IPSA / N.DECKNER 2024

Cours 2 - Contrôle du flot d'instructions

Notions : if, else, else if, switch, while, do while, for, range-for, break, continue

L'instruction conditionnelle if / else

l'instruction `if` execute les instructions seulement si la condition est évaluée à `true`.

if

```
if (condition) {
    instructions
}
```

```
if (x < 0) {
    x = 1;
}
```

if else

l'instruction `if / else` execute les instructions du `if` si la condition est évaluée à `true` et sinon execute les instruction du `else`.

```
if (condition) {
    instructions
}
else {
    instructions
}
```

else if

l'instruction `else if` est un raccourcit. Les deux codes suivant sont équivalents.

```
if (condition) {
    instructions
}
else if (condition) {
    instructions
}
```

```
if (condition) {
    instructions
}
else  {
    if (condition) {
        instructions
    }
}
```

L'instruction conditionnelle switch

L'instruction `switch` exécute des instructions en fonction d'une valeur. Chaque bloc `case` doit se terminer par l'instruction `break`.

```
int x = 2;
int result = 0;

switch (x) {
    case 1:
        result = 101;
        break;
    case 2:
        result = 102;
        break;
    case 3:
        result = 103;
        break;
    default:
        result = 100;
        break;
}

// result vaut 102
```

L'équivalent avec des `if` / `else` :

```
int x = 2;
int result = 0;

if (x == 1) {
    result = 101;
else {
    if (x == 2) {
        result = 102;
    }
    else {
        if (x == 3) {
            result = 103;
        else {
            result = 100;
        }
}
```

L'équivalent avec des `else if` :

```
int x = 2;
int result = 0;

if (x == 1) {
    result = 101;
else if (x == 2) {
    result = 102;
}
else if (x == 3) {
    result = 103;
}
else {
    result = 100;
}
```

Les boucles

while

Exécute les instructions tant que la condition est vrai.

```
while (condition) {
    instructions
}
```

do while

Exécute les instructions une première fois, puis les execute à nouveau tant que la condition est vrai.

```
do {
    instructions
}
while (condition);
```

for

```
for (initialisation; condition; expression) {
    instructions
}
```

```
x = 0;

for (int i = 0; i < 4; ++i) {
    x += i;
}

// après la boucle x vaut 6
```

range-for

```
for (déclaration : collection) {
    instructions
}
```

```
x = 0;
vector<int> numbers = {1, 2, 3};

for (int number : v) {
    x += number;
}

// après la boucle x vaut 6
```

Saut d'instructions

Les instructions `break` et `continue` permettent de sauter des instructions dans une boucle while, do while, for, range-for ou d'un switch.

break

L'instruction `break` fait quitter la boucle.

```
x = 0;

for (int i = 0; i < 4; ++i) {
    if (i == 2) {
        break;
    }
    x += i;
}

// x vaut 1
```

continue

L'instruction `continue` passe au prochain tour de boucle.

```
x = 0;

for (int i = 0; i < 4; ++i) {
    if (i == 2) {
        continue;
    }
    x += i;
}

// x vaut 4
```