Exercices : type de base

1. Entiers naturels

**Exercice 1 : conversion en vrac : entier naturel**

1. Convertir les nombres décimaux suivants en binaire et en hexa :

* 653
* 15
* 85
* 147

1. Convertir les nombres binaires suivants non signé en décimal et en hexadécimal:

* 10111011(2)
* 01010101(2)
* 10000001(2)
* 11110001(2)

1. Convertir les nombres hexadécimaux suivants en décimal et en binaire.

* FAB(16)
* 6E2(16)
* 5D8(16)
* C5A(16)

**Exercice 2 : opération en binaire**

On travaille sur un registre **8 bits .**

1. Effectuer l'addition binaire suivante : 0101 0001(2) et 0111 0111(2)
2. Faites la même chose avec les nombres suivants : 0101 0001(2) et 1111 0111(2)

Que se passe-t-il ?

1. Pour les plus motivés : Calculer la multiplication (en binaire) entre 1010 et 0110

**Exercice 3 : programmation Python addition**

Programmer la fonction permettant de faire l'addition de 2 nombres binaires (sans passer par une conversion en décimal évidemment)

**Exercice 4 : Pour aller plus loin, changement de base générique**

Créer la fonction decodage(nombre, baseN) permettant de donner en base 10 la valeur de nombre codé en baseN

1. Entiers relatifs

**Exercice 5 : décimal vers binaire**

1. Donner la représentation sur 8 bits (en complément à deux) des nombres suivants : -67 , 23, -1
2. Donner le nombre maximal en complément à 2 sur 8 bits ( valeur décimale et écriture binaire), idem pour la valeur minimale.

**Exercice 6 : binaire vers décimal**

Donner valeurs décimale des nombres binaires en complément à 2 (sur 8 bits) suivants :

N1=10001010(2) , N2=10101010(2) et N3=00101010(2)

**Exercice 7 : addition**

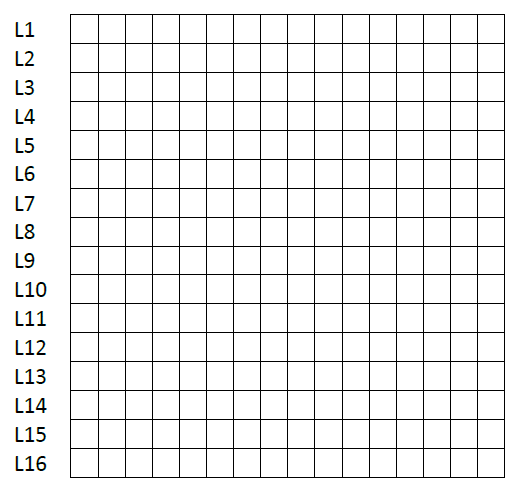
Additionner sur 8 bits les nombres suivants et commenter le résultat obtenu : 0101 0001(2) et 10101111 (2)

1. Un mélange d'entiers

**Ex 8 : Enigme en image**

L’objectif de cet exercice est de dessiner une image matricielle dans le quadrillage 16x16 grâce aux réponses aux questions de conversions entre les bases numériques. Chaque case de l’image correspond à un bit. Une ligne de l’image fait 16 cases, soit 16 bits, soit 2 octets. Lorsque le nombre est négatif, on considèrera le binaire complément à 2. Lorsque le bit est à 1, la case est noire, lorsque le bit est à 0, la case est blanche.

Vous allez devoir compléter la moitié gauche de l'image en trouvant le nombre binaire d'1 octet se cachant derrière la définition donnée ligne par ligne (cf après quadrillage page suivante). La moitié droite de l'image s'obtient par symétrie.



Que représente l'image ?

L1 : 78(16)

L2 : 135 (10)

L3 : Le double de 01011010(2)

L4 : 1011010(2)+101010(2)

L5 : F0(16)/2

L6 et L8 : MSB à 1 , le reste à 0

L7 : 58(16)+57(16)

L9 : complément à 1 de L8

L10 : 60(16)

L11 : -96

L12 : FF(16)-01110101(2)

L13 : l'opposé (négatif) de 10110110(2)

L14 : L12 +1

L15 : 27+26+25+24+2

L16: 111(2)\*4

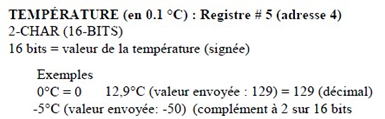
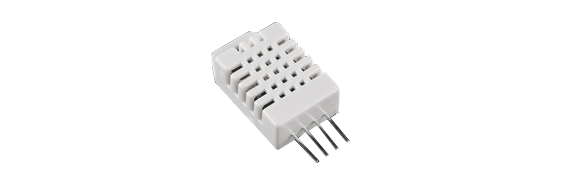
**Exercice 9 : décodage d'une trame de sonde de qualité de l'air E4000**

Les capteurs de température et d’humidité sont basés sur un protocole MODBUS (non détaillé ici), codant les informations en binaire et les envoyant via une liaison série.

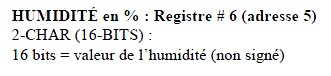
L’analyse des trames envoyées qui suit permettra de décoder l’information et de la recoder suivant la norme IEE754.

Le protocole Modbus permet à un matériel maître d’accéder jusqu’à 255 esclaves connectés sur un même bus. Chaque esclave se voit attribué une adresse qui le différencie des autres esclaves connectés sur le bus.

On ne détaillera qu’une trame ***simplifiée***:

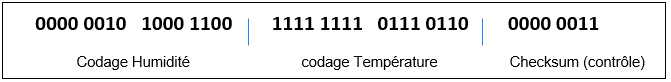


*Le capteur et sa partie électronique*



**Code contrôle** (8 bits) : somme des **octets** envoyés *(extrait doc. Constructeur)*

Un exemple de trame de données (*simplifiée, manque en tête, adresses…*) transmise en réponse par le capteur serait :



Les données transmises sont donc :

* 16 bits, avec le bit de poids fort à gauche ; représentant des dixièmes de % d’Humidité,
* 16 bits, avec le bit de poids fort à gauche ; représentant des dixièmes de la Température, codée en nombre relatif en complément à 2,
* 8 bits de code de contrôle calculé en sommant les octets 1 à 4, en partant de la gauche de la trame.

1. Le maître envoie une trame avec sa demande à un ou tous les esclaves connectés (voir description en introduction). De combien d'adresses différentes a-t-on besoin ? Combien de bits doivent être consacrés au choix de l’esclave ?
2. Décoder le taux d’humidité (RH) en décimal transmis dans la trame
3. Décoder la température en décimal transmis dans la trame
4. Quelle serait la température maximale codable ?
5. Vérifier si le code transmis possède une erreur en calculant son checksum d’après ses digits, et vérifier avec le checksum réellement transmis dans la trame.
6. La communication s’effectue en pratique en hexadécimal, donner donc le codage héxa de la trame binaire reçue donnée en exemple.