1. Supposons un nombre entier positif représenté sur 8 bits. Écrivez 4, 12, 83 et 242 en binaire et en hexadécimal.

Solution:
$4 = 00000100_b = 04_h$
$12 = 00001100_b = 0C_h$
$83 = 01010011_b = 53_h$
$242 = 11110010_b = F2_h$

2. Supposons un nombre entier positif représenté sur 16 bits. Écrivez 34211 (base 10) en base 12 (A = 10, B = 11, Par exemple, 154 en base 10 vaudrait 10A en base 12.

Solution:
34211/12 = 2850 reste 11
2850/12 = 237 reste 6
237/12 = 19 reste 9
19/12 = 1 reste 7
reste 1
Réponse : 1796B
Vérification : $1 \times 12^4 + 7 \times 12^3 + 9 \times 12^2 + 6 \times 12^1 + 11 \times 12^0 = 20736 + 12096 + 1379$
$= 34211$

3. Supposons des nombre entiers signés représentés sur 8 bits en notation complément 2 (C2). Écrivez, en hexadécimal, la valeur des nombres suivants : -1, -81, -127.

Solution:

- $-1 = FF_h$. -1, sur 8 bits en C2 est 11111111_b $(-128+64+32+16+8+4+2+1)$. Pour obtenir 11111111_b, vous pouvez aussi prendre 1 (00000001_b), l'inverser (11111110_b), puis ajouter 1 (11111111_b) en binaire est FF_h car $F = 8 + 4 + 2 + 1$.
- $-81 = AF_h$. -81, sur 8 bits en C2 est 10101111_b $(-128+0+32+0+8+4+2+1)$. Pour obtenir 10101111_b, vous pouvez aussi prendre 81 (01010001_b), l'inverser (10101110_b), puis ajouter 1 (10101111_b) en binaire est AF_h car $A = 10 = 8 + 2$.
- $-127 = 81_h$. -127, sur 8 bits en C2 est 10000001_b $(-128+0+0+0+0+0+0+1)$. Pour obtenir 10000001_b, vous pouvez aussi prendre 127 (01111111_b), l'inverser (10000000_b), puis ajouter 1 (10000001_b) en binaire est 81_h.

4. Écrivez 1_d, -1_d, -32768_d et 32767_d en binaire, en utilisant la notation complément 2, sur 16 bits. Écrivez aussi ces nombres en hexadécimal.

Solution:
1 = 0000000000000001_b ou 0x0001
-1 = 1111111111111111_b ou 0xFFFF
-32768 = 1000000000000000_b ou 0x8000
32767 = 0111111111111111_b ou 0x7FFF

5. Supposons des nombre entiers signés représentés sur 8 bits en notation complément 2. Additionnez les nombres hexadécimaux suivants, et indiquez la valeur des drapeaux "carry" et "overflow" après l'opération:

(a) $1A_h + 41_h$

Solution: $5B_h$, pas de carry ni d'overflow

(b) $6B_h + 17_h$

Solution: 82_h, pas de carry, overflow

(c) $01_h + FC_h$

Solution: FD_h, pas de carry, ni d'overflow

(d) $D3_h + 5A_h$

Solution: $2D_h$, carry, pas d'overflow

(e) $80_h + 91_h$

Solution: 11_h, carry et overflow

(f) $E1_h + E5_h$

Solution: $C6_h$, carry, pas d'overflow

6. Supposons des nombre entiers signés représentés sur 8 bits en notation complément 2. Effectuer les soustractions suivantes, et indiquez la valeur des drapeaux "carry" et "overflow" après l'opération:

(a) $A2_h - 07_h$

Solution: $9Bh$, pas d'emprunt, pas de débordement

(b) $41_h - 7A_h$

Solution: $C7_h$, emprunt, pas de débordement

(c) $21_h - 9F_h$

Solution: 82_h, emprunt, débordement

(d) $C2_h - 51_h$

Solution: 71_h, pas d'emprunt, débordement

7. : Si $C1280000_h$ est une fraction représentant un nombre sur 32 bits dans le format IEEE754, quelle est la valeur décimale de ce nombre?

Solution: En binaire, $C1280000_h$ s'écrit 1100 0001 0010 1000 0000 0000 0000 0000.
Le signe est négatif car le premier bit est 1: "1" 100 0001 0010 1000 0000 0000 0000 0000.
L'exposant vaut 130: 1 "100 0001 0" 010 1000 0000 0000 0000 0000.
La mantisse vaut $1/4 + 1/16$: 1100 0001 0 "010 1000 0000 0000 0000 0000".

Selon la norme IEEE 754, le nombre est $-1^{\text{signe}} \times 2^{\text{exposant}-127} \times 1.\text{mantisse}$
Donc, le nombre est $-1 \times 2^3 \times 1.3125 =$ -10.5

8. Écrivez 5 sur 32 bits en utilisant le format IEEE754. Vous pouvez vous inspirer de l'algorithme suivant :
   1. Déterminer le bit de signe (le bit vaut 1 si le nombre est négatif)
   2. Déterminer l'exposant. La valeur de l'exposant est $127 + \lfloor \log_2(\text{nombre}) \rfloor$.
   3. Déterminer la mantisse comme suit:
      (a) Mettre le bit de signe positif
      (b) Diviser le nombre par $2^{\text{exposant}-127}$
      (c) Soustraire 1
      (d) Pour chaque de i allant de 1 à 23
         i. Vérifier si le nombre est supérieur à 2^{-i}
         ii. Si oui, mettre le bit i de la mantisse à 1 et soustraire 2^{-i} au nombre. Notez que le bit le plus significatif de la mantisse apparait lorsque i vaut 1.

Solution: 5 = 0 10000001 01000000000000000000000.

9. (a) Donnez un exemple d'addition sur 8 bits en notation complément 2 où on obtient une retenue au neuvième bit.

Solution: Lorsque vous additionnez deux chiffres négatifs, il y a toujours une retenue sur le neuvième bit. Il y a aussi une retenue sur le neuvième bit lorsque vous additionnez un chiffre positif avec un chiffre négatif et que le résultat de l'addition est nul ou positif.

(b) Donnez un exemple de soustraction où on emprunte un neuvième bit, toujours sur 8 bits en notation complément 2.

Solution: Il y a toujours un emprunt sur le neuvième bit lorsqu'on soustrait un nombre négatif à un nombre positif. Lorsque vous soustrayez deux chiffres positifs et que le résultat est négatif, il y a aussi un emprunt sur le neuvième bit. Finalement, il y a un emprunt sur le neuvième bit si on soustrait deux nombres négatifs et que le résultat de la soustraction est négatif.

10. Vous retrouvez, en mémoire, la séquence de nombres suivante : 0x41, 0x4C, 0x4C, 0x4F. Sachant que cette séquence représente un mot codé en ASCII, quel est ce mot?

Solution: ALLO.