Université de Montréal

[Exercice notée 1]

Par

[PAUZE Alexy, 20152794 ]  
[VIGNERON Emmanuelle, 20152444 ]

Département de psychologie

Faculté des arts et des sciences

Travail présenté à [Madame, TSIKHANOVICH Alena]

dans le cadre du cours IFT [1016-B-A19],  
[Programmation 1]

[25, septembre 2019]

**Exercice 1 :**

1. Conversions de bases :

* Déterminons la puissance de chaque chiffre pour un nombre de 5 chiffres en base 7 :

Prenons par exemple le chiffre 123457 . Alors, le poids de 1 est 74, celui de 2 est 73, celui de 3 est 72, celui de 4 est 71 et celui de 5 est 70.

* Convertissons le nombre 2AA316 en décimal :

On sait que 2AA316 équivaut à 0010 1010 1010 00112.

Donc, avec la méthode du poids :

213+211+29+27+25+21+20= 1091510

2AA316 vaut 10915en décimal.

* Convertissons le nombre 4B16:
  + D’hexadécimal en décimal :

On sait que B équivaut à 11 en hexadécimal donc 4B16 en décimal est égal à :

4B16 = 4\*161+ 11\*160 = 7510

* + D’hexadécimal à binaire :

Sachant que 416 vaut 01002 et que B16 10112 donc par conséquent 4B16 est égal en binaire à 010010112.

* + D’hexadécimal à octal :

Nous avons vu plus haut que 4B16 est égal à 7510.

75/8 = 9 il reste 3

9/8 = 1 il reste 1

1/8 = 0 il reste 1

4B16 vaut donc 1138.

* Encodons l’entier 101110 avec la notation hexadécimale de JavaScript :

Afin d’encoder l’entier 101110 avec l’hexadécimale de JavaScript, nous devons dans un premier temps le convertir en hexadécimale, puis nous devons rajouter le préfixe 0x.

101110 en base 16

1011/16 = 63 il reste 3

63/16 = 3 il reste 15 (15 en base 16 devient F)

3/16 = 0 il reste 3

En hexadécimale de JavaScript, 101110 devient 0x3F3.

* Trouvons la valeur de 0xee (JavaScript) :

0x étant le préfixe, nous précisons que nous sommes en JavaScript. Il nous reste ee16.

ee = 1110 11102 = 27+26+25+23+22+21 = 23810

0xee convertit en base 10 vaut donc 23810.

1. Représentons le nombre 1710 selon la convention non signée sur 5 bits :

1710= 24 (16) + 20 (1)

1710 en base 2 sur 5 bits vaut donc 100012.

1. Trouvons les valeurs encodées par la convention complément à 2 sur 5 bits des chaines binaires :

* 01101

Puisque la chaine binaire commence par 0, nous savons que la valeur encodée est positive.

011012 = 23+22+20= 13

Donc la valeur encodée par la convention complément-à-2 sur 5 bits de la chaine binaire 01101 est 1310.

* 10011

Comme la chaine binaire commence par 1, nous pouvons en conclure que la valeur encodée est négative.

10011 en complément à 1 devient 01100. Afin de passer en complément-à-2, nous rajoutons 1 à 01100, qui devient 01101. Cela nous donne donc 1310. Cependant, la valeur est négative, donc la valeur encodée par la convention complément-à-2 sur 5 bits de la chaine binaire 10011 est -1310.

1. L’anatomie d’un nombre à virgule flottante précision double IEEE 754 (64 bits) :
   * Dans un premier temps, tout à gauche, le premier bit représente le signe (positif 0 ou négatif 1).
   * Ensuite les 11 bits suivant représentent l’exposant. Pour trouver la valeur encodée, il faut ajouter 1023 à l’exposant avant de le transformer en binaire.
   * Les 52 bits restant (à droite) représentent les chiffres se trouvant après la virgule flottante, sois la fraction.

* Trouvons l’encodage en précision double IEEE 754 (64bits) de :
  + 3.15

D’abord, il faut transformer l’entier, c’est-à-dire 3, en base 2.

310 -> ?2

3/2 = 1, reste 1

1/2 = 0, reste 1

En binaire, 3 vaut donc 112.

Ensuite, il faut convertir ce qui suit la virgule, donc le .15. On utilise donc la multiplication successive :

0.15\*2 = 0.30

0.30\*2 = 0.60

0.60\*2 = 1.20

0.20\*2 = 0.40

0.40\*2 = 0.80

0.80\*2 = 1.60

0.60\*2 = 1.20

0.20\*2 = 0.40

0.40\*2 = 0.80

0.80\*2 = 1.60

….. jusqu’à obtenir 52 bits. On remarque ici que la valeur «1001» se répète.

Ainsi, 3.15 en binaire nous donne 11.0010011001….2

Nous devons ensuite normaliser ce résultat, donc l’écrire sous la forme 1.f. On obtient donc : 1.10010011001….2 X 21

Pour encoder l’exposant (dans notre cas, l’exposant vaut 1), il faut ajouter 1023 à sa valeur, puis encoder en binaire.

e1 = 1+ 1023 = 1024

1024 – 210 (1024) = 0

On obtient donc 100000000002

|  |  |  |
| --- | --- | --- |
| 0 | 10000000000 | 1001001100110011001100110011001100110011001100110011 |

s e f

* + (-410)

Pour convertir cette valeur en binaire, il faut utiliser la méthode de complément-à-2 :

|-410| -> en base 2

4/2 = 2, reste 0

2/2 = 1, reste 0

1/2 = 0, reste 1

Nous obtenons donc 1002.

On inverse ensuite les 0 et les 1, puisque nous sommes en présence d’un chiffre négatif.

0112

+ 1

1002

On normalise ensuite la réponse : 1.00 X 22

On encode l’exposant :

e1 = 2+1023 = 1025

1025 – 210 (1024) - 20 = 0

On obtient donc e = 100000000012

|  |  |  |
| --- | --- | --- |
| 1 | 10000000001 | 0000000000000000000000000000000000000000000000000000 |

s e f

1. Trouvez la plus petite expression JavaScript (ayant le minimum de caractères incluant les parenthèses et symboles, et pas de blancs) contenant les nombres 10, 2, 3, et 4 (exactement une fois chaque), et les opérateurs +, - et \* (autant de fois que vous voulez), dont la valeur est 9.

10-2+4-3 = 9