1. Although a derived class inherits methods from the base class, it can **change** or ***override*** an inherited method if necessary
   In order to override a method definition, a new definition of the method is simply placed in the class definition

   The access permission of an overridden method can be changed **from private** in the base class **to public** (or some other more permissive access) in the derived class
   qHowever, the access permission of an overridden method **can not** be changed **from public** in the base class **to a more restricted** access permission in the derived class

   When a method is **overridden**, the new method definition given in the derived class has the **exact same number and types of parameters** as in the base class
   qWhen a method in a derived class has a **different signature** from the method in the base class, that is **overloading**

2. If the modifier **`final`** is placed before the definition of a *method*, then that method may not be redefined in a derived class

3. A derived class uses a constructor from the base class to initialize all the data inherited from the base class
   In order to invoke a constructor from the base class, it uses a special syntax:

```
public derivedClass(int p1, int p2, double p3)
{
super(p1, p2); instanceVariable = p3;
}
```

   No double super

   ❑ Often, a no-argument constructor uses **this** to invoke an explicit-value constructor
   ➢ No-argument constructor (invokes explicit-value constructor using **this** and default arguments):

```
public ClassName()
{
this(argument1, argument2);
}
```

4. Since the inherited instance variables should be initialized, and the base class constructor is designed to do that, then an explicit call to **super** should always be used
5.
6. The **instanceof** operator checks if an object is of the type given as its

second argument
Object instanceof ClassName
This will return **true** if **Object** is of type
ClassName, and otherwise return **false**
Note that this means it will return **true** if **Object** is the type of *any descendent class* of **ClassName**

Every object inherits the same **getClass()**
method from the **Object** class
This method is marked **final**, so it cannot be overridden
qAn invocation of **getClass()** on an object returns a representation *only* of the class that was used with **new** to create the object
The results of any two such invocations can be compared with **==** or **!=** to determine whether or not they represent the exact same class
(object1.getClass() == object2.getClass())

7. *Upcasting* is when an object of a derived class is assigned to a variable of a base class (or any ancestor class)

   Downcasting is when a type cast is performed from a base class to a derived class (or from any ancestor class to any descendent class)
   Downcasting has to be done very carefully
   In many cases it doesn't make sense, or is illegal:

8. Abstract
   沒有methos body的method
   較abstract method
   The class that contains an abstract method is called an *abstract class(***It cannot be private)**

```
public abstract double getPay();
public abstract void doIt(int count);
interface
```

   However, *an interface is not a class*
   ØIt is a type that can be satisfied by any class that implements the interface
   ➢ It contains **method headings** and **constant definitions**
   only
   - Any variables defined in an interface must be public, static, and final
   - It contains **no instance variables nor any complete method definitions**
   - **public interface Shape {**
   - **int *color = 1; // => public static final int color = 1;***
   - }

   Multiple inheritance is not allowed in Java
   qInstead, Java's way of approximating multiple inheritance is through interfaces

9.
10.
11.
12. Extends:
    It allows code to be *reused*, without having to copy it into the definitions of the derived classes
    Polymorphism:
    A same operation can behave differently (be implemented by different methods).

13. ***Early binding*** or ***static binding***
    which method is to be called is decided at compile- time
    Overloading*: an invocation can be operated on arguments of more than one type*
    Java uses ***static binding*** *with* ***private****,* ***final****, and* ***static*** *methods*
    - *In the case of* ***private*** *and* ***final*** *methods, late binding would serve no purpose*
    - *However, in the case of a static method invoked using a calling object, it does make a difference*

    Late binding or ***dynamic binding***
    which method is to be called is decided at run- time
    Overriding: a derived class inherits methods from the base class, it can change or override an inherited method
14.
15. Classification: (Class & Object)
    objects with the same attributes and operations are grouped into a class (data abstraction)
    each object is said to be an instance of its class
    e.g. Bicycle object -------> Bicycle class

    Creating instances of a class is called *instantiation*.

    Polymorphism

    A same operation may behave differently on

    different classes.

Abstraction

Focus on the essential, inherent aspects of an entity

and ignore its accidental properties

Encapsulation

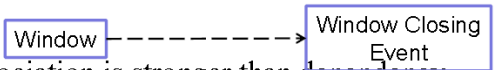
Object orientation separates the external aspects of

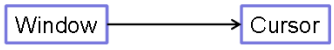
an object accessible to their objects from the

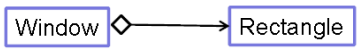
internal implementation details of the object

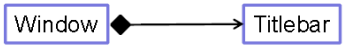
hidden from other objects.

16.

- Dependency is the weakest relationship between classes.
  - Uses-a
  - A transient relationship, that is, doesn't retain a relationship for any real length of time
  - A dependent class briefly interacts with the target class
  - 

- Association is stronger than dependency.
  - One class retains a relationship to another class over an extended period of time
  - Has-a
  - 

- Aggregation is a stronger version of association.
  - **Implies** ownership
  - Owns-a
  - 

- Composition represents a very strong relationship between classes to the point of containment.
  - A whole-part relationship
  - Is-part-of
  - 

- Generalization
  - Is-a
  -