```java
package logique;

public class AND extends OperateurBinaire {

	public AND(Expression exp1, Expression exp2) {
		super(exp1, exp2);
	}

	public boolean evaluate() {
		if (!membres[0].evaluate())
			return false;
		else
			return membres[1].evaluate();
	}

	protected String printOperateur() {
		return " AND ";
	}

}
```



```java
package logique;

public abstract class Constante implements Expression {

}
```

oct. 01, 10 13:59 **Expression.java** Page 1/1

```
package logique;

public interface Expression {

	public boolean evaluate();

	public String print();
}
```

sept. 29, 10 16:53 **FALSE.java** Page 1/1

```
package logique;

public class FALSE extends Constante {

	@Override
	public boolean evaluate() {
		return false;
	}

	@Override
	public String print() {
		return "FALSE";
	}

}
```

oct. 01, 10 13:59 **Main.java** Page 1/1

```
package logique;

public class Main {

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		Expression exp1 = new NOT(new AND(new FALSE() , new TRUE()));
		System.out.println(exp1.print()+" donne "+exp1.evaluate());
	}

}
```

oct. 01, 10 14:56 **MainObserver.java** Page 1/1

```
package logique;

public class MainObserver {
	public static void main(String[] args) {

		Expression exp1 = new NOT(new AND(new FALSE(), new TRUE()));
		Expression exp2 = new NOT(new OR(new FALSE(), new TRUE()));
		Expression exp3 = new AND(new FALSE(), new TRUE());
		Expression exp4 = new OR(new FALSE(), new TRUE());
		SuperCalculateur sc = new SuperCalculateur();
		sc.ajouterExpression(exp1);
		sc.ajouterExpression(exp2);
		sc.ajouterExpression(exp3);
		sc.ajouterExpression(exp4);
		SuiviDeCalcul co = new SuiviDeCalcul();
		sc.attach(co);
		sc.lancerCalcul();
	}
}
```

```
package logique;

public class NOT extends OperateurUnaire {

	public NOT(Expression exp) {
		super(exp);
	}

	@Override
	public boolean evaluate() {
		return !membres[0].evaluate();
	}

	public String print() {
		return "NOT("+membres[0].print()+")";
	}

}
```



```
package logique;

public abstract class Observer {
	abstract public void update();

}
```

```
package logique;

public abstract class OperateurBinaire extends Operateur {
        public OperateurBinaire(Expression exp1, Expression exp2) {
                membres = new Expression[2];
                membres[0] = exp1;
                membres[1] = exp2;
        }
        public String print() {
                return "("+membres[0].print()+ printOperateur() + membres[1].pri
nt()+")";
        }

        protected abstract String printOperateur();
}
```



```
package logique;

public abstract class Operateur implements Expression {
        Expression[] membres;

}
```

```java
package logique;

public abstract class OperateurUnaire extends Operateur {
	public OperateurUnaire(Expression exp){
		membres = new Expression[1];
		membres[0] = exp;
	}
}
```



```java
package logique;

public class OR extends OperateurBinaire {

	public OR(Expression exp1 , Expression exp2) {
		super(exp1, exp2);
	}

	public boolean evaluate() {
		if (membres[0].evaluate())
			return true;
		else
			return membres[1].evaluate();
	}

	protected String printOperateur() {
		return " OR ";
	}

}
```

oct. 01, 10 13:56 **Subject.java** Page 1/1

```
package logique;

import java.util.ArrayList;

public abstract class Subject {
        ArrayList observers;

        public void attach(Observer obs) {
                observers.add(obs);
        }

        public void detach(Observer obs) {
                observers.remove(obs);
        }

        public void notifiyAllObs() {
                for (Object ob : observers) {
                        ((Observer)ob).update();
                }
        }

        public Subject() {
                observers = new ArrayList();
        }

}
```

oct. 01, 10 14:56 **SuiviDeCalcul.java** Page 1/1

```
package logique;

public class SuiviDeCalcul extends Observer {

        public void update() {
                System.out.println("Calcul TerminÃ©");
        }

        public SuiviDeCalcul() {
                super();
        }

}
```

oct. 01, 10 14:01 **SuperCalculateur.java** Page 1/1

```
package logique;

public class SuperCalculateur extends Subject {
	Expression[] exp;
	int cursor;

	public void ajouterExpression(Expression ex) {
		if (cursor < exp.length)
			exp[cursor++] = ex;
	}

	public void lancerCalcul() {
		// Une boucle infinie vide la pile exp et evalue les expressions
		while (cursor > 0) {
			Expression current = exp[--cursor];
			current.evaluate();
			// TODO: Ici il faudrait que le SuperCalculateur annonce
  ces
			// observateurs, qu'il a fini un calcul
			notifiyAllObs();
		}
	}

	public SuperCalculateur() {
		super();
		exp = new Expression[10];
		cursor = 0;
	}

}
```

sept. 30, 10 14:30 **TRUE.java** Page 1/1

```
package logique;

public class TRUE extends Constante {

	public boolean evaluate() {
		return true;
	}


	public String print() {
		return "TRUE";
	}

}
```