```
package com.tcfacs.logisim;
interface Node{
   boolean eval();
}

class Switch implements Node{
   boolean state;
   public Switch(boolean state)     { this.state = state; }
   public void toggle()             { this.state = !this.state; }
   public boolean eval()            { return state; }
}

class NOT implements Node {
   Node n;
   public NOT()                     {}
   public NOT(Node n)               { this.setSource(n); }
   public void setSource(Node n)    { this.n=n; }
   public boolean eval()            { return !n.eval(); }
}

class OR implements Node {
   Node a,b;
   public OR()                      {}
   public OR(Node a, Node b)        { this.setA(a); this.setB(b);}
   public void setA(Node n)         { this.a=n;}
   public void setB(Node n)         { this.b=n;}
   public boolean eval()            { return a.eval() | b.eval(); }
}

class AND implements Node {
   Node a,b;
   public AND()                     {}
   public AND(Node a, Node b)       { this.setA(a); this.setB(b);}
   public void setA(Node n)         { this.a=n;}
   public void setB(Node n)         { this.b=n;}
   public boolean eval()            { return a.eval() & b.eval(); }
}
```

```
public class Main {

   static void evaluateXOR(Switch A, Switch B, Node xor){
       System.out.println(A.eval() + " " + B.eval() + " : " + xor.eval());
       A.toggle();
       System.out.println(A.eval() + " " + B.eval() + " : " + xor.eval());
       B.toggle();
       System.out.println(A.eval() + " " + B.eval() + " : " + xor.eval());
       A.toggle();
       System.out.println(A.eval() + " " + B.eval() + " : " + xor.eval());
       B.toggle();
       System.out.println(A.eval() + " " + B.eval() + " : " + xor.eval());
   }

   public static void simpleXOR(){
       Switch A = new Switch(false);
       Switch B = new Switch(false);
       Node xor = new OR(new AND(A, new NOT(B)), new AND(B, new NOT(A)));
       evaluateXOR(A,B,xor);

   }

   public static void dynamicXOR(){

       // declare the objects
       //
       Switch A = new Switch(false);
       Switch B = new Switch(false);
       NOT g1 = new NOT();
       NOT g2 = new NOT();
       AND g3 = new AND();
       AND g4 = new AND();
       OR  g5 = new OR();

       // wire the objects
       //
       g1.setSource(A);
       g2.setSource(B);
       g3.setA(A);
       g3.setB(g2);
       g4.setA(g1);
       g4.setB(B);
       g5.setA(g3);
       g5.setB(g4);

       evaluateXOR(A,B,g5);
   }
```

```
public static void main(String[] args) {

        System.out.println("Static XOR");
        simpleXOR();

        System.out.println("Dynamic XOR");
        dynamicXOR();

    }
}
```