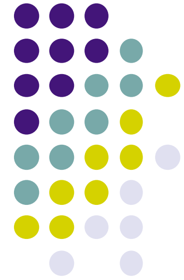
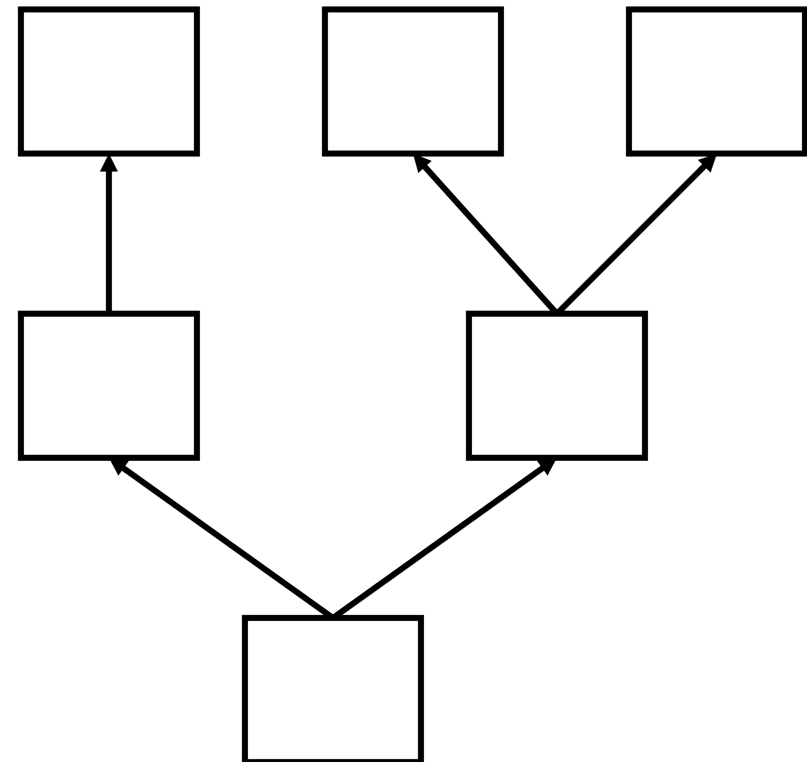
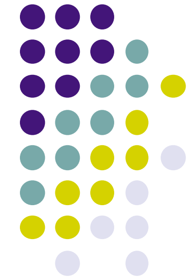
The inheritance hierarchy



- We add an edge between each class and its direct superclasses
  - This gives a directed acyclic graph called the inheritance hierarchy
- We know how to define a class that inherits from one class (single inheritance), but how can a class inherit from more than one (multiple inheritance)?
  - Multiple inheritance is complicated but it can be a powerful tool
- We give a simple example; for much more see the book
  - *Object-oriented Software Construction* by Bertrand Meyer, Prentice-Hall, 1997



Example of multiple inheritance



- Geometric figures

```
class Figure
   meth draw ... end
   ...
end
class Line from Figure
   meth draw ... end
   ...
End
```

- A compound figure is *both* a figure and a linked list
- Multiple inheritance works in this case because the two superclasses are *independent*

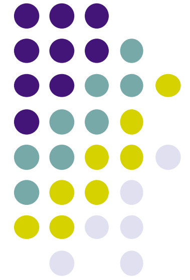
- Linked lists

```
class LinkedList
   meth forall(M)
      ...  % invoke M on all elements
   end
   ...
end
```

- Compound figures

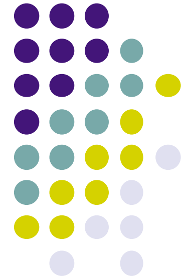
```
class CompoundFigure from
   Figure LinkedList
   meth draw
      {self forall(draw)}
   end
   ...
end
```

Java interfaces and multiple inheritance



- Java only allows single inheritance for classes
  - Multiple inheritance is forbidden, but to keep some of its expressiveness, Java introduces the concept of interface
- An interface is similar to an abstract class with no method implementations
  - The interface gives the method names and their argument types, without the implementation
- Java allows multiple inheritance for interfaces

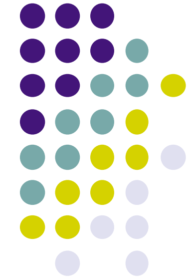
Example of a Java interface

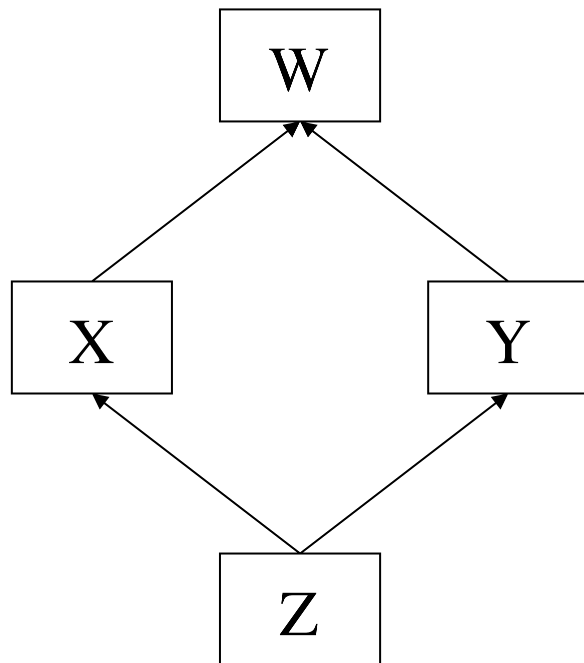


```
interface Lookup {
   Object find(String name);
}

class SimpleLookup implements Lookup {
   private String[] Names;
   private Object[] Values;
   public Object find(String name) {
      for (int i=0; i<Names.length; i++) {
         if (Names[i].equals(name))
            return Values[i];
      }
      return null;
   }
}
```

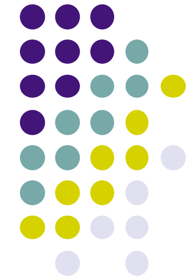
The diamond problem

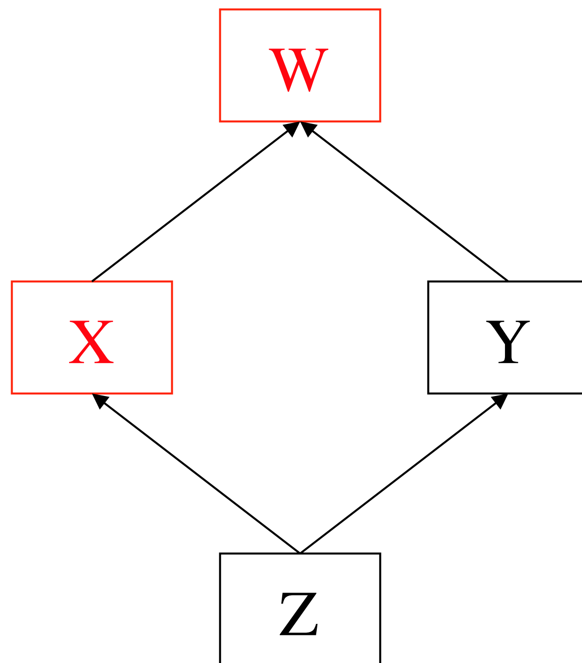




- The diamond problem is a classic problem with multiple inheritance
- When class W has state (attributes), who will initialise W? X or Y or both?
  - There is no simple solution
  - This is one reason why multiple inheritance is not allowed in Java
- Interfaces give a partial solution to this problem

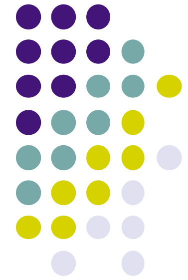
A solution with interfaces

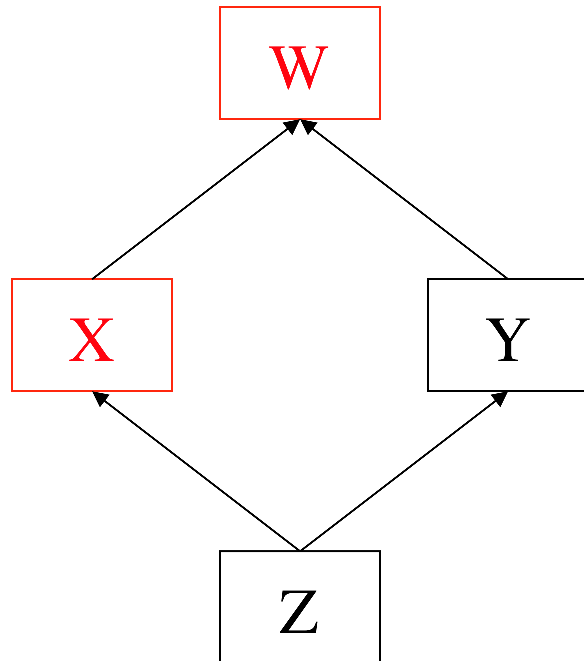




- Interfaces are given in red
- There is no more diamond inheritance: class Z only inherits from class Y
- For an interface, inheritance is just a constraint on the method headers (names and arguments) in the classes
  - Multiple inheritance means more constraints on the method headers
  - An interface contains no code; no code means no diamond problem

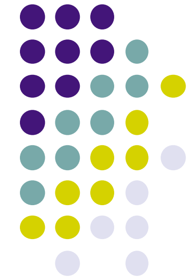
Java syntax for the diamond example

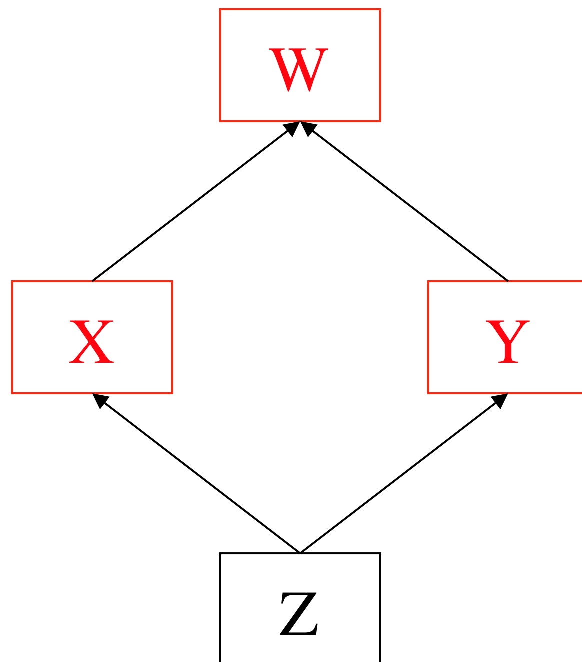





```
interface W { }
interface X extends W { }
class Y implements W { }
class Z extends Y
        implements X { }
```

Another solution for the same example





- In this solution, Z is the only class in the hierarchy
- It has the following syntax:

```
interface W { }
interface X extends W { }
interface Y extends W { }
class Z implements X, Y { }
```

- Are there any other solutions for this example?