

Cours complet

Jean-Christophe Dubacq

S1 2016

1 Représenter une information

1.1 Du sens à la mesure

1.1.1 Qu'est-ce que l'information

- Q1** Proposez différents symbolismes utilisés pour noter un nombre. Donnez l'exemple de leur notation avec le nombre 9. Donnez des inconvénients de votre méthode.
- Q2** Travaillez en paire (ou triplettes). Proposez une méthode pour transmettre d'une personne à une autre le résultat d'un lancer de dé (lancer caché par la première personne, la deuxième doit pouvoir énoncer le résultat). Votre méthode fonctionne-t-elle si le dé comporte 20 faces ? Et si le dé est à six faces mais étiqueté par des couleurs ?

1.1.2 Digital ou analogique ?

- Q3** Est-ce que les données suivantes sont digitales ou analogiques :
- Le fait d'avoir un rendez-vous à une certaine heure un certain jour
 - La pression de l'air
 - Le résultat (stable) d'un dé
 - Votre nom de famille
 - Votre nombre de frères et sœurs
 - Votre taille
 - La couleur de vos yeux

1.2 Mesurer l'information

1.2.1 Conversions

Q4 Convertissez 24×10^8 bits en Go. $24 \times 10^8 / 8 = 3 \times 10^8 = 0,3 \text{Go}$

Q5 Convertissez 2^{16} octets en Mib. Donnez une approximation en Mb. Quel est l'ordre de grandeur de l'approximation faite ? $8 \times 2^{16} = 2^{19} = 0,5 \text{Mib}$, soit environ 0,5 Mb, à 5% près.

Q6 Un élément d'ordinateur est capable d'émettre 1024 bits en 0,5 nanosecondes. Quel est le débit (quantité d'information divisée par le temps) de cet élément en bits par secondes ? Quelle est la bonne unité pour ce débit ? $1024 \text{ bits en } 0,5 \text{ nanosecondes} = 1024 / (0,5 \times 10^{-9}) = 2048 \times 10^9 \text{ bits/secondes}$. L'unité appropriée est sans doute le Tb/s (et pas le Tib/s, car si on a un 2048, on a pas une pure puissance de 2, et la division du résultat par 2^{40} n'est pas un entier du tout...).

1.3 De l'analogique au digital

1.3.1 Signal électrique

Q7 Un signal électrique qui va de 0 à 2,559 V est quantifié sur un quantum de 0,01 V. Quel est le nombre de quanta ? Quelle quantité d'information est transportée par un quantum ?

8 bits par échantillon. La valence est de 256.

Q8 Ce signal est périodique, et se décompose avec des fréquences maximales qui vont jusqu'à 10 kHz. Quelle est le débit d'information nécessaire pour reconstituer ce signal à l'identique ?

Chaque échantillon prend un octet. f_e doit être plus grand que 2×10^3 échantillons/s. Donc le débit 20 ko/s (binaires).

Q9 Quelle est la taille de l'information nécessaire pour enregistrer ce signal pendant une heure ?

*$3600 \text{ s} * 20 \text{ ko/s} = 72 \text{ Mo}$ (décimaux)*

1.3.2 CD audio

Q10 Un CD audio contient de la musique échantillonnée en stéréo sur 16 bits par piste à 44100 Hz (nombre d'échantillons par seconde). Il dure environ 80 minutes. Calculez (de tête) l'ordre de grandeur de la quantité d'information écrite dans un CD audio.

Environ 800 Mo. En vrai : 807,5 Mio ou 846 Mo. 88200 octets par seconde par piste, soit 167400 octets/s, soit 10 Mo/minute.

2 Représenter un nombre

2.1 Les systèmes de numération

2.1.1 Puissances de 2

Q11 Écrivez la liste de toutes les puissances de 2, de 2^{-4} à 2^{16} .

0,06125 – 0,125 – 0,25 – 0,5 – 1 – 2 – 4 – 8 – 16 – 32 – 64 – 128 – 256 – 512 – 1024 – 2048 – 4096 – 8192 – 16384 – 32768 – 65536

Et jusqu'à 1024, c'est à savoir par cœur. Aucune discussion à avoir.

Q12 Écrivez une table de conversion des chiffres hexadécimaux et octaux vers le codage naturel écrit en binaire (4 bits ou 3 bits).

0 – 0000, 1 – 0001, 2 – 0010, 3 – 0011, 4 – 0100, 5 – 0101, 6 – 0110, 7 – 0111, 8 – 1000, 9 – 1001, A – 1010, B – 1011, C – 1100, D – 1101, E – 1110, F – 1111. Et pour les octaux, la même chose de 0 à 7 (on peut mettre seulement sur 3 bits pour l'octal, en supprimant le 0 initial).

C'est à savoir par cœur. Aucune discussion à avoir.

2.1.2 Conversions

Q13 Écrivez en binaire et en hexadécimal les nombres décimaux suivants : 28 ; 149 ; 1285.

1 1100 ; 1001 0101 ; 101 0000 0101 1001. 0x1C ; 0x95 ; 0x505.

Q14 Convertissez en décimal les nombres suivants : 0x48 ; 0xA1C ; 0b1010010010011111.

72 ; 2588 ; 42143.

Q15 Comment trouver midi à quatorze heures ?

En base 8. Ne pas insister sur cette question.

2.2 Des entiers naturels aux réels

2.2.1 Changements de base

Q16 Écrivez en binaire et en hexadécimal les nombres décimaux suivants : 0,3125 ; 164,3125.

0,0101 ; 1010 0100,0101. 0x0,5 ; 0xA4,5.

Q17 Convertissez en décimal le nombre suivant : 0b1010,0011.

10,1875.

2.3 Codage des entiers

2.3.1 Codage d'entiers

Q18 Ce tableau comporte des cases inutilisées. Complétez-le :

Décimal	Écriture Binaire	Type de codage	Codage (binaire)	Codage (hexa)
-18	-1 0010	VA+S (8 bits)	1001 0010	0x92
424	110101000	NAT (16 bits)	0000000110101000	0x01A8
-138	-1000 1010	C2 (16 bits)	1111111101110110	0xFF76
-115	-111 0011	C1 (8 bits)	10001100	0x8C
-4197	-10000 01100101	VA+S (24 bits)	100000000001 000001100101	0x801065
-84	-101 0100	C1 (8 bits)	1010 1011	0xAB
341	101010101	NAT (8 bits)	Impossible!	XXX

Rappel : pour décoder un complément à 2, il faut : inverser les bits si signe=1, ajouter 1 à la valeur obtenue, on a la valeur absolue. Pour encoder un complément à 2, il faut soustraire 1 à la valeur à encoder, inverser les bits (si le signe est égal à 1). Si le signe est positif, rien à faire de différent de NAT.

2.4 Codage des réels

2.4.1 Codage IEEE754

Q19 Ce tableau comporte des cases inutilisées. Complétez-le :

Décimal	Binaire	Virgule flottante	E	Codage IEEE754			Hexa
				S	E (8b)	M (23b)	
19,5	10011,1	$1,00111 \times 2^4$	131	0	10000011	00111 0...0 18 fois	419C0000
-7,5	<i>111,1</i>	<i>$1,111 \times 2^2$</i>	<i>129</i>	<i>1</i>	<i>10000001</i>	<i>111 0...0</i> 20 fois	<i>C0F00000</i>
-46,25	<i>101110,01</i>	<i>$1,0111001 \times 2^5$</i>	<i>132</i>	<i>1</i>	<i>10000100</i>	<i>0111001 0...0</i> 16 fois	<i>C2390000</i>
0,3125	<i>0,0101</i>	<i>$1,01 \times 2^{-2}$</i>	<i>125</i>	<i>0</i>	<i>01111101</i>	<i>01 0...0</i> 21 fois	<i>3EA00000</i>
-0,1875	<i>0,0011</i>	<i>$1,1 \times 2^{-3}$</i>	<i>124</i>	<i>1</i>	<i>01111100</i>	<i>1 0...0</i> 22 fois	BE400000
$+\infty$	—	—	255	0	11111111	<i>0...0</i> 23 fois	7F800000
0	<i>0</i>	—	<i>0</i>	<i>0</i>	<i>00000000</i>	<i>0...0</i> 23 fois	<i>00000000</i>
$-26,375 \times 2^{40}$	<i>$-11010,011 \times 2^{40}$</i>	<i>$-1,1010011 \times 2^{44}$</i>	<i>171</i>	<i>1</i>	<i>10101011</i>	<i>1010011 0...0</i> 16 fois	<i>D5D30000</i>

3 Les opérations

3.1 Les entiers

3.1.1 De la fonction à l'algorithme

La numération grecque (simple) est proche de la numération romaine que vous connaissez : on note les nombres comme suit :

1	5	10	50	100	500	1 000	5 000	10 000	50 000
I	Γ	Δ	Γ _Δ	H	Γ _H	X	Γ _X	M	Γ _M

C'est à la différence près que l'on a pas de règle soustractive : le nombre 4 s'écrit IIII, pas IIΓ. La position des chiffres n'a théoriquement aucune importance, mais on les classait dans l'ordre décroissant de valeur.

Q20 Ce système est-il un système de numération positionnelle ? *Non*

Q21 Écrivez votre âge et votre date de naissance en numération grecque.

XΓ_HHHHHΓ_ΔΔΔIII et ΔΔΔΓIII pour moi.

- Q22** Écrivez un algorithme d'addition des nombres représentés en numération grecque. Est-ce que cet algorithme est le même qu'en décimal ?
On prend les nombres à additionner, on les colle les uns aux autres, on rassemble les chiffres identiques. Ensuite on répète en partant de la droite : tant que l'on a cinq (ou deux) chiffres identiques, on les enlève et on fabrique un chiffre suivant à leur place. C'est un exemple, il y en a plusieurs possibles.
- Q23** Faites l'addition de votre âge et de votre année de naissance avec votre algorithme (vous devriez obtenir XXΔIII ou XXΔIII). De quelle représentation partez-vous ? *On part des représentations grecques, bien sûr. Ou alors des représentations en décimal, que l'on convertit, mais 1- ça a déjà été fait et 2- ça ne doit pas faire partie de l'algorithme.*
- Q24** Faites la même chose en décimal. De quelles représentations partez-vous ? Est-ce que l'algorithme est le même ? Est-ce que la fonction calculée est la même ?
La représentation en décimal sert de départ, l'algorithme est différent (même si un peu similaire), mais le résultat (et donc la fonction calculée) est identique.

3.1.2 Calcul en binaire et hexadécimal

- Q25** Faites les additions en binaire : $0b1101\ 0101 + 0b1110\ 0101$; $0b1,1 + 0b110 + 0b100,1 + 0b111,1 + 0b1010,1 + 0b100,1$.
1 1011 1010 et 10 0010, 1 (ils peuvent vérifier en décimal).
- Q26** Faites les opérations suivantes en hexadécimal : $0x122 + 0x233$; $0x87 + 0x54$; $0x18 + 0x9$; $0xED + 0xED$; $0x100 - 0x3$.
355, DB, 21, 1DA, FD
- Q27** Faites la multiplication suivante : 17×129 à la fois en décimal et en binaire.
2193 et 100010010001.
- Q28** Faites la multiplication suivante en binaire : 110110×1101 .
*1010111110 (54*13=702)*

3.2 Addition et codage

3.2.1 Limites de la multiplication

Expliquez pourquoi le résultat d'une multiplication de deux nombres représentés dans l'un des 4 codes classiques est toujours représentable à condition de doubler la taille du code.

Il suffit de regarder la valeur maximale autorisée par un code et de l'élever au carré (produit le plus grand possible).

3.2.2 Addition en C2

- Q29** Faites les opérations suivantes en transformant les nombres au préalable en codage C2 sur 8 bits (résultat aussi en C2 sur 8 bits) :

- 45+17
- 45-17 (soit 45+(-17))
- -17-17
- 17-45
- 221+45

Dites aussi si le résultat obtenu est correct et s'il est représentable.

00101101+00010001=...

00101101+11101111=...

11101111+11101111=11011110

...+...=...

11011101+00101101=00001010

Ils sont tous corrects s'ils sont représentables, donc tous sauf le dernier.

3.3 Les champs de bits

3.3.1 Tables de vérité

Q30 Faites une table qui montre toutes les paires d'arguments possibles pour les opérateurs AND, OR, XOR et qui montre le résultat à côté.

A	B	A×B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

3.3.2 Opérations booléennes

Q31 Que vaut 0b10000110 AND 0b11101001 ? *10000000*

Q32 Que vaut 0b10000110 OR 0b11101001 ? *11101111*

Q33 Que vaut 0b10000110 XOR 0b11101001 ? *01101111*

Q34 Que se passe-t-il si on calcule (a est une variable booléenne) : $a + 0$? $a + 1$? $a \times 0$? $a \times 1$? $a + a$? $a + a + a + a + a + a$?

Q35 Démontrez que $a + ab = a$; *Vu en cours. $a + ab = a(1 + b) = a$.*

Q36 Démontrez que $a + bc = (a + b)(a + c)$; $(a + b)(a + c) = a + ab + ac + bc = a + ac + bc = a + bc$

Q37 Démontrez que $a + \bar{a}b = a + b$; *Vu en cours.* $a + \bar{a}b = a(b + \bar{b}) + \bar{a}b = ab + a\bar{b} + \bar{a}b + ab = a(b + \bar{b}) + b(a + \bar{a}) = a + b$. NB : $x = x + x$.

3.3.3 Analyse d'un masquage

Dans un champ de bits qui contient $a = 0b11001001$, on veut faire les choses suivantes :

Q38 On veut vérifier si le bit 0 est actif ou non. Décomposez l'opération.

Q39 On veut changer le bit 1 en 1 et le bit 3 en 0. Décomposez les opérations qui permettent de le faire.

Q40 Changez le bit 5, en expliquant les valeurs intermédiaires.

3.3.4 Analyse de touches

Dans un système, la fonction `keyEvent ()` renvoie une valeur entière sur 16 bits (dont 5 ignorés) :

- Les 8 premiers bits correspondent au numéro de la touche sur le clavier (pour les touches ordinaires)
- Le 9^e bit correspond à la touche SHIFT (1 : pressée, 0 : pas pressée)
- Le 10^e bit correspond à la touche CONTROL (1 : pressée, 0 : pas pressée)
- Le 11^e bit correspond au fait d'appuyer sur une touche (1) ou de l'avoir juste relâchée (0)

Q41 Écrivez un programme qui appelle cette fonction (`a=keyEvent ()`) puis qui en fonction de `a` affiche un texte du genre : « Vous venez de lâcher la touche 27 en ayant SHIFT appuyé et CONTROL lâché »



4 Les textes

4.1 De l'écrit au binaire

4.1.1 La table ASCII

Trouvez dans la table ASCII :

1. Le caractère de code 0x41
2. Le caractère de code 0x30
3. Le caractère *a* et *A*. Comparez l'écriture binaire des codes numériques correspondants.
4. Le caractère de code 0x20. Quel est-il ?
5. Le caractère *retour chariot* (son nom est NEWLINE ou NL).

Comment passe-t-on d'une lettre à la suivante ? D'une majuscule à une minuscule ?

4.2 Jeux de caractères et codages

4.2.1 Codage nationaux et Mojibake

Soit le texte : Coefficient marée trop fort pour livraison tomates cœur-de-bœuf

1. Identifiez dans ce texte les ligatures linguistiques et les ligatures esthétiques

Le œest une ligature linguistique (la seule en français avec le œ, beaucoup plus rare). Le ffi de coefficient est une ligature esthétique. Notez que le oe de coefficient n'est pas une ligature.

2. Est-il possible de représenter ce texte dans le jeu de caractères ASCII ?

Non, à cause du œ. Le œn'est d'ailleurs pas une lettre en français ; par contre le ch en espagnol l'est (voir n'importe quel dictionnaire tchèque, où ch n'est pas entre cg et ci, mais juste après hz ; c'était aussi le cas en espagnol jusqu'à une réforme en 1994).

3. Dans le jeu de caractère ISO-8859-15 (dit latin-9), il est possible de coder ce texte. Chaque caractère est alors codé par un octet unique. Quelle est la taille du fichier qui contient uniquement ce texte ? *85 octets (et non 86)*

4. Un polonais lit sur son vieil ordinateur le texte précédent. Il voit qu'une des lettres a été remplacée par " (c'est un double accent aigu, comme dans Erdős, et pas un tréma comme dans Gwenaël). Laquelle et pourquoi ? S'il renvoie le texte tel quel a son correspondant français du début, que verra le français et pourquoi ?

En vérité, il y a de bonnes chances qu'il lise son texte comme étant du ISO-8859-2, et non pas du ISO-8859-15, donc il verra un double accent aigu à la place de sa lettre. Mais le contenu du fichier est inchangé ; s'il est renvoyé au français, le texte apparaîtra normalement.

L'encodage d'un fichier ne peut pas être deviné simplement comme ça (il faut faire une analyse des mots pour déterminer la langue et donc l'encodage probable).

NB : bien sûr, il peut y avoir des problèmes ; les logiciels de courrier indiquent parfois l'encodage des pièces jointes, même s'il a été mal deviné ; certains éditeurs de texte sauvegardent les textes dans un encodage différent de celui qui a été deviné pour l'ouverture... bref, les problèmes peuvent exister. Mais le fichier n'est a priori pas modifié sauf logiciels qui ne fonctionnent pas bien.

4.2.2 UTF8

1. Le caractère de numéro 0x0041 (A) est codé par quel(s) octet(s) en UTF-8 ? *0x41*
2. Le caractère de numéro 0x00E9 (é) est codé par quel(s) octet(s) en UTF-8 ? *0xc3 0xa9*
3. Le caractère de numéro 0x0F03 (འཇམ་མགོན་པོ་འཕགས་པ་) est codé par quel(s) octet(s) en UTF-8 ? *0xe0 0xbc 0x83 C'est le caractère GTER YIG MGO 'IM GTER SHEG MA en tibétain (à vos souhaits).*
4. Le caractère de numéro 0x12084 (𐎠𐎢𐎡𐎣) est codé par quel(s) octet(s) en UTF-8 ? *0xf0 0x92 0x82 0x84 (4 octets) C'est le caractère DOUN en cunéiforme (babylonien).*

5. Dans un fichier codé en UTF-8, on trouve les six octets suivants. Combien de caractères sont réellement codés dans ce texte ?

3 caractères (un sur trois octets, un sur deux, un sur un)

Ce code a l'avantage que l'on peut aussi trouver facilement en lisant une suite d'octets représentant de l'UTF-8 combien d'octets occupe chaque caractère codé : on les écrit en binaire, et on sait automatiquement avec le premier octet dans quelle ligne on se trouve, et donc combien d'octets sont utilisés pour le caractère. On peut alors sauter au caractère suivant facilement.

6. L'anglais n'utilise que des caractères dont le numéro est dans la première ligne, et est codé traditionnellement en ISO-8859-1 (1 caractère = 1 octet). Le français utilise 5% de caractères de la deuxième ligne (le reste de la première), et est codé pareil (1 caractère = 1 octet). L'arabe (le russe, l'hébreu, le grec) sont aussi codés traditionnellement par 1 caractère = 1 octet, et comportent 95% de caractères de la deuxième ligne (le reste de la première ligne). Le chinois, en revanche est traditionnellement codé en BIG5 (1 caractère = 2 octets). Les textes chinois sont à 99% des caractères de la troisième ligne (le reste de la première ligne).

Pour un texte de 1000 caractères codé en UTF-8, combien d'octets seront utilisés en moyenne pour un texte anglais, français, russe et chinois ?

Anglais : 1000 octets. Français : 1050 octets. Russe : 1950 octets. Chinois : 2980 octets.

7. Quel est en chinois l'augmentation de la taille du texte par rapport au codage traditionnel ? $(2980 - 2000) / 2000 = 980 / 2000 = 49\%$

4.3 Les chaînes de caractères

4.3.1 Les échappements en C

Dessinez quelle est la structure en mémoire des chaînes C suivantes ? Comment sont elles affichés ?

1. "Toto"
2. "Bonjour le monde\n"
3. "Acheter:\n\tponey\n\tporte-avions\n"
4. "\303\251\n"
5. "\U20AC" (symbole euro)
6. "\0"



Une bizarrerie historique du C/C++ fait que certaines séquences sont remplacées avant compilation par d'autres caractères :

Trigraphe	??(??) ??<
Remplacement	[] {

7. "Hello??!"
8. "Bye??/n"

5 Les séquences de codage

5.1 Les formats complexes

5.1.1 Types simples ou composés

Q42 Identifiez dans les types suivants lesquels sont susceptibles d'être des types de base et lesquels sont plutôt des types construits par assemblage :

- Un nombre entier positif ou nul
- Un nombre complexe
- Un point dans l'espace
- Un nombre avec un très grand nombre de chiffres non fixé à l'avance
- Un intervalle
- Une date
- Un étudiant (nom, prénom, date de naissance)
- Un caractère
- Une chaîne de caractères

Un nombre entier positif ou nul est en général un type simple, de même que un caractère. Pour tout le reste, ce sont des types composés. Un nombre complexe, un point dans l'espace sont des tableaux de nombres (2 ou 3), de même qu'un intervalle. Une date, c'est souvent une série de nombres, même si on peut se fixer un point de référence et compter par exemple le nombre de secondes depuis ce point de référence. Une chaîne de caractères ou un nombre avec un très grand nombre de chiffres non fixé à l'avance, c'est une liste d'éléments simples (caractères ou chiffres). Quant à l'étudiant, ce sont des types de données hétérogènes, donc forcément composées.

5.1.2 Une date

Q43 Décrivez à partir de quels éléments on peut composer une donnée qui représente un moment précis de la journée. *Voir plus loin pour une correction possible. Le fuseau horaire se discute (et dans ce cas, c'est une chaîne de caractères ou un code dans une liste).*

Q44 Discutez les éléments précis selon que l'on considère qu'un moment est pris à la seconde près ou beaucoup plus précis. *Le nombre de secondes peut être soit un entier, soit un flottant.*

5.2 La représentation en mémoire

5.2.1 Stockage d'une date (suite)

Une date est composée des éléments suivants :

- Une année (disons de -2 milliards à +2 milliards)
- Un mois, un jour du mois

- Un fuseau horaire qui est une « adresse »
- Une heure, une minute (entiers)
- Un nombre de secondes qui est un flottant simple précision

Q45 Dites quels sont les types de base du C à utiliser pour coder cette information, d'après les limites connues de stockage pour chaque type. Utilisez les tailles les plus petites possibles.

*Un **int** 32 bits pour l'année, des entiers très courts (donc **char** ou **unsigned char**) pour mois, jour du mois, heure, minute. Une adresse pour le fuseau horaire qui est en fait une chaîne de caractères. Et un **float** pour les secondes.*

Q46 Donnez leur noms à la fois dans un modèle 32 bits et un modèle 64 bits.

*Voir dans le tableau, mais **int** remplit bien sauf pour ILP64 (il s'appelle **_int32**).*

Q47 Si on avait voulu aller de -5 milliards à +5 milliards d'années, quel type aurait-on dû utiliser ?

*Un entier sur 64 bits donc un **long int** (voire un **long long int** sous Microsoft).*

Q48 Si les données sont dans l'ordre indiqué dans un type composé, précisez à quel moment les contraintes d'alignement provoquent des « trous » dans la structure.

On commence par 4 octets, puis 2 fois un octet. Puisqu'une adresse fait 4 ou 8 octets, il faut là un trou qui fait 2 octets (alignement sur 4 octets) ou 6 octets (alignement sur 8 octets).

*Après, on a encore 2 octets (des **char** suffisent pour heures et minutes), suivi d'un **float** donc sur 4 octets. Encore 2 octets de trous sont donc nécessaires.*

Q49 Quelle est la taille totale de la structure (avec les trous) ?

En 32 bits pour une adresse : $4+2+2+4+2+2+4=20$ octets.

En 64 bits pour une adresse : $4+2+2+8+2+2+4=24$ octets.

Si on veut aller jusqu'à 5 milliards, il faut rajouter 4 octets à chaque fois.

5.2.2 Stockage d'une date (suite)

Q50 On veut stocker la date du 24 décembre -4 à 21h45 dans un système 32 bits. Calculez les valeurs à stocker en hexadécimal (hormis l'adresse du fuseau horaire). Vous prendrez comme valeur d'adresse pour le fuseau horaire 0x12345678.

*L'an -4 c'est **0xFFFFFFFFC**. Le mois, c'est **0xC**, le jour c'est **0x18**, l'heure c'est **0x15** et les minutes c'est **0x0x2D**. Les secondes c'est 0, et 0 en float c'est **0x00000000**.*

Q51 Écrivez, les uns après les autres, les octets qui composent cette date si on est dans un système 32 bits *big-endian*

***FF FF FF FC 0C 18 ????** **12 34 56 78 15 2D ????** **00 00 00 00**. Les ?? sont du bourrage (ça peut être n'importe quelle valeur).*

6 Les médias

6.1 Format des images

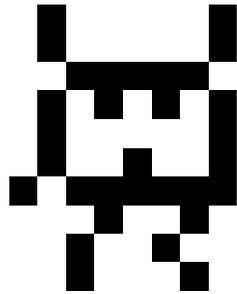
6.1.1 Décodage d'un fichier PBM

Voici la séquence d'octets qui compose un fichier PBM :

50 34 0a 38 20 31 30 0a 41 41 3e 55 41 49 bf 12 24 22

Q58 Repérez l'entête du fichier, et traduisez-là en ASCII. *P4(retour chariot)8(espace)10(retour chariot)*

Q59 Quelle est la taille de cette image (en pixels) ? *8 par 10*



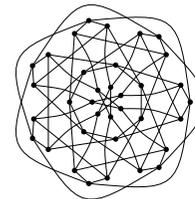
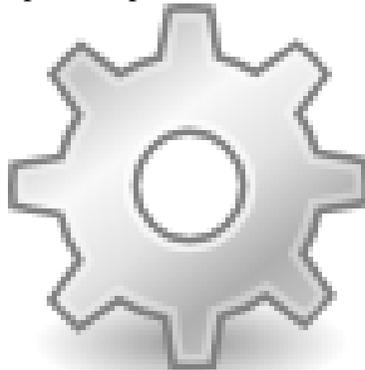
Q60 Dessinez le fichier résultant.

Q61 Quelle est la taille (minimale) de l'entête et la taille des données ? *5 + 2 × 1 = 7 pour l'entête et 10 octets pour les données*

Q62 Même question pour une image 8000 par 8000. *5 + 2 × 4 = 13 pour l'entête et 8000000 octets pour les données. L'entête est donc négligeable.*

6.1.2 Choix de format d'image

Q63 Voici quatre images. Imaginez le format le plus adapté à chacune d'entre elles. Expliquez votre choix.



HOME ORGANIZATION TIP:
JUST GIVE UP.

JPEG (photo), PNG (mais pixelisé) ou vectoriel pour un icône, Vectoriel, PNG (peu de couleurs : palette petite, compression idéale, dessin au trait pas idéal en JPG).

6.1.3 Palette

Q64 Une image 1000×1000 utilise 3 octets pour décrire la couleur de chaque pixel. Calculez la taille occupée par les données de cette image en PNG.

3 Mo

Q65 Cette image n'a que 256 couleurs au total. On peut utiliser une palette de couleurs. Calculez la taille de la palette et la taille des données de l'image utilisant la palette.

1 Mo+768 o pour la palette

6.2 Les couleurs

6.2.1 Décomposition de couleurs

Donnez des composantes couleur plausibles RGB des couleurs suivantes. Utilisez la notation HTML.

— Rouge, vert, bleu

— Cyan, magenta, jaune

— Blanc, noir

— Gris 50%

— Marron foncé, rose pâle, orange vif

FF0000, 00FF00, 0000FF

00FFFF, FF00FF, FFFF00

FFFFFF, 000000

808080

200000 (ou autre faible quantité de rouge), FFE0E0 (rouge un peu plus que les autres, mais très blanc), FF7F00 (ou autre chose équivalente : mélange jaune+rouge)

6.2.2 Scanner

Un scanner scanne en RGB à une résolution de 1200 points par pouce (dans les deux directions). Pour simplifier, on considérera qu'il y a une surface de 10 pouces × 6 pouces scannable. Chaque couleur est scannée en 12 bits. Quelle est la quantité d'information résultant de chaque scan ?

1200×1200×60×12×3=3 110 400 000 soit environ 389 Mo.

6.2.3 Conversion HTML-CMJ

La trichromie consiste à n'utiliser que trois couleurs et faire le noire par mélange des autres couleurs. Dans ce cas, la formule est simple : la proportion d'une encre est 100% - la proportion de la couleur complémentaire.

Convertissez la couleur suivante en CM : #FA0140. Quel genre de teinte est-ce ? Est-elle très saturée ?

FFFFFF-FA0140=05FEBF en CMJ. Elle est assez saturée (beaucoup d'encre magenta et pas mal de jaune). C'est quelque chose assez rouge (un peu violacé).

6.2.4 Vitesse d'impression

Une imprimante en quadrichromie est capable d'imprimer 6 pages par minutes, en 1200 points par pouce en mode RVB 8 bits par composante. Pour simplifier, on considérera qu'il y a une surface de 10 pouces × 6 pouces imprimable. Quelle est la quantité d'information qu'on doit fournir à l'imprimante pour une page ? Pour une minute d'impression ?

Même chose que précédemment, mais seulement 8 bits, donc chaque page représente : 259 200 000 octets, soit environ 1,5 Go par minute.

6.3 Les sons

6.3.1 Compression audio MP3

Le codec MP3 permet de compresser le signal sonore dans une grande variété de débits finaux (après compression), le plus commun étant 128 kb/s. La fréquence d'échantillonnage est quasi-toujours 44,1 kHz. **Calculatrice autorisée.**

Q66 Quel est le débit non compressé pour de l'audio stéréo 16 bits ? *16 bit/sample × 44100 samples/second × 2 channels / 1000 bits/kilobit=1411,2 kb/s.*

Q67 Quel est le taux de compression du format MP3 le plus classique (débit final 128 kb/s) ? *1411,2/128=11,02 (et des poussières)*

Q68 Et avec le format plus généreux à 320 kb/s au final ? *1411,2/320=4,41*

6.4 Les films

.1 Le jeu du fakir

.2 La table ASCII