



Informatique 1

5. Boucles et conditions

Objectifs du cours

Changements dans le flux séquentiel

- ▶ Exécution conditionnelle
 - Branchements : **if**, **if/else**, **switch**
- ▶ Répétitions de code
 - Boucles : **while**, **do**, **for**

Executing this or that

5.1 EXÉCUTION CONDITIONNELLE / BRANCHEMENTS

Exécution séquentielle

- **Généralement**, le code s'exécute *séquentiellement*
→ "de haut en bas"
- Comme vu, cela permet de faire l'équivalent d'une simple calculatrice (mais très rapide) avec un ordinateur
- Il faut pouvoir faire plus !

Exécution séquentielle (2)

Moyenne

- 3 éléments : $(a_1 + a_2 + a_3) * \frac{1}{3}$
- 3'000'000 éléments ?
 $(a_1 + a_2 + \dots + a_{3 * 10^6}) * \frac{1}{3 * 10^6}$
- Il faut pouvoir répéter des opérations !

boucles



Équation 2^{ème} degré

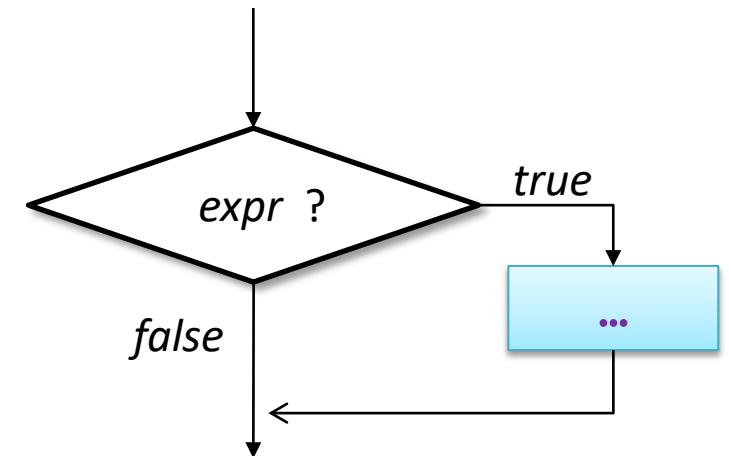
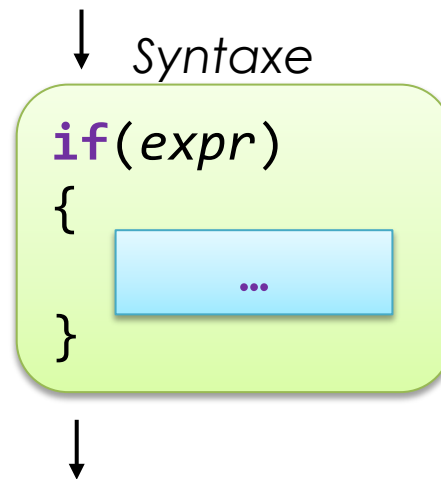
$$ax^2 + bx + c = 0 \quad \begin{cases} x_1 = \frac{-b + \sqrt{\Delta}}{2a} & \text{et} & x_2 = \frac{-b - \sqrt{\Delta}}{2a} \\ \Delta = b^2 - 4ac & \left\{ \begin{array}{l} ax^2 + bx + c = a \left(x + \frac{b}{2a}\right)^2 \\ \text{et} \quad x_1 = x_2 = -\frac{b}{2a} \end{array} \right. \end{cases}$$



branchements

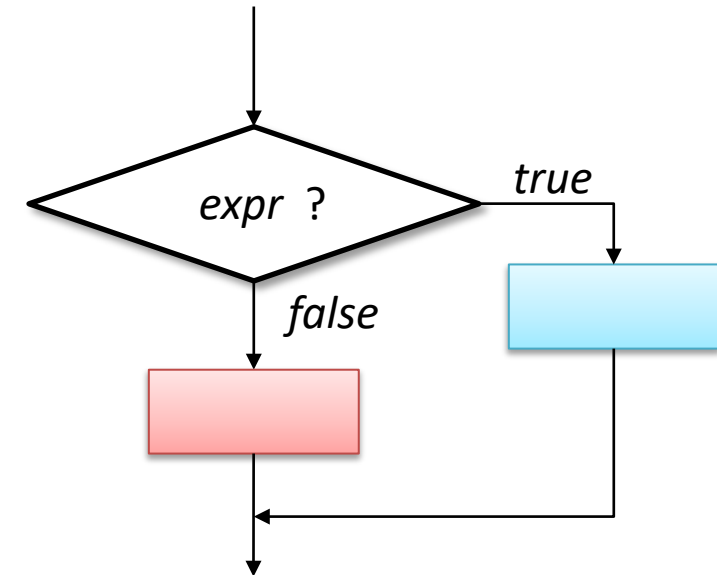
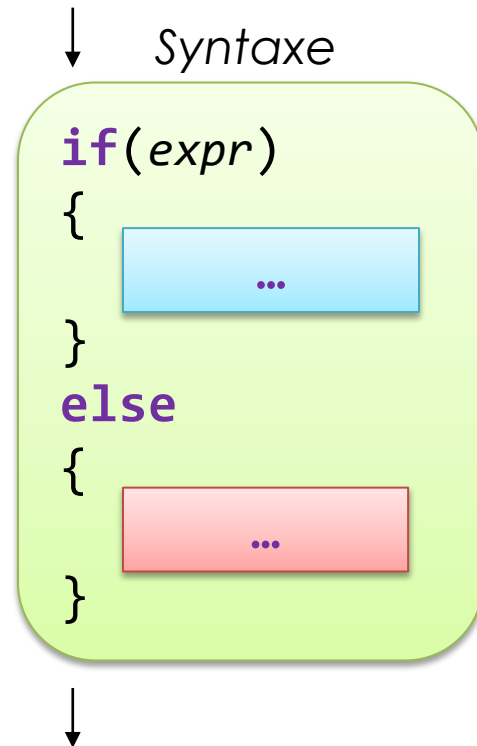
L'instruction **if**

- Instruction de branchement
- Permet de **décider** si un bloc est exécuté ou non
 - *expr* est une expression **booléenne**
 - Si *expr* s'évalue à **true**, le bloc bleu est exécuté, autrement non



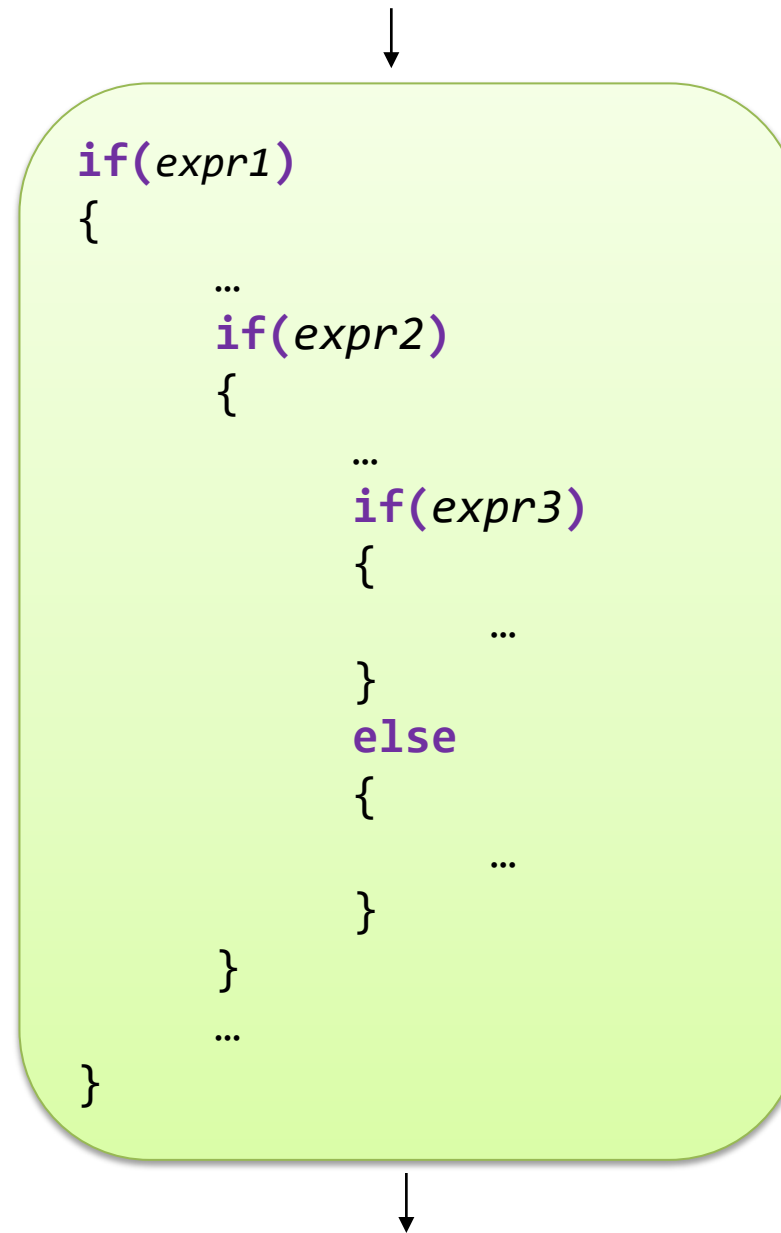
L'instruction **if ... else**

- Même principe que **if** mais avec une alternative.
- Si *expr* s'évalue à **true**, le bloc bleu est exécuté, autrement le bloc rouge est exécuté



If imbriqués

- Il est possible d'imbriquer plusieurs niveaux de **if** / **if...else**
- La notion de *bloc* est très importante pour déterminer ce qui sera exécuté



L'instruction **if ... else** (2)



```
if(temp <= 0){  
    System.out.println("Warning ! Very cold");}  
else if(temp > 0 && temp < 15){  
    System.out.println("Wear a pullover");}  
else{  
    System.out.println("Temperature is warm");}
```

Remarque

- Si une seule instruction dans le **if**, les { } optionnels

- Attention !



5.1 Écrivez le code permettant de calculer la valeur absolue d'une variable a et de l'afficher sur la console

5.2 Écrivez le code permettant de stocker dans une variable b si une variable a est paire

L'instruction **switch ... case**

- Même principe que **if** imbriqués.
- En fonction de la valeur de *expr*, **l'un des blocs** est exécuté.
- Permet gérer plusieurs cas simplement
- Il y a une valeur par défaut → **default**
- **break** termine le switch

Syntaxe

```
switch(expr)
{
  case a:
    ...
    break;

  case b:
    ...
    break;

  ...

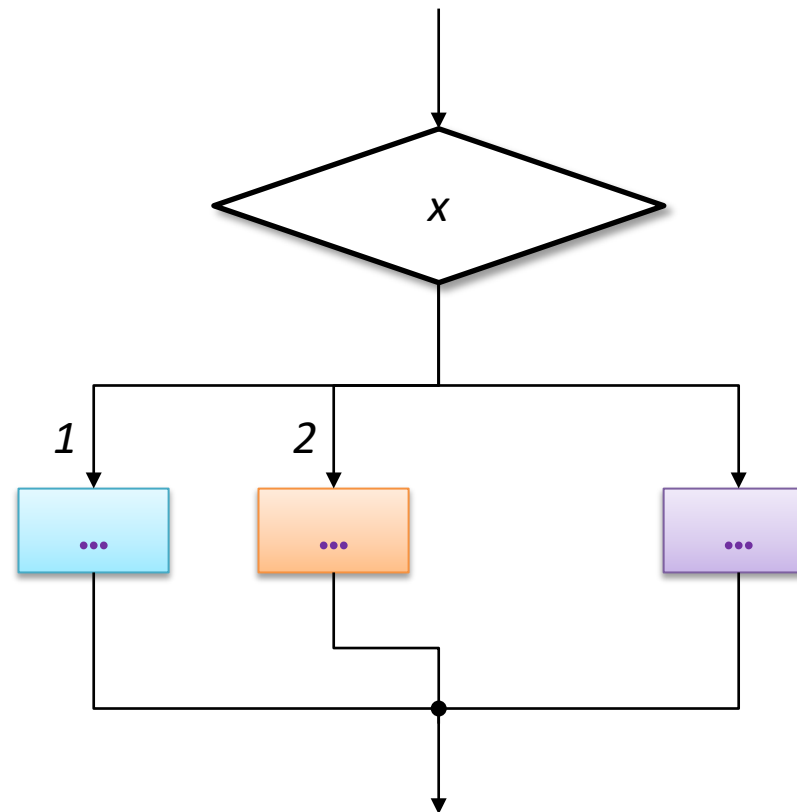
  default:
    ...
    break;
}
```

L'instruction `switch ... case` (2)

↓ *Syntaxe*

```
int x = 2;
switch(x)
{
  case 1:
    ...
    break;
  case 2:
    ...
    break;
  default:
    ...
    break;
}
```

↓



Exercice: if to switch

```
System.out.println("The Java lecture is : ");
System.out.println("\t1 - Boring");
System.out.println("\t2 - I've never been there");
System.out.println("\t3 - Ok");
System.out.println("\t4 - Do we have a Java class?");

System.out.println("What is your choice ? ");
int value = Input.readInt();

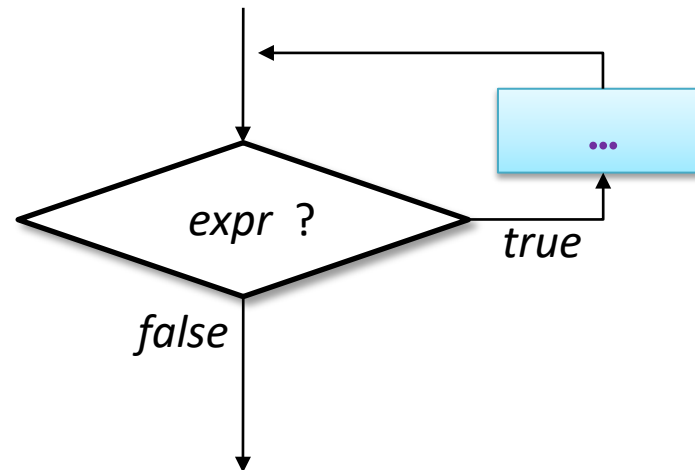
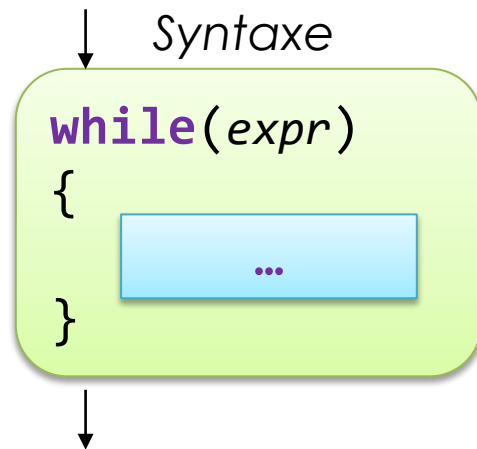
if(value == 1)
    System.out.println("No it's not !");
else if(value == 2)
    System.out.println("You should then give it a go");
else if(value == 3)
    System.out.println("Danke!");
else if(value == 4)
    System.out.println("Yes! Every Tuesday morning");
else
    System.out.println("Not a valid input");
```

Again and again

5.2 BOUCLES

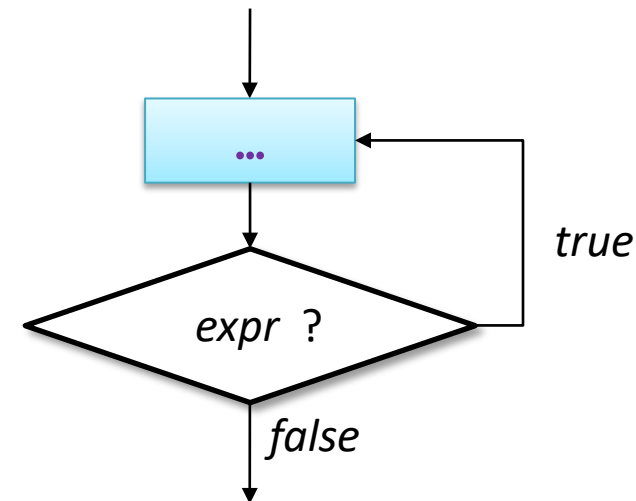
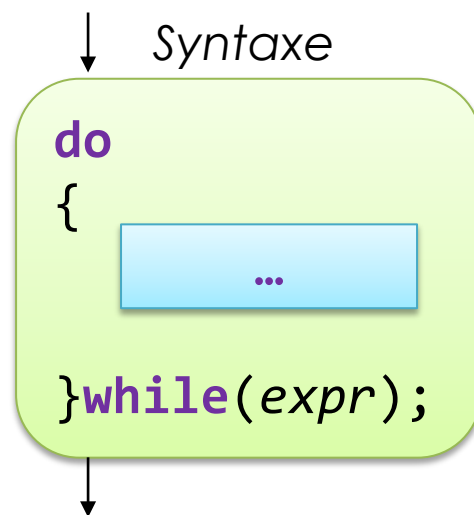
L'instruction **while**

- Répète un bloc tant qu'une condition est vraie.
 - ▶ Tant que *expr* s'évalue à **true**, le bloc bleu est exécuté.



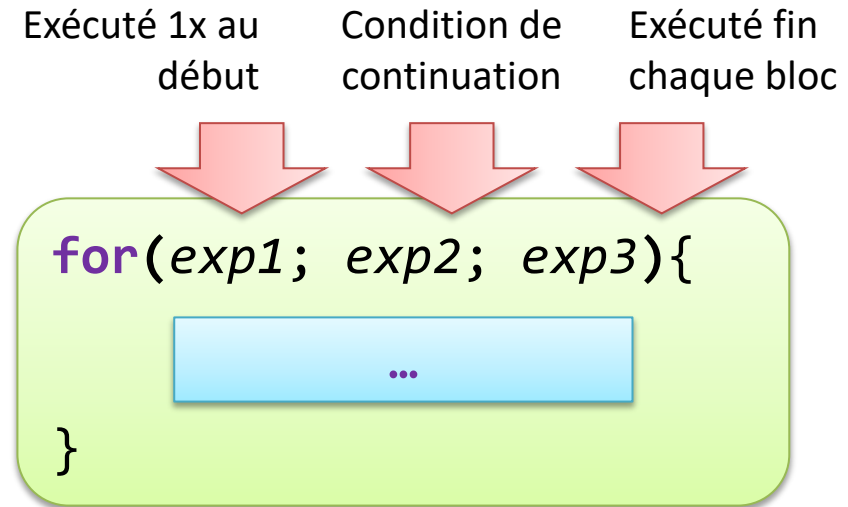
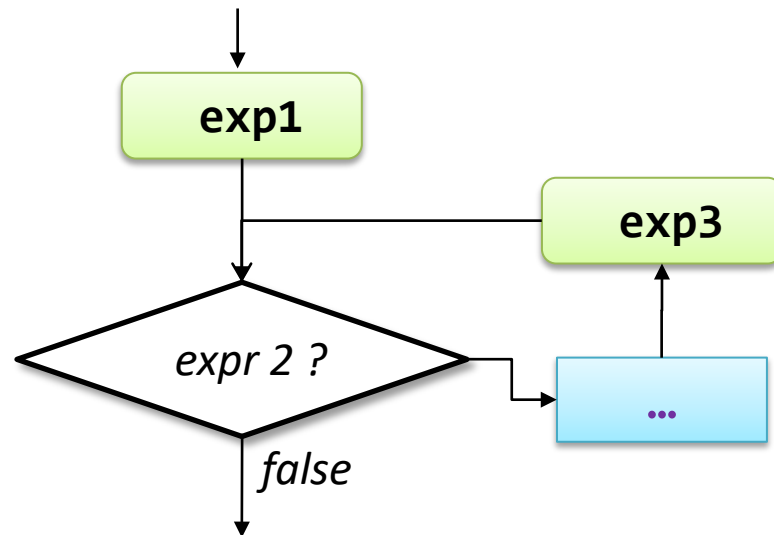
L'instruction **do ... while**

- Très similaire à **while**. Par contre le bloc est toujours exécuté **au moins** une fois.
 - Le bloc bleu est exécuté puis, tant qu'**expr** s'évalue à **true**, le bloc bleu est exécuté à nouveau.



L'instruction **for**

- Très utilisé !
- Permet de réaliser une boucle un certain nombre de fois

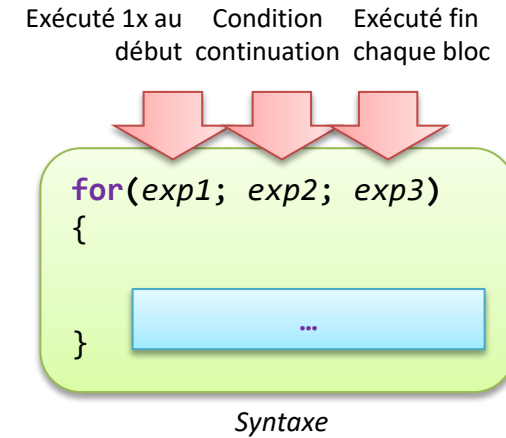


Syntaxe

L'instruction **for** – Exemple 1

```
↓  
for(int i = 0; i < 5; i++)  
{  
    System.out.println(i);  
}  
↓
```

Exemple



Résultat affiché

L'instruction **for** – Exemple 2

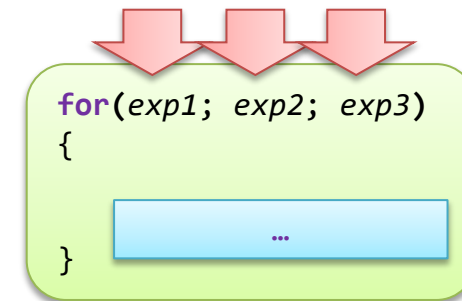
$$\frac{1+2+3+\dots+9+10}{10} = \frac{1}{10} \sum_{i=1}^{10} i$$

```
double average = 0.0;
// Sum
for(int i = 1; i <= 10; i++)
{
    average += i;
}

// Final division
average /= 10.0;
```

Exemple

Exécuté 1x au début Condition continuation Exécuté fin de chaque bloc



Syntaxe

Aussi la moyenne de 1 à n

$$\frac{1}{n} \sum_{i=1}^n i$$

Rupture - **break**

- L'instruction **break** **stoppe** la première structure de boucle rencontrée vers le haut
- Utilisée en général en conjonction avec un **if**

```
int i = 0;

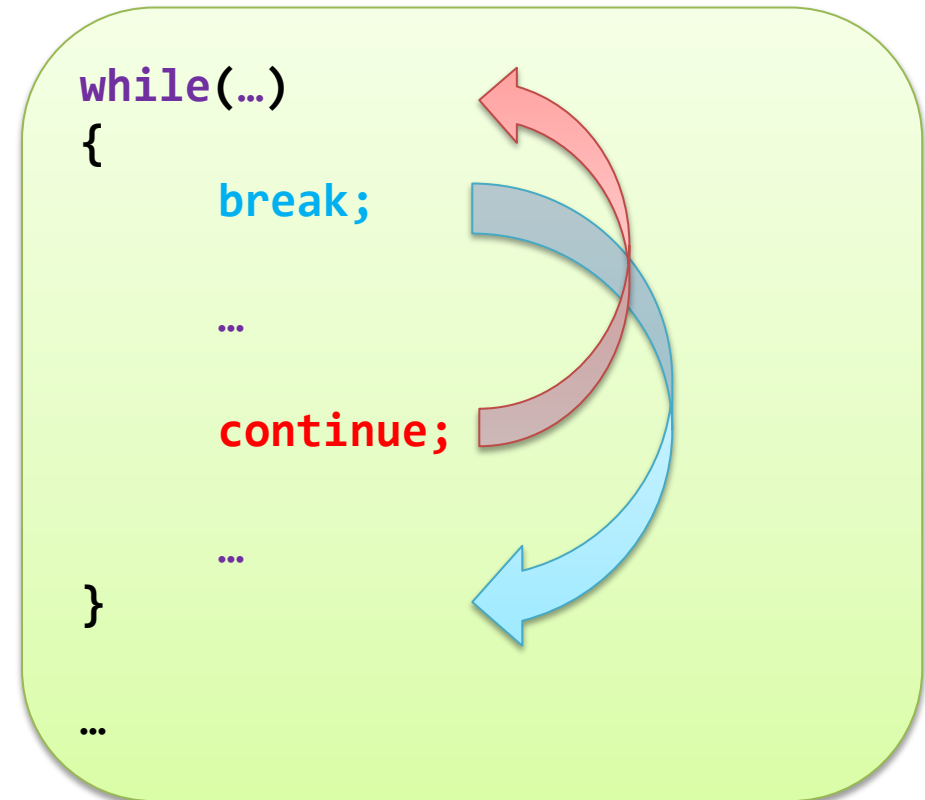
while(true)
{
    if(i > 10)
    {
        break;
    }

    System.out.println(i++);
}
```

Exemple

Saut - **continue**

- Dans une boucle, parfois on veut passer directement au pas suivant de la boucle
 - Instruction **continue**
- On peut se passer des instructions **break** et **continue** ...
- Utilisée en général en conjonction avec un **if**



Exemple

Erreurs typiques

- Il n'y a PAS de point-virgule à la fin du **if**, **for** et **while**

```
if(a == 0) ;
```

```
for(int i = 0; i < 10; i++) ;
```

```
while(a == 0) ;
```

Conclusion du cours

- Vous avez appris dans ce cours à :
 - ▶ Modifier le flux séquentiel en fonction de différents types de choix.
 - **if ... else**
 - **switch**
 - ▶ Comment réaliser des boucles de différents types
 - **while**
 - **do**
 - **for**