Algorithmique & Programmation

# Notion d'alternative

yann.secq@univ-lille.fr

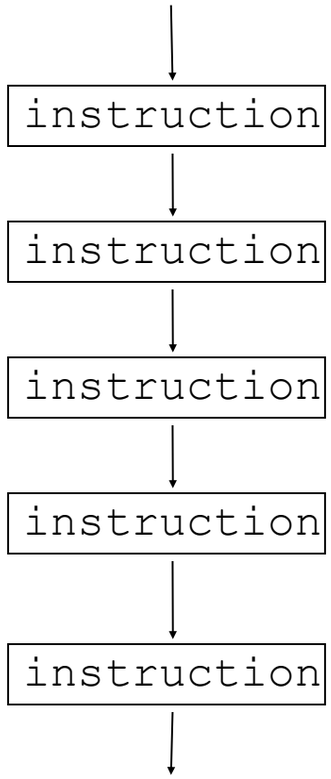
ABDELKADER Omar, BIRLOUEZ Martin, BONEVA Iovka, DELECROIX Fabien, LEQUINIOU Erwann, MARSHALL-BRETON Christopher, REKIK Yosra, SECQ Yann, SOW Younoussa, SUDHEENDRAN Megha, SUE Yue

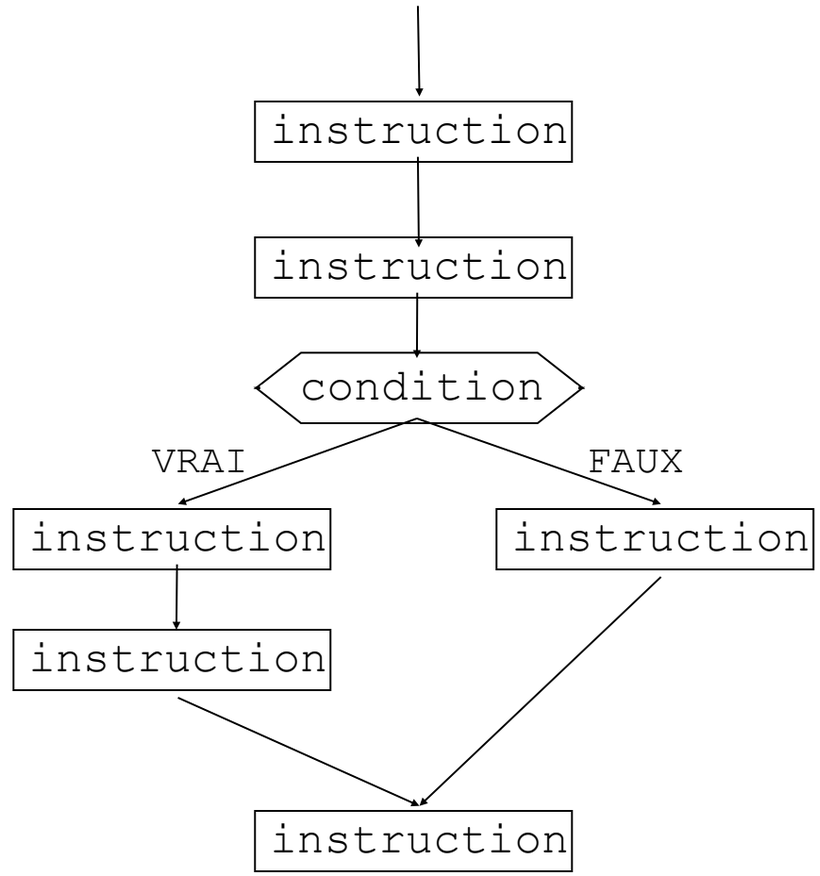

*Saison 2025-2026 / R1.01+R1.02*
*Initiation au développement*

# Structure de contrôle

- La base est la séquence d'instruction
- Nécessité d'influencer le choix des instructions à exécuter
- Les **structures de contrôle** (du flux):
  - **Alternative** (instruction conditionnelle) et **répétition** (boucle)

# Séquence vs. Alternative



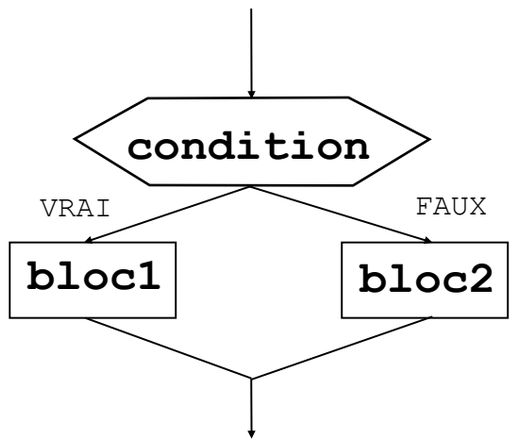

**Exécution séquentielle**

**Exécution conditionnelle**

# Structure de contrôle: **alternative**

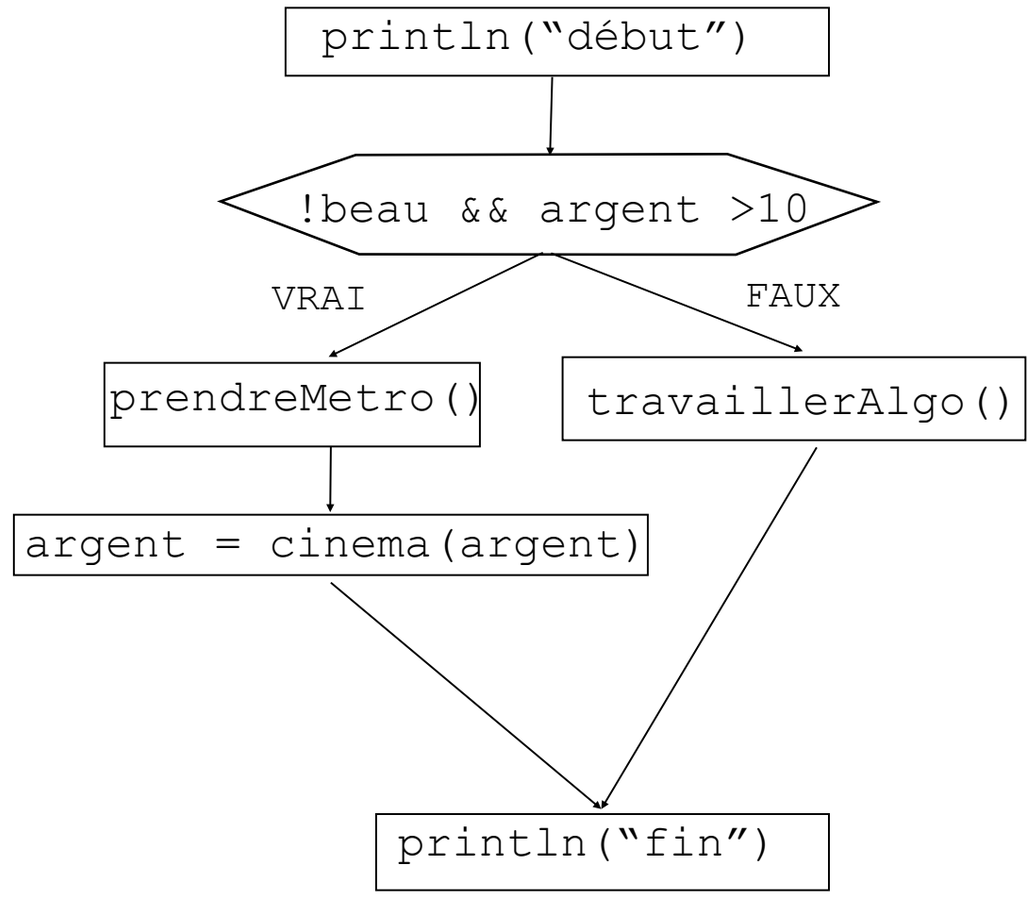
- L'alternative permet de faire un choix entre deux séquences d'instructions en fonction d'une condition
- Si la condition est vérifiée (`VRAI`) alors la première séquence est exécutée, sinon c'est la seconde

```
if (<condition>) {
  <bloc1>
} else {
  <bloc2>
}
```



# **if** (<cond>) **{...} else {...}**

```
class Alternative extends Program {
  void algorithm() {
    boolean soleil = ...;
    int argent = ...;
    // supposons soleil et argent initialisés
    println("début");
    if (!soleil && argent > 10) {
      prendreMetro();
      argent = cinema(argent);
    } else {
      travaillerAlgo();
    }
    println("fin");
  }
}
```

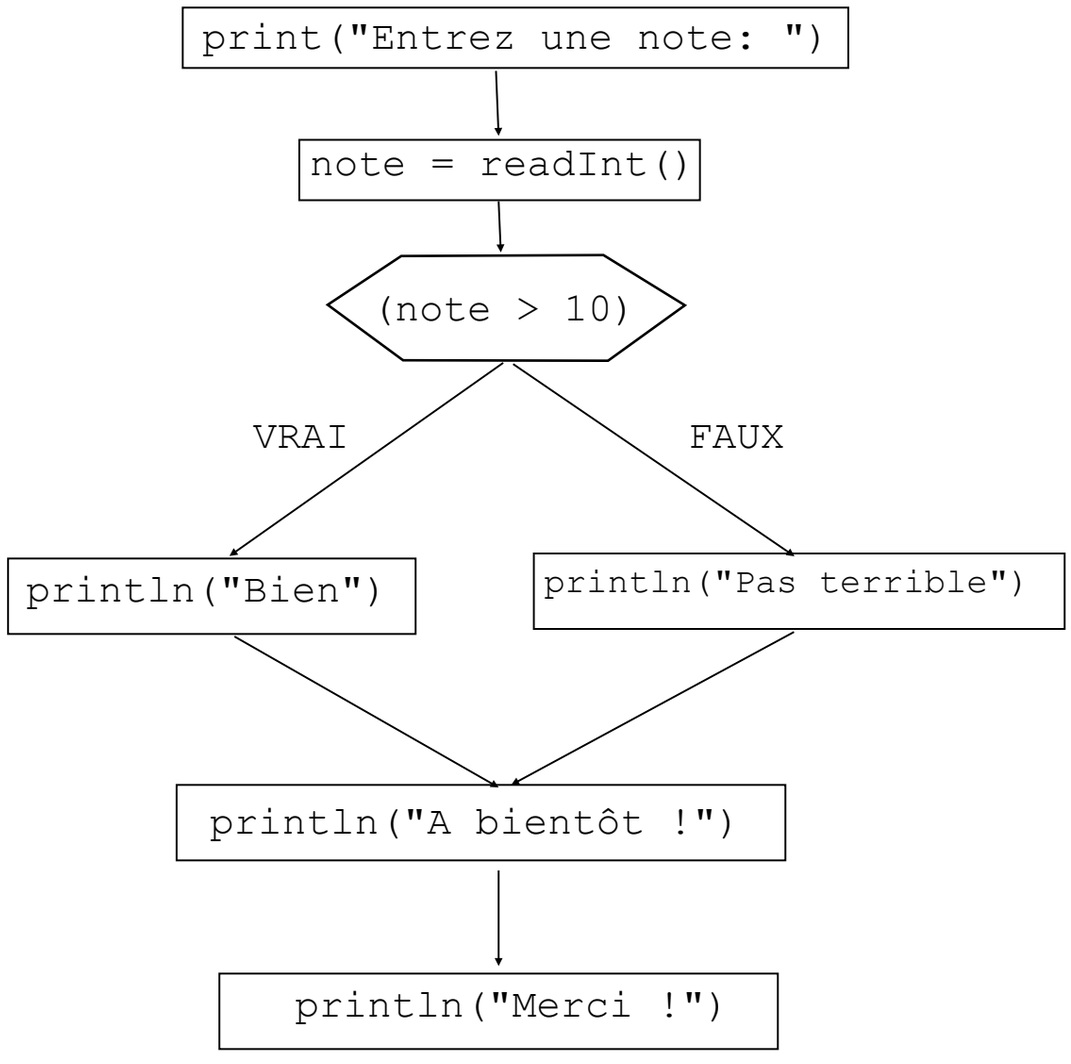


# Exemple

- Afficher un commentaire sur une note en fonction de sa valeur
- En entrée, on dispose d'un nombre compris entre [0,20] et l'on souhaite afficher en sortie `Bien` si la note est strictement supérieure à 10 et `Pas terrible` sinon

# Alternative simple

```
class TestNote extends Program {
  void algorithm() {
    int note;

    print("Entrez une note: ")
    note = readInt();
    if (note > 10) {
      println("Bien");
    } else {
      println("Pas terrible");
    }
    println("A bientôt !");
    println("Merci !");
  }
}
```

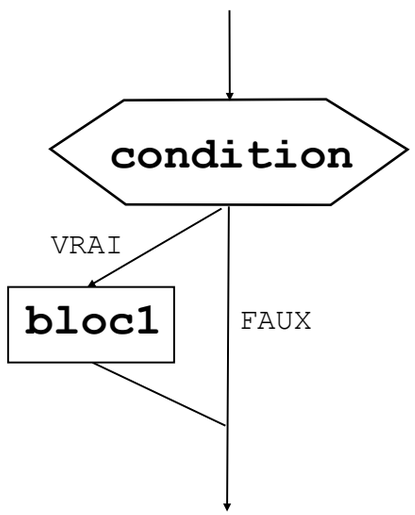


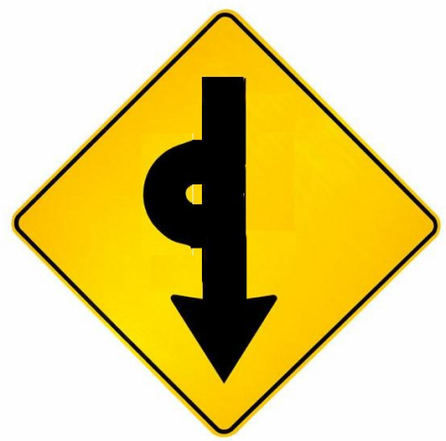
# Structure de contrôle: **alternative « dégénérée »**

- Alternative « dégénérée » : lorsqu'il n'y a pas de deuxième branche (ie. pas de `SINON/else`)

- Si la condition est vérifiée (`VRAI`) alors la première séquence est exécutée, sinon rien

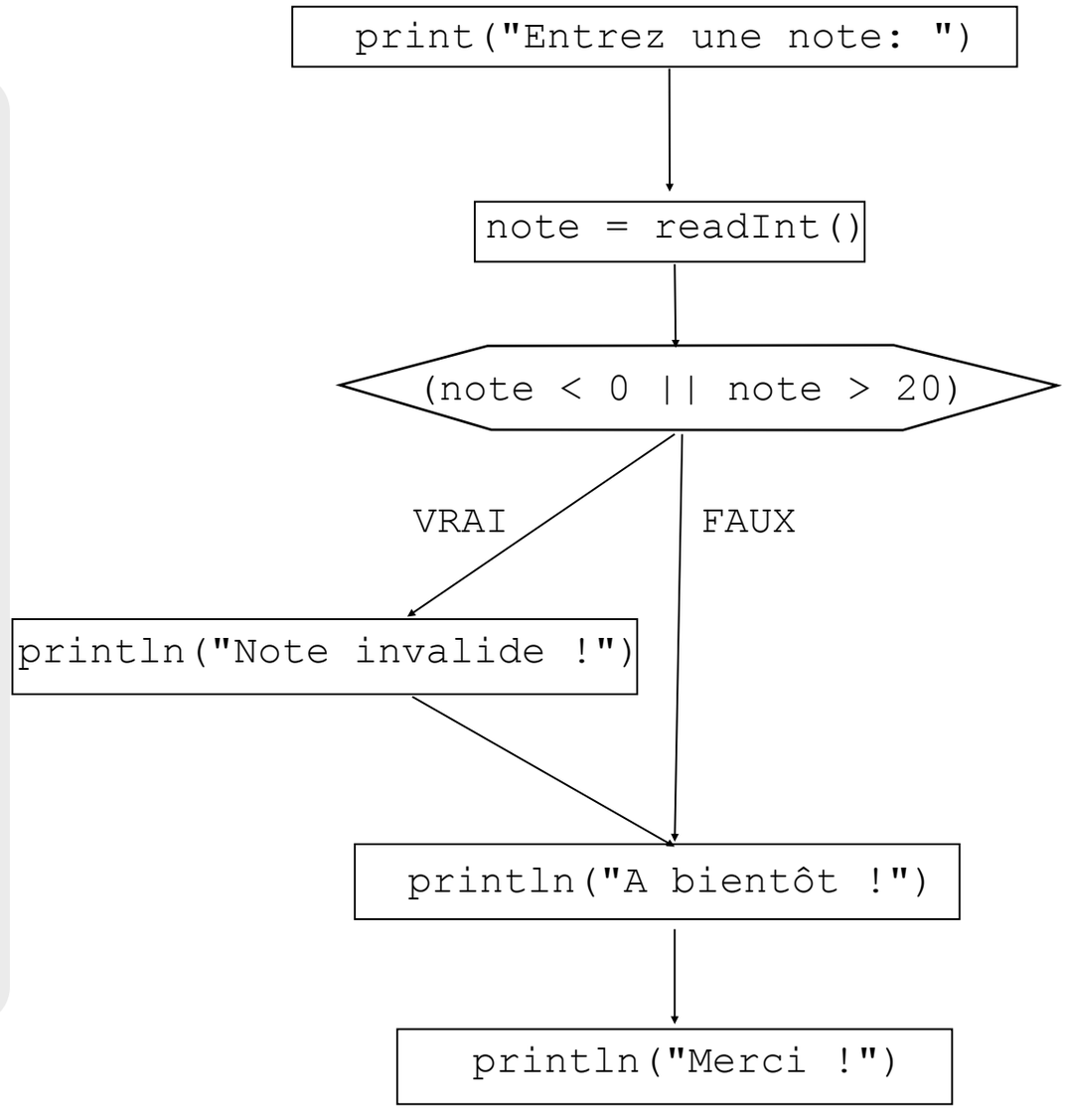
```
if (<condition>) {
  <bloc1>
}
```



# Alternative dégénérée

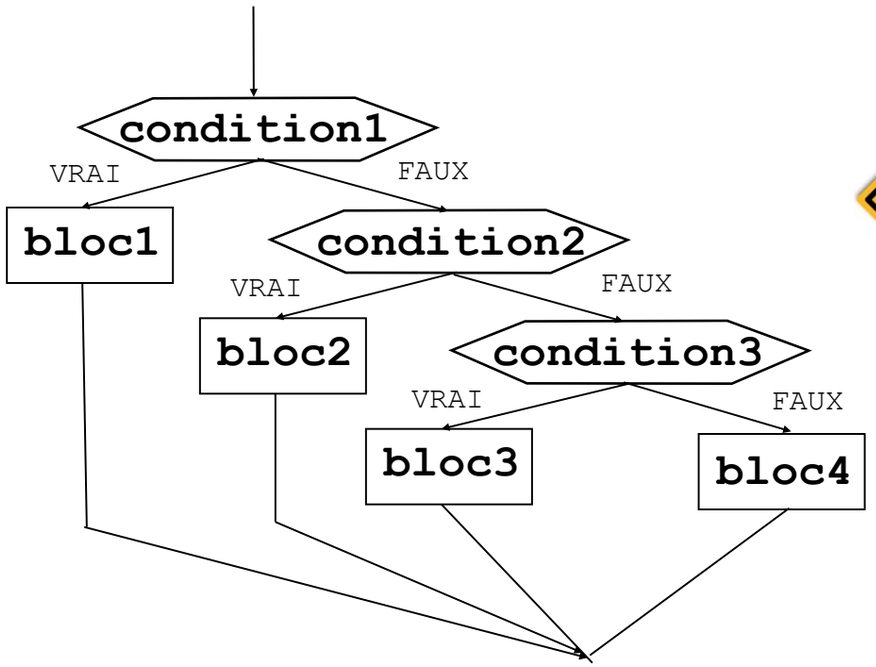
```
class TestSiDegenere extends Program {
  void algorithm() {
    int note;

    print("Entre une note : ")
    note = readInt();
    if (note < 0 || note > 20) {
      println("Note invalide !")
    }
    println("A bientôt !");
    println("Merci !");
  }
}
```

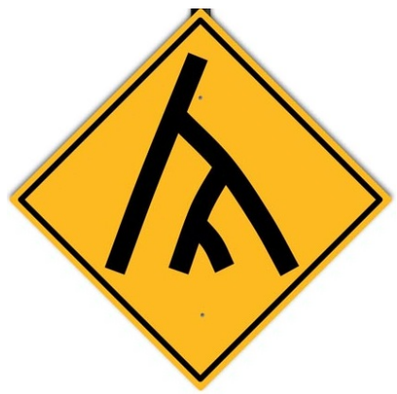


# Structure de contrôle: **alternatives en cascade**

- Possibilité d'alternatives en cascade pour discriminer plus que deux situations
- Règle d'écriture pour simplifier la structure

```
if (<condition1>) {
  <bloc1>
} else if (<condition2>) {
  <bloc2>
} else if (<condition3>) {
  <bloc3>
} else {
  <bloc4>
}
```



# Utilisation d'alternatives

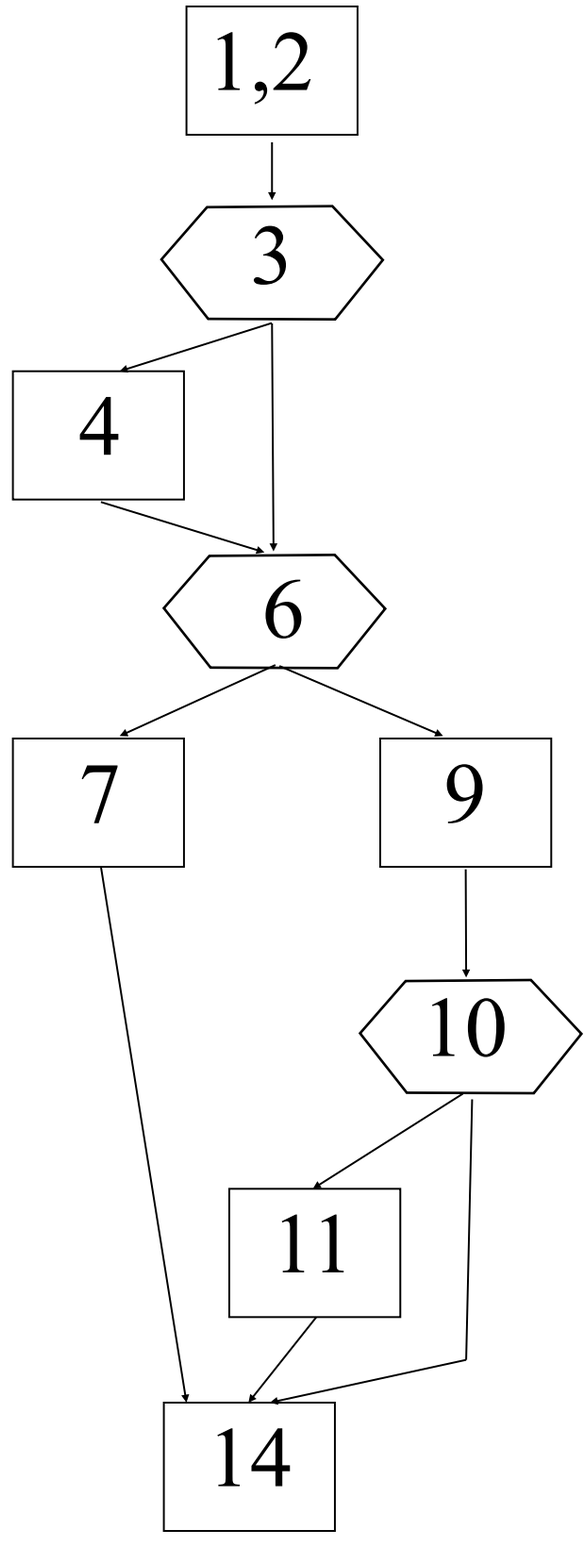
- Réfléchir à la condition (simple ou complexe)
- Type d'alternative: `if ... else ...` **ou** `if ...` ou cascade d'alternatives
- Possibilité d'imbriquer plusieurs alternatives
- **Pensez à l'indentation** : visualisation des niveaux d'imbrication de structures

# Exemple

- Afficher un commentaire sur une note en fonction de sa valeur
- En entrée, un nombre représentant une note et en sortie un des messages suivants:

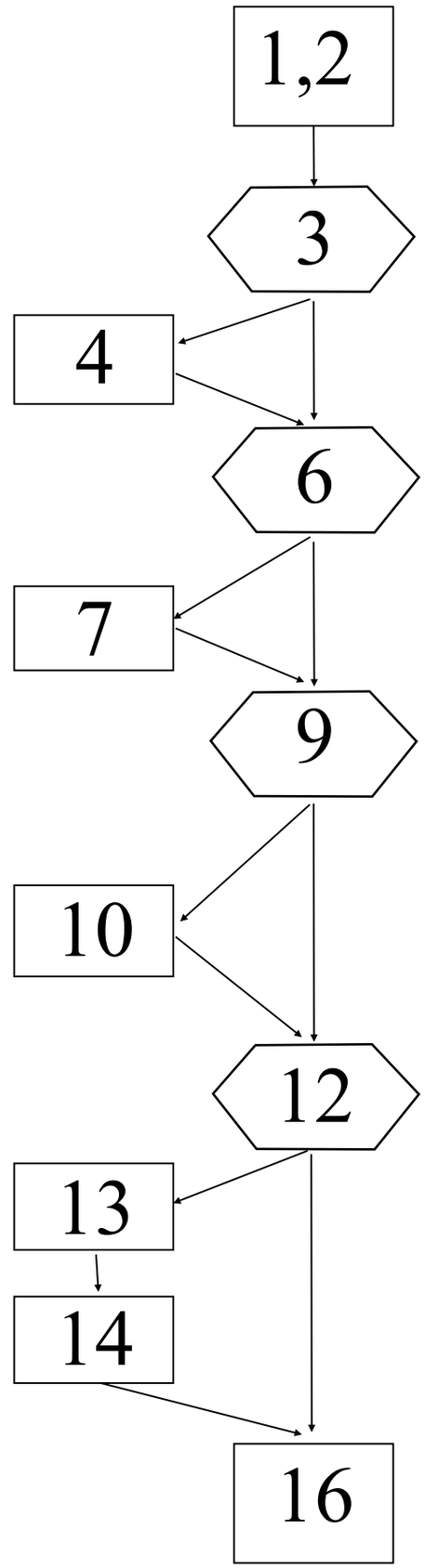
| | |
|---|---|
| `note < 0 OU note > 20` | `"Note invalide"` |
| `note < 10` | `"Redoublement"` |
| `note >= 10` | `"Passage"` |
| `note > 16` | `"Félicitations !"` |

```
  class TestEnchainement extends Program {
    void algorithm() {
      int moyenne;
1     print("Entrez votre moyenne : ");
2     moyenne = readInt();
3     if (moyenne < 0 || moyenne > 20) {
4       println("Note invalide");
5     }
6     if (moyenne >= 0 && moyenne < 10) {
7       println("Redoublement");
8     } else {
9       println("Passage");
10      if (moyenne > 16) {
11        println("Félicitations !");
12      }
13    }
14    println("A bientôt … ");
    }
  }
```



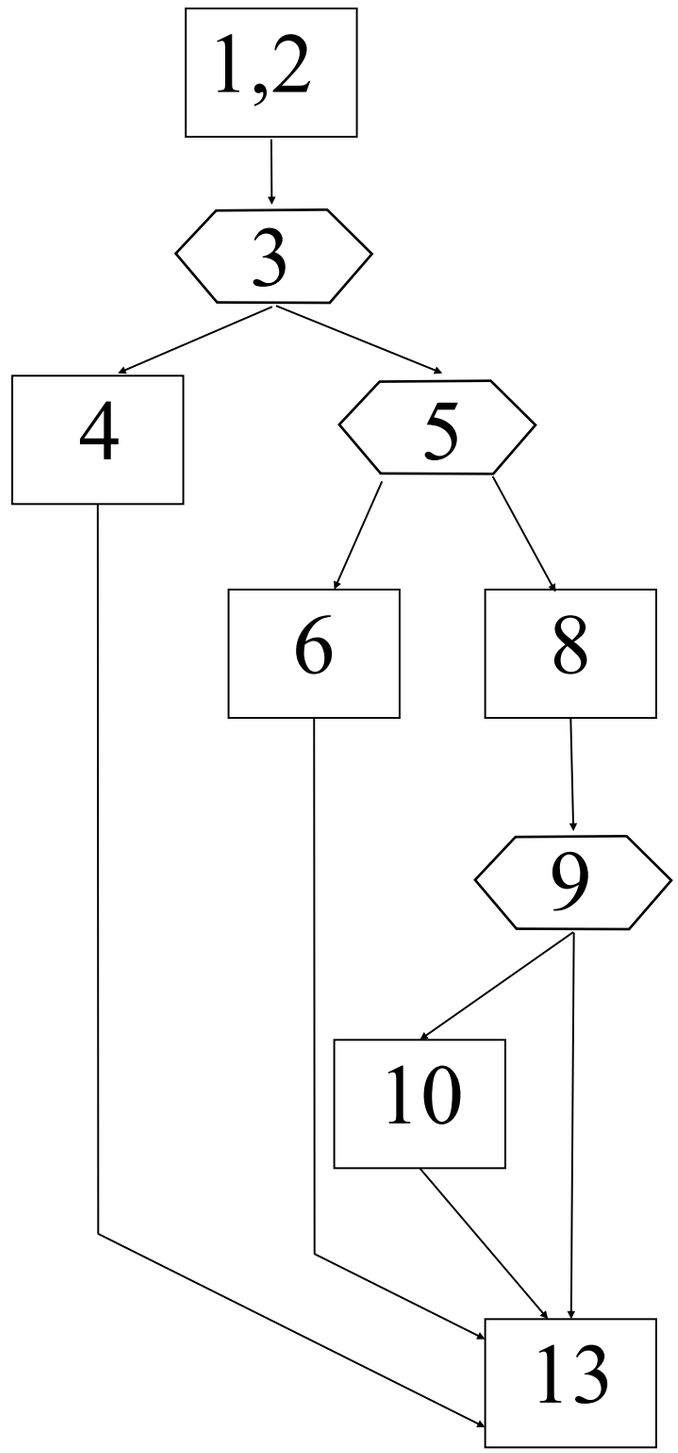

# Problème ?

```
  class TestEnchainement extends Program {
    void algorithm() {
      int moyenne;

1     print("Entrez votre moyenne : ");
2     moyenne = readInt();
3     if (moyenne < 0 || moyenne > 20) {
4       println("Impossible !");
5     }
6     if (moyenne >= 0 && moyenne < 10) {
7       println("Redoublement");
8     }
9     if (moyenne >= 10 && moyenne < 16) {
10      println("Passage");
11    }
12    if (moyenne >= 16 && moyenne <= 20) {
13      println("Félicitations !");
14      println("Félicitations !");
15    }
16    println("A bientôt … ");
    }
  }
```




# Problème ?

```
  class TestEnchainement extends Program {
    void algorithm() {
      int moyenne;

1     print("Entrez votre moyenne : ");
2     moyenne = readInt();
3     if (moyenne < 0 || moyenne > 20) {
4       println("Impossible !");
5     } else if (moyenne < 10) {
6       println("Redoublement :(");
7     } else {
8       println("Passage");
9       if (moyenne > 16) {
10        println("Félicitations !");
11      }
12    }
13    println("A bientôt … ");
    }
  }
```



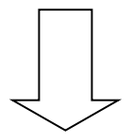

# Nombre de chemins ?

# Quelques cas classiques

- Simplification d'enchaînement d'alternatives par introduction d'une condition complexe
- Erreur de logique lors d'imbrications d'alternatives
- Suppression de code redondant dans différentes branches d'une alternative

# Complexité de structure ou de condition

```
if (a) {
  if (b) {
    println("a et b sont VRAI !");
  }
}
```



```
if (a && b) {
  println("a et b sont VRAI !");
}
```

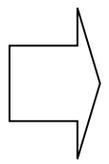
# Exclusion mutuelle de conditions

- Correspond à une erreur de logique lors de l'énumération des cas possibles

```
if (a >= 0) {
  if (a < 0) {
    println("Erreur : jamais exécuté !");
  }
}
```

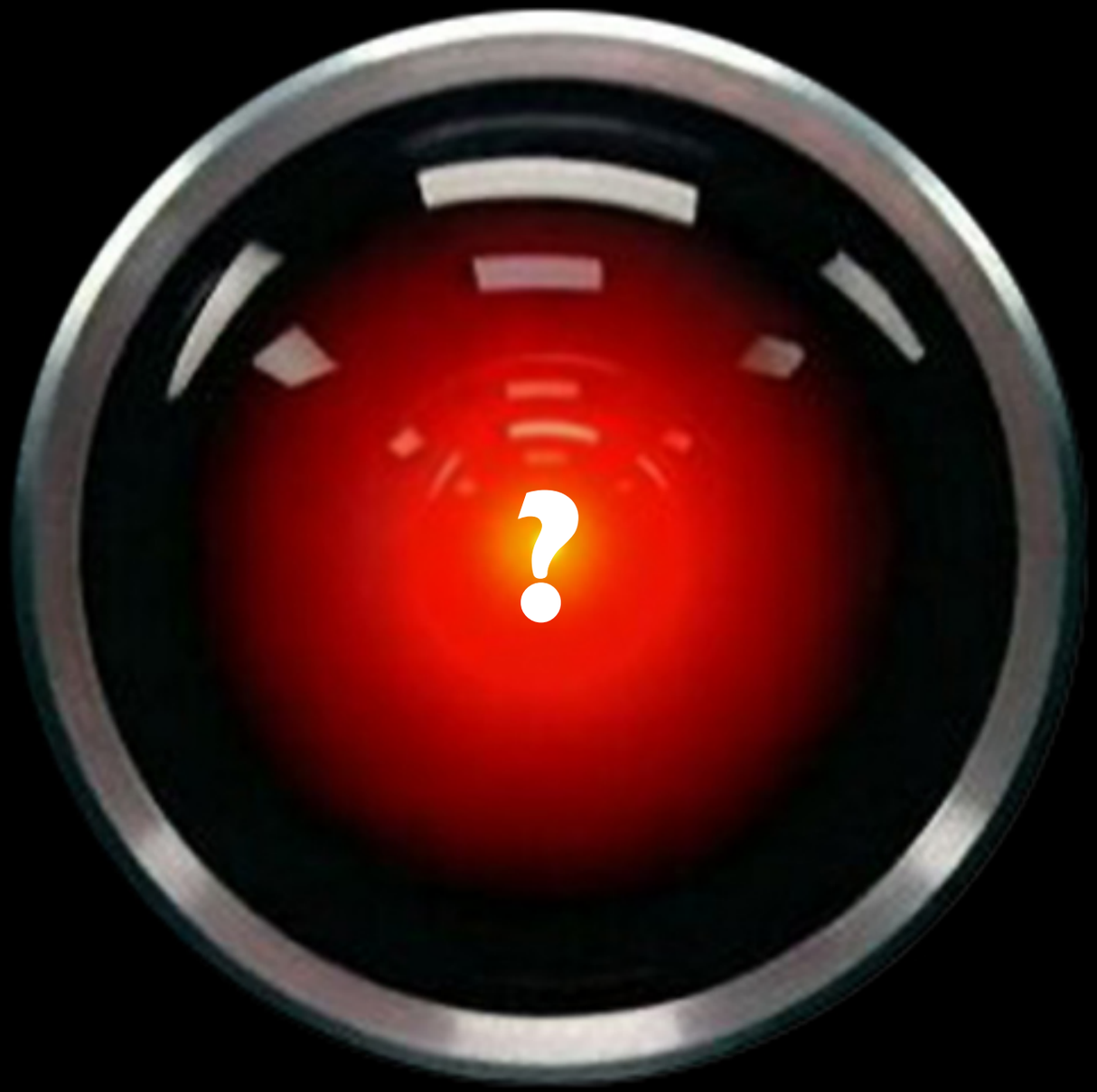
# Suppression de code redondant

```
if (a > 0) {
  if (b > 0) {
    println("Oui");
  } else {
    println("Non");
  }
} else {
  if (b > 0) {
    println("Non");
  } else {
    println("Oui");
  }
}
```



```
if ((a>0) == (b>0)) {
  println("Oui");
} else {
  println("Non");
}
```

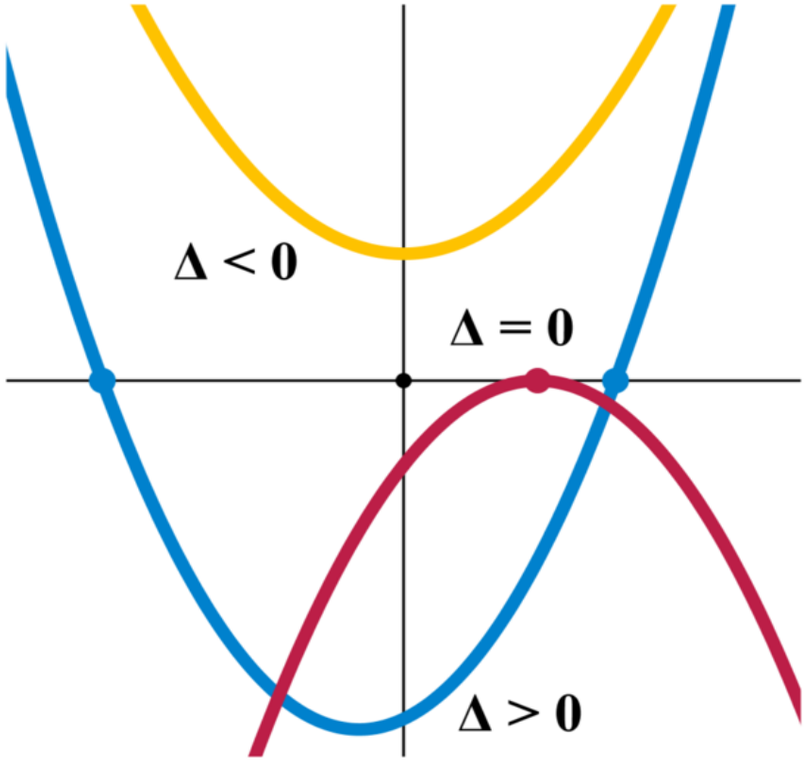
```
if ((a>0) ^ (b>0)) {
  println("Non");
} else {
  println("Oui");
}
```



# Racines d'un polynôme

Soit un polynôme: $\mathbf{a.x^2 + b.x + c = 0}$

Calcul du discriminant: $\Delta = b^2-4ac$




$\Delta < 0$ : pas de racine

$\Delta = 0$ : $x_0 = -b / 2a$

$\Delta > 0$ : $x_1 = (-b - \sqrt{\Delta}) / 2a$

$x_2 = (-b + \sqrt{\Delta}) / 2a$

*Algorithme (au tableau :)*