1. Le langage machine

Bonjour, bienvenue dans cette troisième partie de "Informatique et ses fondements". Cette partie sera consacrée à la programmation. Après avoir vu l'algorithmique avec du pseudo-code qui décrit un algorithme, on va vraiment passer à la partie implémentation en machine de ces algorithmes.

Cette partie est découpée en trois morceaux qui sont d'une part, le langage machine. C'est-à-dire qu'est-ce que le processeur exactement est capable de comprendre. Ensuite, on ira vers la compilation des langages de plus haut niveau, plus vers le programmeur. Ensuite en parlera de bugs et on classifiera un peu les langages de programmation.

On va commencer par le plus proche du processeur de la machine de l'architecture, le langage machine. Gérard Berry l'a dit l'ordinateur est complètement con. C'est-à-dire qu'il ne sait pas faire quelque chose d'intelligent. La seule intelligence, c'est dans l'esprit du programmeur ou de la programmeuse. En fait, l'ordinateur ou le smartphone ou la tablette, tout ce genre de choses, c'est pas des objets magiques, ce sont des choses complètement bêtes. Et on a besoin de leur expliquer ce qu'elles ont besoin de faire et de leur découper, de leur prémâcher les choses à faire. Donc l'algorithme doit être prémâché dans ses plus infimes détails. Son dispositif électronique très poussé, très avancé, très compliqué, mais qui ne sont quand même que de l'électronique. Donc l'ordinateur étant très bête, il ne sait qu'obéir à des ordres simples et en fait ces ordres simples, c'est du langage machine qu'on appelle aussi de l'assembleur et qui dépend d'ailleurs du processeur que vous avez. Donc ces instructions sont extrêmement limitées. Il y en a qu'un nombre fini, assez faible d'ailleurs et donc le genre de choses qu'un ordinateur sait faire, c'est déplacer des choses de la mémoire lointaine à la mémoire proche. Il sait comparer deux valeurs. Il sait ajouter, multiplier, diviser deux valeurs. Il sait aussi sauter un autre point de programme pour faire des boucles ou des tests. Et donc c'est des choses qui sont très simples et très unitaires.

Je vais vous donner un exemple. Je vous ai mis un petit d'assembleur X86. Je vais vous l'expliquer un petit peu. Donc movl, ça sert à mettre des valeurs dans des registres. Là, on va mettre 0 dans un certain registre et ensuite 7 donc 7$ dans un certain registre. Ensuite, on a une comparaison cmpl qui va comparer la valeur 0 avec ce qu'il y a dans le registre numéro 4. Et ensuite à partir du résultat de cette comparaison, on a un test. Et en fonction du résultat de ce test, on va sauter un autre point du programme qui est le point de programme.L2. Et donc là où on fait d'autres choses. Ce qu'on voit tout de suite, c'est que ce n'est pas beau, ce n'est pas compréhensible et ça va être très difficile après de comprendre ce qu'il y a dedans. Donc la seule conclusion à faire, c'est que on n'a pas du tout envie d'écrire du langage assembleur, c'est pourquoi on va utiliser des langages de plus haut niveau et de la compilation.