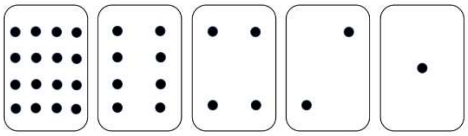
Comment coder l'information ?

Vincent IMBERT – Lycée Alexis Monteil d'après séquence proposée par N Tourreau @IANum_Techno

Nom Prénom : Classe :

Coder des chiffres



À partir des cartes distribuées, répondre aux questions :

Que constates-tu concernant les points qui figurent sur les cartes ? **Chaque carte contient le double de points que la précédente.**

Combien de points devrait avoir la prochaine carte à gauche ? **32** et la suivante ? **64, 128**

Nous allons utiliser ces cartes pour représenter des nombres. Pour cela, retourner certaines et additionner les points qui restent visibles.
Afficher le nombre 6 (cartes 4 points et 2 points), puis le nombre 15 (cartes 8, 4, 2 et 1 points), puis 21 (16, 4 et 1).

Lorsqu'une carte d'un nombre binaire n'est pas visible, elle est représentée par un **0**. Lorsqu'elle est visible, elle est représentée par un **1**. C'est le système de **numération binaire**, contrairement à l'écriture des nombres du **système décimal** qui utilise les chiffres de 0 à 9.

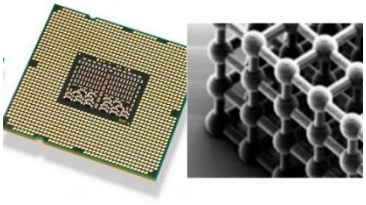
Trouver comment obtenir 3, 12, 19. Existe-t-il plusieurs moyens d'obtenir un nombre ?
3 => 00011 ; 12=>01100 ; 19 => 10011 Non, il n'y a qu'une seule combinaison possible.

Quel est le plus grand nombre que l'on peut obtenir ? **31** Quel est le plus petit ? 0

Y a-t-il un nombre compris entre le plus grand et le plus petit que l'on ne puisse pas obtenir ? **Non**
Inversement, trouver combien fait 10111 ? **23** 11101 ? **29**

Le code binaire

À partir de la vidéo jusqu'à la 4min43s expliquer d'où vient le codage binaire :



Le microprocesseur a besoin de courant électrique. Le microprocesseur est composé de millions de transistors (sorte de minuscule interrupteur électrique hyper rapide). Ainsi ces transistors laissent passer ou ne laissent pas passer le courant électrique et cela des millions de fois par seconde. C'est comme cela qu'un microprocesseur peut faire des opérations logiques, stocker des informations et exécuter ce qui est demandé par un utilisateur.

Coder des caractères

Ouvrir les fichiers snt.txt et snt.doc dans un éditeur de texte.

Que remarques-tu concernant leurs contenus ? **Je remarque que ces deux fichiers ont un contenu identique** et concernant la taille de ces deux fichiers ? **mais une taille très différente.**

Ouvrir les fichiers snt.txt et snt.doc à partir de l'application en ligne hexed.it

Que remarques-tu concernant le nombre d'octets nécessaires (Taille du fichier) pour chaque fichier ? **Le fichier snt.txt a nécessité 49 octets ~ 1ko pour être codé, en revanche le fichier snt.docx a nécessité 12996 octets =12.996ko=13ko**

Quelle est la taille de chaque fichier ?

1 ko

13 ko

Quelle est l'unité ? **ko Kilo octet**

D'où vient ce mot ? **1 ko correspond à 1000 octets et 1octet correspond à 8 bits. Une information codée sur 8 bits permet de compter de 0 à 255 soit 256 possibilités.**

Comment expliquer la différence de taille ? **Etonnamment le fichier *.txt est beaucoup plus léger (1 ko) que le fichier *.docx (13 ko). Le fichier *.docx doit également comporter des informations supplémentaires (telle que la taille de caractère, la police, l'interligne, etc.) qui explique que ce fichier soit plus lourd.**

Qu'est-ce que le code ASCII ? 

American Standard Code Information Interchange [aski]. C'est une norme depuis 1960 pour le codage des caractères sur 7 bits.

En adoptant le même codage, tous les systèmes informatiques peuvent alors échanger des caractères.

Caractère du clavier->codage en hexadécimal->binaire

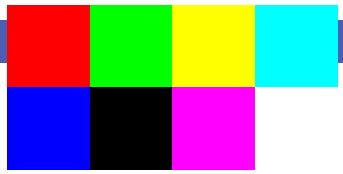
Aide : vidéo à partir de 5'20

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

En utilisant la table de conversion ASCII donne l'équivalent décimal puis binaire pour le « s » minuscule, le « S », le « N » et le « T » majuscules :

Caractère	s	S	N	T
Décimal	**115**	**83**	**78**	**84**
Binaire	**0 1 1 1 0 0 1 1**	**0 1 0 1 0 0 1 1**	**0 1 0 0 1 1 1 0**	**0 1 0 1 0 1 0 0**

Coder une image



Ouvrir le fichiers 8px.bmp à partir de l'application en ligne HexEd.it

Noter les résultats obtenus :

	Taille (octets)	Code (Hexadécimal)				Signification
En-tête Valeur constante : **Les 54 premiers octets**	1ère en-tête	42	4D			Caractères B (42) et M (4D) indiquant **le type de** fichier. Ici **bitmap**
		4E	00	00	00	**Taille** du fichier 4E 00 00 00 = **78** octets
		00	00	00	00	Réservé (toujours à 0)
		36	00	00	00	Offset ou Décalage de l'image = **$36_{(16)}$=$54_{(10)}$** Les informations concernant l'image commencent au 54ème octet.
	2ème en-tête	28	00	00	00	
		04	00	00	00	**Largeur** de l'image = **4 pixels**
		02	00	00	00	**Hauteur** de l'image = **2 pixels**
		01	00			Nombre de plans utilisés =**1** (Cette valeur vaut toujours 1)
		18	00			**$18_{(16)}$ = $24_{(10)}$** Nombre de **bit** par pixel = **24** soit **3** octets (1, 4, 8, 16, 24 ou 32)
		00	00	00	00	Méthode de compression : 0 pas de compression
		00	00	00	00	Taille de l'image 00000018 = 24 octets = 8 (pixels) x 3 (octets par pixel)
		C4	**0E**	00	00	Résolution horizontale = **19 614** pixels par mètre
		C4	**0E**	00	00	Résolution verticale = **19 614** pixels par mètre
		00	00	00	00	0, cela signifie que la palette entière de couleurs est utilisée.
		00	00	00	00	Nombre de couleurs important (Ce champ peut être égal à 0 lorsque chaque couleur a son importance)
Code Image Longueur (px) X Largeur (px) : **24 octets**		FF	00	00		Bleu
		00	00	00		Noir
		FF	00	FF		Magenta
		FF	FF	FF		Blanc
		00	00	FF		Rouge
		00	FF	00		Vert
		00	FF	FF		Jaune
		FF	FF	00		Cyan
Total : **78** octets						

ATTENTION :
- Le premier octet est la quantité de bleu
- Le deuxième octet est la quantité de vert
- Le troisième octet est la quantité de rouge

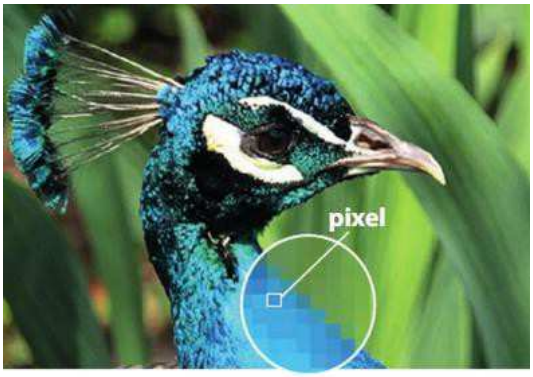
Dans le fichier, les données sont inscrites dans l'ordre BGR (Blue, Green, Red), alors que l'on parle pourtant d'image RVB (Red, Green Blue)

Coder une couleur

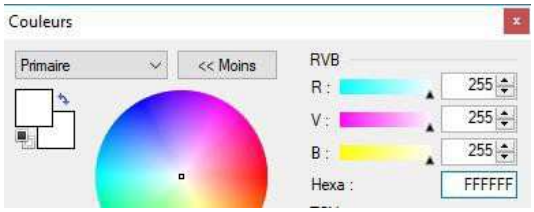
D'après les résultats précédents et l'image ci-dessous, comment semble être codée une image ?

L'image étudiée précédemment est composée d'une carte de bits. L'image est définie point par point. Les points de l'image sont appelés des pixels. Les pixels sont chacun codés sur 3 octets (soit 24 bits) , un pour le Rouge, un pour le Vert et un pour le Bleu.

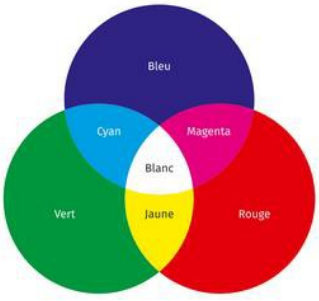
L'image est codée en commençant du bas vers le haut et de la gauche vers la droite. De nombreux octets servent à coder l'en-tête de l'image (on parle de métadonnées).



Comment sont codées chacune des couleurs pour chaque pixel ?



R De 0 à 255 (256 possibilités) **G** De 0 à 255 (256 possibilités) **B** De 0 à 255 (256 possibilités)

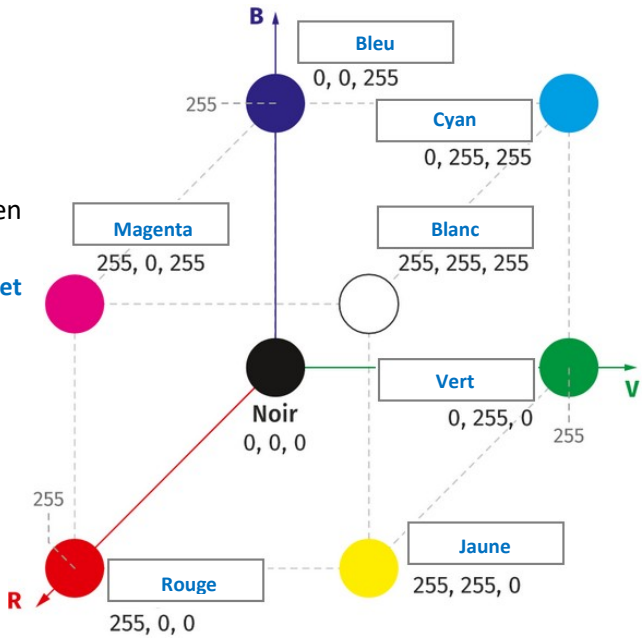


Sur quel principe scientifique se base ce codage ?

Ce codage est basé sur le principe de l'additivité des couleurs.

Avec un codage sur 24 bits ou 3 octets combien de nuances de couleurs sont possibles ?

Une image avec une profondeur de couleur de 24 bits (8 bits par couleur) permet d'obtenir 256 x 256 x256 = plus de 16 millions de nuances de couleurs.



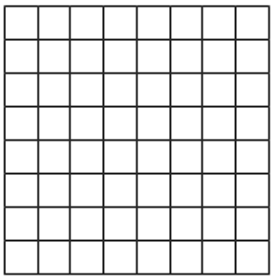
Compléter la figure ci-contre :

Dans le code d'une image, dans quel sens se fait la lecture de l'image ?

Dans le code d'une image la lecture se fait de gauche à droite mais du bas vers le haut.

On donne ci-dessous une portion du fichier correspondant à une image de 8 x 8 pixels. La séquence bmp étant répétitive, seul le début est indiqué :

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
30							FF	FF	FF	00	00	00	FF	FF	FF	00
40	00	00	FF	FF	FF	00	00	00	FF	FF	FF	00	00	00	00	00
50	00	FF	FF	FF	00	00	00	FF	FF	FF	00	00	00	FF	FF	FF
60	00	00	00	FF	FF	FF	FF	FF	FF	00	00	00	FF	FF	FF	00



Colorier ci-contre l'image correspondante. Décrire le résultat obtenu : **Il s'agit d'un damier.**

Un autre code pour créer des images…

Ouvrir les fichiers *Plan-Lyon-Metro-Tramway.bmp* et *Plan-Lyon-Metro-Tramway.svg* avec le navigateur Firefox.

Zoomer à 300% (CTRL + Molette souris).

Remarques **: L'image au format *.svg reste toujours nette quel que soit le zoom ce qui n'est pas le cas de l'image *.bmp.**

Dans Firefox, faire un clic droit sur chaque image. Les options disponibles sont-elles les mêmes ? Cliquer sur afficher le code source pour l'image .svg.

Remarques : **Les options disponibles ne sont pas les mêmes pour une image et pour l'autre. Le format *.svg permet de récupérer le code source de l'image.**

Avantage du format svg ? **Le *.svg c'est tout simplement du code xml qui va remplacer une image matricielle. Ce format permet de redimensionner facilement une image sans perte de résolution**.

Copier le code dans le BlocNote et enregistrer sous le nom cercle.svg. Ouvrir le fichier avec le navigateur Firefox :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<svg width="800px" height="800px" xmlns="http://www.w3.org/2000/svg">
<circle cx="400px" cy="400px" r="100px" fill="black" />
<title> Disque noir en SVG </title>
<desc> <Creator>Prenom NOM</Creator> </desc></svg >
```