Manuel D'aide

version 1.0

DIVERSITY by EPITECH

I. Introduction

Dans ce petit manuel, vous pouvez trouver tout ce dont vous avez besoin pour aider l'oiseau jaune à prendre son envol !

Tout expert qui ce respect à lut ce petit manuel

Voici un sommaire de ce que le livre contient :

- Une fonction c'est quoi ?
- Instruction conditionnelle en veux-tu ou en voilà
- Les opérateurs sont de sortie

a. Chapitre 1 : Les fonctions c'est quoi

Une fonction permet d'exécuter une suite d'opération. Pour créer une fonction, il vous faut définir son prototype.

```
void nom_de_la_fonction(parametre_1, parametre_2);
```

Pour plus d'informations :

* `void` : N' attardez-vous pas longtemps dessus. Dites-vous que vous en aurez besoin une seule fois et que cela sera expliqué
* `nom_de_la_fonction` : c'est via ce texte que vous pourrez appeler votre fonction
* `parametre_1` & `parametre_2` : ce sont des valeurs que vous devez envoyer à la fonction lors de son appel. Il peut y avoir plus que 2 paramètres, mais il peut aussi en avoir aucun

Plus concrètement, une fonction peut ressembler à ça :

```
void setup() {
    … //code qui sera exécuté une fois la fonction appelée
}
```

b. Chapitre 2 : Instruction conditionnelle en veux-tu ou en voilà

Les instructions conditionnelles ou plus simplement condition vont permettre de rendre votre programme beaucoup plus flexible. En parallèle, cela va le rendre plus complexe.

Voici un exemple général :

```
if (condition1) {
    … //code s'exécutant si la condition1 est vraie
}
else if (condition2) {
    … //code s'exécutant si la condition2 est vraie et que la condition1 est fausse
}
else if (condition3) {
    … //code s'exécutant si la condition3 est vraie et que la condition1 et que la condition2 sont fausses
}
… //autre condition si nécessaire
else {
    … //code s'exécutant si aucune des conditions précédentes n'est vraie
}
```

c. Chapitre 3 : Les opérateurs sont de sortie

Il existe plusieurs types d'opérateurs, dans un premier temps il y a des opérateurs Mathématiques qui sont assez classique pour faire des calculs :

- « + » | « - » | « * » | « / » représentent l'addition, la soustraction, la multiplication ainsi que la division
- « % » correspond au modulo. Il sert à obtenir le reste d'une division euclidienne : 13 % 4 = 1

Les opérateurs comparatifs permettent de comparer des valeurs pour faire des conditions :

- « == » | « != » la première notation vérifie si les valeurs sont égales. À l'inverse, la deuxième vérifie si les valeurs sont différentes
- « < » | « > » vérifient respectivement si la valeur de gauche est « strictement plus petite » ou « strictement plus grande » que la valeur de droite
- « <= » | « >= » sont très similaire aux opérateurs vus précédemment, mais rajoutent une subtilité. Ils vérifient respectivement si la valeur de gauche est « supérieure ou égale » ou « inférieure ou égale » à la valeur de droite

Les opérateurs logiques permettent d'enrichir les conditions :

- « && » représente un « ET ». Par exemple :

```
if (condition1 && condition2) {
    … //code s'exécutant si la condition1 ET la condition2 sont vraies
}
```

- « || » correspond à l'inverse au « OU ». Par exemple :

```
if (condition1 || condition2) {
    … //code s'exécutant si la condition1 OU/ET la condition2 est/sont
vraie(s)
}
```