### Loose coupling

1.

public class Demo {  
    public static void main(String[] args) {  
        B b = new B();  
        b.returnA();  
    }  
}  
  
// top level class // class ගනනාවක පාවිච්චි වෙන class එකක්   
class  A {  
    public void getA(){  
        System.*out*.println("giving A");

// top level class එකේ වෙනස්කම් වුනොත් අවුල්.  
    }  
}  
  
// low level class  
class B{  
    public void returnA(){  
   A a = new A();  
        a.getA();

// tight coupling (low level class එකක්, top level class එකක් මත directly depend වෙලා තිබීම.)

    }  
}

2.

public class Demo{  
    public static void main(String[] args) {  
        B b = new B();  
        b.returnA();  
    }  
}  
  
interface SuperA{  // interface එකක් = agreement එකක්   
    void getA();  
}  
  
// top level class  
class  A implements SuperA{ //   
    public void getA(){  
        System.*out*.println("giving A");

// top level class එකේ වෙනස්කම් කරන්න බෑ. (implement වුණු නිසා)  
    }  
}  
  
// low level class  
class B{  
    public void returnA(){  
        // loosely coupling **// run time polymorphism use** වෙන්නෙ

**SuperA a = new A();**  
        a.getA();  
    }  
}

### Dependency Injection

1.

public class D1 {  
    public static void main(String[] args) {  
        Boy b = new Boy();  
        b.cattingWithGirl();  
    }  
}  
  
interface GoodGirl{  
    void chatting();  
}  
  
class Girl implements GoodGirl{  
    @Override  
    public void chatting() {  
        System.*out*.println("Hi");  
    }  
}  
  
class Boy{  
    GoodGirl girl = new Girl(); **//(1) property inject**  
  
    public void cattingWithGirl(){  
        //Loose Coupling Applied  
        girl.chatting();  
    }  
}

2.

public class D2 {  
    public static void main(String[] args) {  
        Boy b = new Boy(new Girl());  
        b.cattingWithGirl();  
    }  
}  
interface GoodGirl{  
    void chatting();  
}  
class Girl implements GoodGirl{  
    @Override  
    public void chatting() {  
        System.*out*.println("Hi");  
    }  
}  
class Boy{  
    GoodGirl girl ;  
    **// (2) constructor injection**  
    Boy(Girl girl){  
        this.girl = girl;  
    }  
  
   public void cattingWithGirl(){  
        //Loose Coupling Applied  
        girl.chatting();  
    }  
}

3.

public class D3 {  
    public static void main(String[] args) {  
        Boy b = new Boy();  
        b.setInject(new Girl());  
        b.cattingWithGirl();  
    }  
}  
  
interface GoodGirl{  
    void chatting();  
}  
  
class Girl implements GoodGirl{  
    @Override  
    public void chatting() {  
        System.*out*.println("Hi");  
    }  
}  
  
class Boy{  
  
    **//(3) Setter method injection**    
    GoodGirl girl ;  
    public void setInject(Girl girl){  
        this.girl = girl;  
    }  
  
    public void cattingWithGirl(){  
        //Loose Coupling Applied  
        girl.chatting();  
    }  
}

4.

public class D4 {  
    public static void main(String[] args) {  
        Boy b = new Boy();  
        b.setInject(new Girl());  
        b.cattingWithGirl();  
    }  
}  
  
interface GoodGirl{  
    void chatting();  
}  
  
class Girl implements GoodGirl{  
    @Override  
    public void chatting() {  
        System.*out*.println("Hi");  
    }  
}

**// (4)  Interface trough injection**  
interface DI{  
    void setInject(Girl girl);  
}  
  
class Boy implements DI{  
    GoodGirl girl;  
  
    @Override  
    public void setInject(Girl girl) {  
        this.girl = girl;  
    }  
  
    public void cattingWithGirl(){  
        //Loose Coupling Applied  
        girl.chatting();  
    }  
}