Chapitre 1

Codage informatique

Sommaire

Tour de magie !

1. Choisis un nombre entre 0 et 63 compris ! Ne le dis à personne !
2. Dans quelle table apparaît le nombre que tu as choisis ?

CARTE 0		
1	3	5
7	9	11
13	15	17
19	21	23
25	27	29
31	33	35
37	39	41
43	45	47
49	51	53
55	57	59
61	63	

CARTE 1		
2	3	6
7	10	11
14	15	18
19	22	23
26	27	30
31	34	35
38	39	42
43	46	47
50	51	54
55	58	59
62	63	

CARTE 2		
4	5	6
7	12	13
14	15	20
21	22	23
28	29	30
31	36	37
38	39	44
45	46	47
52	53	54
55	60	61
62	63	

CARTE 3		
8	9	10
11	12	13
14	15	24
25	26	27
28	29	30
31	40	41
42	43	44
45	46	47
56	57	58
59	60	61
62	63	

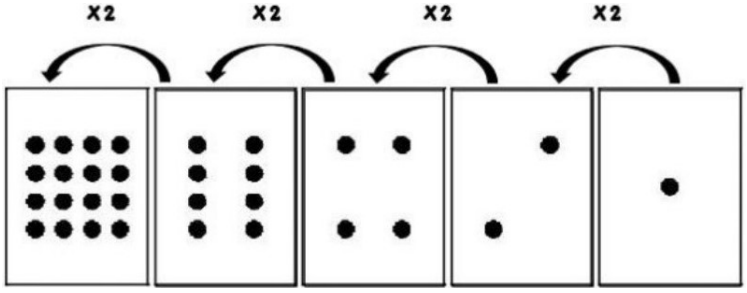
CARTE 4		
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	

CARTE 5		
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	

3. Je connais maintenant le nombre que tu as choisi !

☛ Es-tu capable de deviner le "truc" ?

Coder un entier en binaire



On dispose de 5 cartes avec respectivement les nombres 16,8,4,2 et 1.
On ne peut pas utiliser une carte deux fois.
Par exemple, il est possible de construire le nombre 20 en prenant 1 carte 16, 0 carte 8, 1 carte 4, 0 carte 2 et 0 carte 1. On note ce nombre 10100 en binaire.

1. Comment trouver 3,12 et 19 ? Existe-t-il plusieurs moyens d'obtenir ces nombres ?
2. Quel est le plus grand nombre que l'on peut obtenir ? Le plus petit ?
3. Y a-t-il un nombre que l'on ne puisse pas obtenir entre le plus grand et le plus petit ?
4. Inversement, trouve combien fait 10111 et 11010.

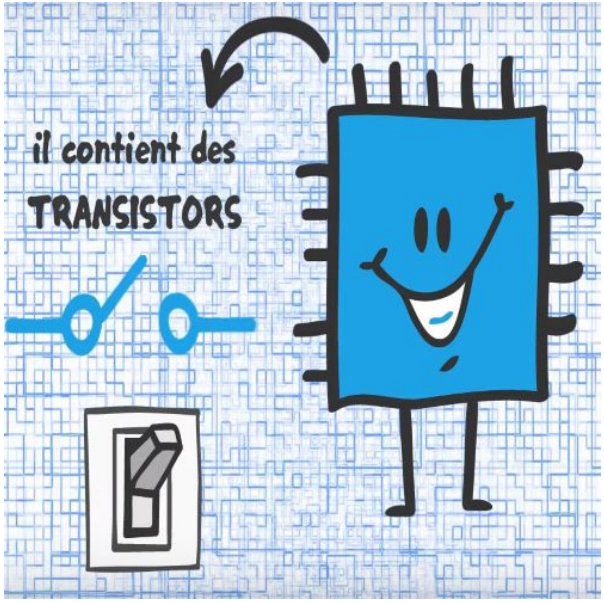
Compter en binaire de 0 à 31

Compléter le tableau ci-dessous :

16	8	4	2	1	Décimal	16	8	4	2	1	Décimal
				0	0						16
				1	1						
			1	0	2						
						1	1	1	1	1	

Découvrir le binaire

Vidéo : Le binaire

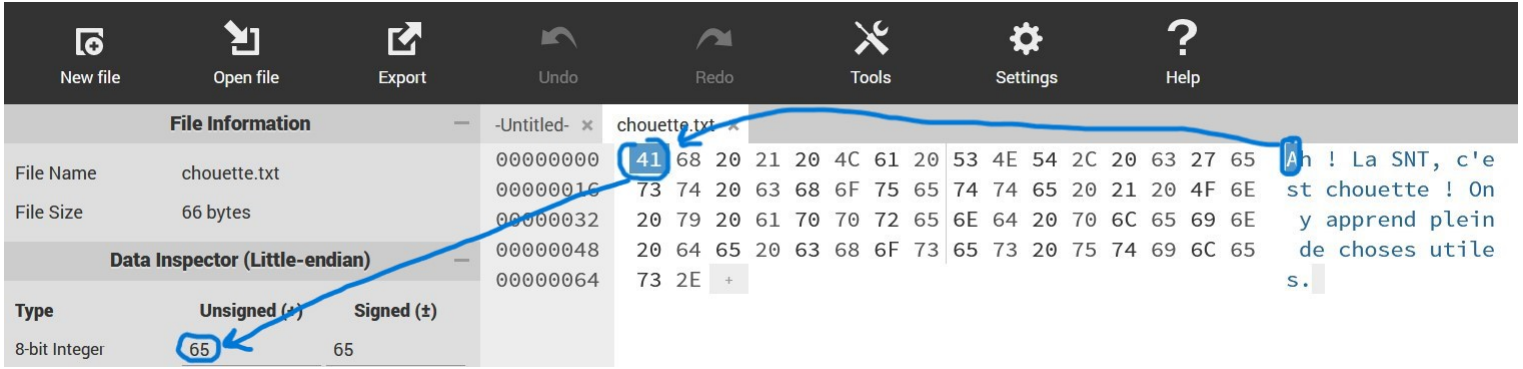


https://www.youtube.com/watch?v=VRdp_vaNRoY&t=76s

1.2 Représenter une lettre

Représentation d'un caractère

Vous trouverez sur le site un fichier nommé `chouette.txt`.
Ouvrez le à partir de `https://hexed.it/` ;
vous verrez ceci :



On s'aperçoit que chaque caractère est codé par un entier. Par exemple, en bleu, le caractère `A` est codé par le nombre `41` en hexadécimal qui donne le nombre décimal `65`.
☛ La base hexadécimale n'est pas au programme de SNT, mais la diapo suivante t'aidera à convertir avec Python

Changer de base avec Python

Il est très facile d'utiliser Python pour changer de base ! Ici le nombre décimal 65 a été converti en base 2, puis en remis en base 10, puis converti en base 16 (hexadécimal) puis à nouveau remis en base 10.

```
>>> bin(65) #pour passer de la base 10 à la base 2
'0b1000001' # le "0b" signifie que ce qui suit est du binaire
>>> 0b1000001 #pour passer de la base 2 à la base 10
65
>>> hex(65) #pour passer de la base 10 à la base 16
'0x41' # le "0x" signifie que ce qui suit est de l'hexadécimal
>>> 0x41
65
```

La table ASCII

En réalité, chaque caractère est codé par un entier entre 0 et 127 :

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[ENG OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x

22	16	10110	26	*[SYNCHRONOUS IDLE]*	70	46	1000110	106	**F**	118	76	1110110	166	**v**
23	17	10111	27	*[ENG OF TRANS. BLOCK]*	71	47	1000111	107	**G**	119	77	1110111	167	**w**
24	18	11000	30	*[CANCEL]*	72	48	1001000	110	**H**	120	78	1111000	170	**x**
25	19	11001	31	*[END OF MEDIUM]*	73	49	1001001	111	**I**	121	79	1111001	171	**y**
26	1A	11010	32	*[SUBSTITUTE]*	74	4A	1001010	112	**J**	122	7A	1111010	172	**z**
27	1B	11011	33	*[ESCAPE]*	75	4B	1001011	113	**K**	123	7B	1111011	173	**{**
28	1C	11100	34	*[FILE SEPARATOR]*	76	4C	1001100	114	**L**	124	7C	1111100	174	**|**
29	1D	11101	35	*[GROUP SEPARATOR]*	77	4D	1001101	115	**M**	125	7D	1111101	175	**}**
30	1E	11110	36	*[RECORD SEPARATOR]*	78	4E	1001110	116	**N**	126	7E	1111110	176	**~**
31	1F	11111	37	*[UNIT SEPARATOR]*	79	4F	1001111	117	**O**	127	7F	1111111	177	*[DEL]*
32	20	100000	40	*[SPACE]*	80	50	1010000	120	**P**					
33	21	100001	41	**!**	81	51	1010001	121	**Q**					
34	22	100010	42	**"**	82	52	1010010	122	**R**					
35	23	100011	43	**#**	83	53	1010011	123	**S**					
36	24	100100	44	**$**	84	54	1010100	124	**T**					
37	25	100101	45	**%**	85	55	1010101	125	**U**					
38	26	100110	46	**&**	86	56	1010110	126	**V**					
39	27	100111	47	**'**	87	57	1010111	127	**W**					
40	28	101000	50	**(**	88	58	1011000	130	**X**					
41	29	101001	51	**)**	89	59	1011001	131	**Y**					
42	2A	101010	52	*****	90	5A	1011010	132	**Z**					
43	2B	101011	53	**+**	91	5B	1011011	133	**[**					
44	2C	101100	54	**,**	92	5C	1011100	134	****					
45	2D	101101	55	**-**	93	5D	1011101	135	**]**					
46	2E	101110	56	**.**	94	5E	1011110	136	**^**					
47	2F	101111	57	**/**	95	5F	1011111	137	**_**					

Remarque importante : les accents, par exemple, ne sont pas dans la table ASCII...Il faudra pour cela utiliser une autre table (Vivement la spé NSI :-))

☛ Sais-tu pourquoi les accents n'étaient pas "prévus" ?

Coder un message

Utiliser la table ASCII précédente pour coder en binaire (chaque lettre sur 7 bits) `J'aime la SNT!` :

	J	'	a	i	m	e	
Décimal	74						
Binaire	1001010						

	l	a		S	N	T	!
Décimal	74						
Binaire	1001010						

Décoder un message

Utiliser la table ASCII précédente pour décoder le message suivant :

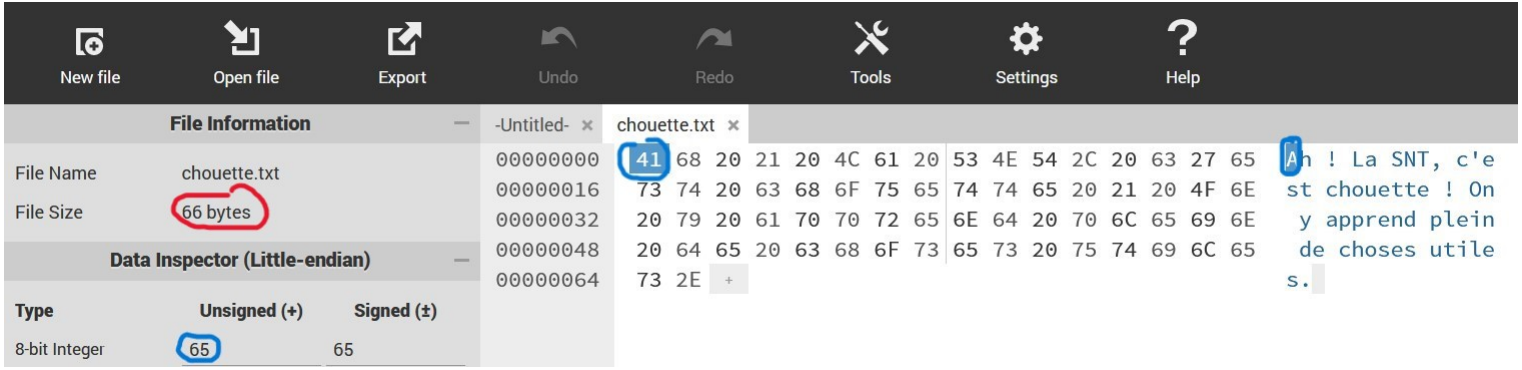
Binaire	1000010	1110010	1100001	1110110	1101111	0100001
Décimal	66					
Décodage						

Pour convertir le binaire en décimal, on pourra utiliser ce tableau :

64	32	16	8	4	2	1	Décimal
1	0	0	0	0	1	0	$64 + 2 = 66$

Stockage des données

Reprenons le screen précédent :



$$1 \text{ byte} = 1\text{B} = 1 \text{ octet} = 8 \text{ bits}$$

L'**octet** est utilisé pour mesurer les espaces de stockages. Comme l'octet est une unité relativement petite, on utilise souvent des sous-multiples :

1 Ko = 1 000 o
1 Mo = 1 000 Ko
1 Go = 1 000 Mo
1 To = 1 000 Go

Tableau de conversion

To				Go			Mo			ko			Octets		
					1	4 ,	6	0	0						
3	2	0	0												

14 600 ko = 14,6 Mo

3,2 To = 3 200 Go

Exercice

Une disquette 3 Pouce ½ à une taille de stockage de 1.44 Mo.

1. Convertir la taille de stockage de la disquette en Ko.
2. Combien de disquettes peut contenir un CD-ROM dont la taille est 650 Mo ?
3. Combien de disquettes peut contenir un disque dur de 80 Go ?
4. En supposant qu'un fichier mp3 prend en moyenne 6 Mo de stockage, combien peut-on mettre de fichiers mp3 sur un cd de 80 Go ?

Quelques ordres de grandeur..

Un CD
700 Mo = 0.7Go

Un DVD
4.7 Go

Un Blu-Ray
25 Go

Une musique
4 Mo

Une photo
6 Mo

Un document
50 Ko

Un film
700 Mo

Un ordinateur récent
de 500 Go à 4 To

Une clé USB / carte mémoire
de 8 Go à 200 Go

Une disquette
1.4 Mo

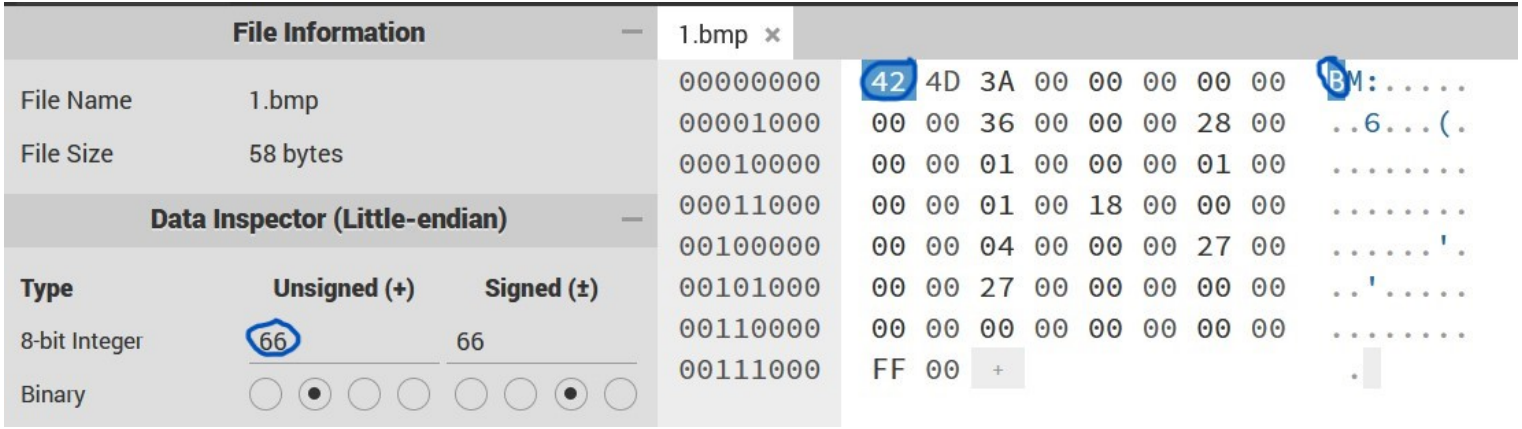
loi de Moore

L'augmentation de la capacité tout en diminuant la dimension des supports est due à la miniaturisation des composants électroniques et l'augmentation de leur capacité de traitement qui a été exponentielle.

La loi de **Moore** doit son nom à Gordon Earle Moore, informaticien qui a été cofondateur d'Intel.
Gordon E. Moore postule que la complexité des semi-conducteurs va doubler tous les ans à coût constant. En 1975, il réajuste sa prédiction. Il va dire cette fois que le nombre de transistors sur une puce de microprocesseur double tous les deux ans. Cette prédiction s'est avérée vraie.
Moore avait prédit que cette loi serait valable jusqu'en 2015, parce qu'à partir de ce moment la taille des transistors serait de l'ordre du nanomètre. Et qu'est ce qui fait environ un nanomètre ? L'atome. On ne peut pas faire de transistor plus petit qu'un ou quelques atomes.

Analyse d'un fichier image

Ouvrez le fichier `1.bmp` avec `Hexed`. Vous devriez voir à peu près cela :



⚠ Un octet est un ensemble de 2 lettres (notation hexadécimale) comme entourée en bleu sur le screen. Ici l'octet noté `42` vaut `66` en valeur décimale (cf screen).

☛ Il est aussi possible d'utiliser la console python pour faire rapidement la conversion. Il suffit de faire précéder le code hexadécimal de `0x` :

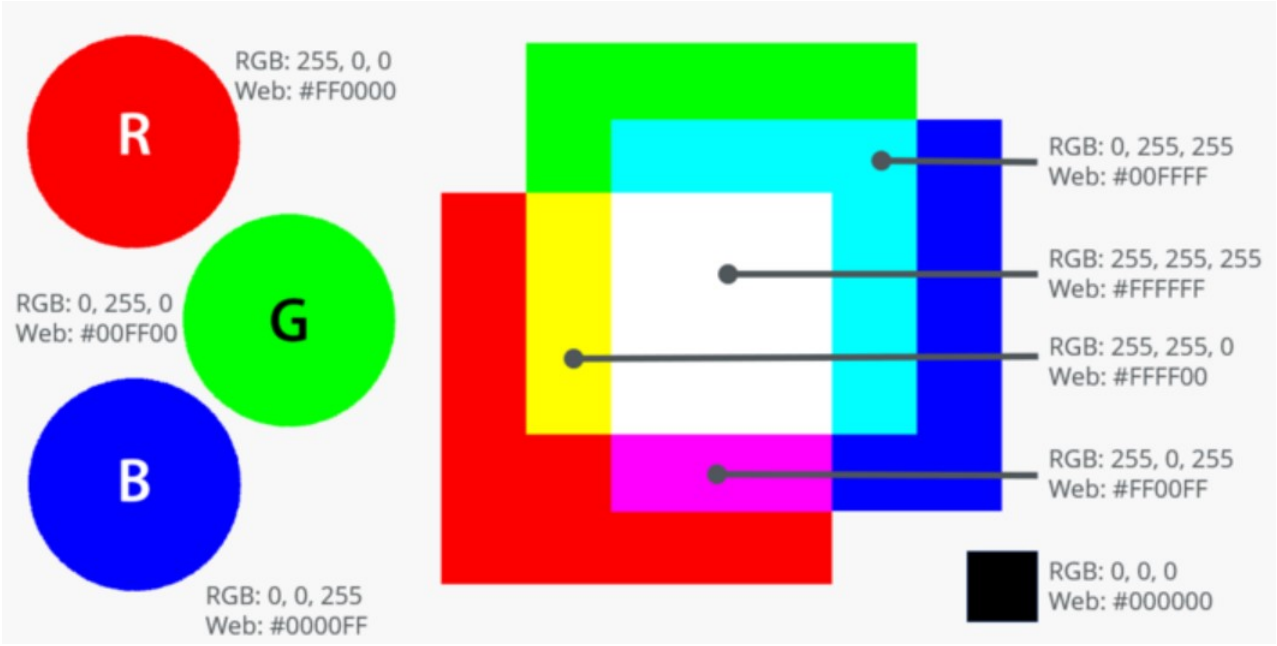
```
>>> 0x42
66
```

Analyse de l'en tête du fichier image

- Les deux premiers octets sont .... et .... : cela correspond aux lettres .... et ...., ce qui signifie que le fichier est un fichier .....
- Les quatre octets suivants sont ....,....,.... et .... : cela correspond à la taille du fichier en octets : Le fichier fait .....octets.
- Les octets 19 à 22 donnent la largeur de l'image : .....
- Les octets 23 à 26 donnent la hauteur de l'image : .....
- Les octets 29 à 32 le nombre de bits pour coder chaque pixel : on trouve ici 24 bits (3 octets), c'est à dire 8 bits (1 octets) pour chacune des couleurs Bleu, vert et Rouge (ds cet ordre !).

Système RGB

Chaque pixel est colorié en fonction d'un système RGB, sur 3 octets, à partir de 3 couleurs principales : le Vert, le Bleu et le Rouge.
Voici quelques couleurs :



☛ Avec ce système, on peut définir 256^3 couleurs différentes !

Analyse de la couleur du pixel contenu dans le fichier image

- A partir de l'octet 55, on voit les informations de l'image en elle-même !
- On commence par le pixel en bas à gauche. Chaque pixel est codé sur 3 octets (pour les 3 couleurs Bleu, Vert et Rouge). On lit tous les pixels de la ligne, puis on remonte d'un cran, toujours en lisant de gauche à droite...
- Les octets 55,56 et 57 sont ...., ... et .... , ce qui correspond à un pixel de couleur ....

Savoir lire un fichier `.bmp` et découvrir les 8 couleurs cachées !

On donne le fichier suivant :

```
00000000  42 4D 4E 00 00 00 00 00  BMN.....
00001000  00 00 36 00 00 00 28 00  ..6...(.
00010000  00 00 08 00 00 00 01 00  ........
00011000  00 00 01 00 18 00 00 00  ........
00100000  00 00 18 00 00 00 27 00  ......'.
00101000  00 00 27 00 00 00 00 00  ..'.....
00110000  00 00 00 00 00 00 00 00  ........
00111000  FF 00 FF 00 FF 00 00 00  . . ...
01000000  00 00 FF FF FF FF FF 00  ..      .
01001000  FF 00 FF 00 FF FF +      . .
```

1. Déterminer la nature du fichier
2. Déterminer la taille du fichier
3. Déterminer La largeur et la hauteur du fichier
4. Déterminer les différentes couleurs (dans l'ordre, de gauche à droite)

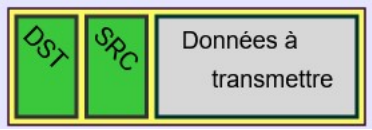
☛ Vous trouverez sur le site un programme python qui permet de découvrir la couleur en fonction des paramètres r (Rouge),g (Vert) et b (Bleu). Par exemple,

```
>>> pixel(255,0,0)
```

va donner un carré de couleur rouge !

La communication par paquets

On nomme paquet une information à transmettre de réseau en réseau dont l'en-tête (ici en vert) est suffisant pour gérer la transmission : pas besoin de lire le contenu du message (ici en gris) pour savoir où il faut l'amener.
De façon basique, il suffit donc que les appareils des différents réseaux partagent une norme commune sur l'en-tête du paquet pour savoir comment le lire et le gérer.



☛ On parle donc de commutation de paquet car c'est le contenu de l'en-tête du paquet qui va servir à choisir le chemin à prendre. En aucun cas les données du message (en gris) ne vont être lues ou modifiés.

Le protocole TCP/IP

Protocole TCPI/IP

https://www.youtube.com/watch?v=aX3z3JoVEdE&t=4s

- Le **protocole IP** est l'ensemble des codes et techniques de communication permettant aux paquets de partir d'un machine émettrice pour atteindre la machine réceptrice en passant de routeur en routeur.
- Le protocole TCP est l'ensemble des codes et techniques de communication permettant :
  - A la machine émettrice de savoir que le message est bien arrivé
  - A la machine réceptrice de savoir si le message reçu est exactement le message émis, sans erreur
  - A la machine émettrice d'émettre à nouveau un message qui n'est pas arrivé correctement du coup (disparition ou modification partielle)

A quoi sert Internet ?

Internet ne sert à rien en lui-même !

☛ Il s'agit de l'infrastructure matérielle et logicielle qui permet d'établir des communications. Par contre, il existe une multitude d'applications qui utilisent Internet pour fonctionner. On trouve ainsi :

-
-
-
-
-
-
- Et bien d'autres utilisations (disparues ou futures :-))