Représentation des entiers relatifs

```
74  : 01001010
-74 : 11001010
```

Pour représenter des chiffres négatifs aussi bien que positif on peut réserver un bit au début de l'espace mémoire pour le signe. Si ce bit est à 1, le chiffre est négatif, sinon, il est positif.

L'un des problèmes est que les opérations arithmétiques ne sont pas facile et qu'il existe la valeur +0 et -0 qui n'ont pas de sens

- Pour savoir le nombre minimal et maximal de nombres pouvant être représenté en n bits. On va de -2^{n-1} à $+2^{n-1}$. On soustrait 1 car il y a maintenant un bit utilisé pour représenter le signe.
- Pour savoir le nombre maximal de chiffres pouvant être représentés.

$$ 2^{n-1} * 2 $$

Complément à 1

```
74 en décimal   : 01001010
-74 en comp à 1 : 10110101 (c'est l'inversion de tout les bits)
```

Pour faciliter les opérations arithmétiques nous pouvons utiliser une autre méthode. Celle d'inverser tous les bits.

Donc pour représenter $ 74_{10} $ sur 8 bits en complément à 1, on obtient `01001010` et pour changer le signe, on inverse tous les bits ce qui donne donc `10110101`. Le bit de signe est donc toujours en action.

Complément à 2

```
74                   : 01001010
-74 en complément à 1 : 10110101 (inversion de tous les bits)
-74 en complément à 2 : 10110110 (complément à 1 + 1)
```

Le complément à 2 corresponds au complément à 1 vu précédemment + 1. Mais il peut être calculé plus rapidement simplement en inversant tout à partir du premier 1 en partant de la gauche:

```
74                   : 01001010
-74 en complément à 2 : 10110110 (on inverse inverse tout jusqu'au premier 1 de gauche)
```

Revision #1
Created 27 April 2023 04:15:02 by SnowCode
Updated 27 April 2023 04:27:26 by SnowCode