Systèmes UNIX. TD 2 : nombres magiques, images aux formats PBM, PGM, PPM

1 Nombres magiques

Par convention, les fichiers du même format souvent commencent par les mêmes octets. Ces préfixes, déterminant le format, sont appelés des *nombres magiques*[*]. Cela est très pratique : des nombres magiques nous permettent de déterminer rapidement le type d'un fichier, même en cas d'extension manquante ou incorrecte. La commande `file`, permettant de déterminer le type d'un fichier, utilise cette technique. Par exemple, les fichiers de type `.zip` commencent toujours par deux octets "`50 4B`" correspondants aux caractères "`PK`" dans le code ASCII.

Code ASCII :

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

EXERCICE 1.1. Télécharger le fichier `nat2022_csv.zip` contenant des données sur les prénoms attribués aux enfants nés en France depuis 1900 :
https://www.data.gouv.fr/fr/datasets/fichier-des-prenoms-depuis-1900/

EXERCICE 1.2. Ouvrir le fichier dans un éditeur de texte et regardez son début.

EXERCICE 1.3. Tester le type du fichier avec la commande `file`.

EXERCICE 1.4. Renommer le fichier `nat2022_csv.zip` en tiktok.gif. Tester a nouveau le type du fichier avec la commande `file`.

ASTUCE : On peut utiliser la commande `mv` pour renommer un fichier.

*. https://fr.wikipedia.org/wiki/Nombre_magique_(programmation)

2 Portable bitmap file format (PBM)

Le format PBM a été défini dans les années 1980 par Jef Poskanzer. L'objectif initial était de donner la possibilité d'encoder les images monochromes afin de les transmettre dans un message email. À l'époque, il n'existait aucun moyen fiable d'envoyer des images par email. Uniquement le texte ASCII a été bien pris en charge par les gestionnaires d'email. Des tentatives d'envoyer des données binaires (sous un autre format que ASCII) ont souvent créé des problèmes de corruption de données. De plus, différents systèmes utilisaient des codes différents pour les caractères de la fin de ligne. Malheuresement même aujourd'hui c'est souvent le cas. Dans ce contexte, il fallait supporter les changements de formatage et d'encodage du texte par des logiciels email, c'est-à-dire être assez tolérant à la présence ou l'absence des espaces et des caractères de la fin de ligne.

Un fichier PBM classique commence par le nombre magique `'P1'` (par deux caractères `'P'` et `'1'`) suivi par un caractère de séparation (un espace, tabulation, passage à la ligne, retour chariot) ; puis on noté la largeur de l'image en pixels (encodée en texte, en décimal) ; puis on met un caractère de séparation ; la hauteur de l'image ; et encore un caractère de séparation. En suite, on encode les pixels de gauche à droite et de haut en bas en utilisant les deux caractères suivants :

- ▷ caractère `1` correspond à un pixel noir ;
- ▷ caractère `0` correspond à un pixel blanc.

Les caractères de séparation à l'intérieur sont ignorés. Les lignes commençant par # sont ignorées. Aucune ligne ne peut pas dépasser 70 caractères. Voici, un exemple de fichier `PBM` classique :

```
P1
# Un exemple de la lettre "O"
7 10
0000000
0111110
0100010
0100010
0100010
0100010
0100010
01000100111110
0 0 0 0000
```

De plus, il existe des formats PGM (Portable GrayMap) et PPM (Portable PixMap) permettant l'encodage des images en niveau de gris et, respectivement, en couleur.

Voici la documentation de ces formats :

- ▷ http://www.delafond.org/traducmanfr/man/man5/pbm.5.html
- ▷ https://fr.wikipedia.org/wiki/Portable_pixmap
- ▷ https://en.wikipedia.org/wiki/Netpbm
- ▷ https://linux.die.net/man/5/pbm

EXERCICE 2.1. En utilisant un éditeur de texte (nano par exemple) créer une image de type PBM de taille 14x10px contenant le texte "OM".

EXERCICE 2.2. Convertire l'image PBM vers une image PNG :
"`convert image.pbm image.png`"

EXERCICE 2.3. Visualiser les images PNG et PBM. On peut faire cela avec logiciel "gimp". Ce logiciel est disponible pour Windows, GNU/Linux et MacOS : https://www.gimp.org/.

EXERCICE 2.4. Créer une version de l'image de l'exercice 1 en niveau de gris, avec le format PGM.

EXERCICE 2.5. Créer une version colorée de l'image de l'exercice 1, avec le format PPM.

ASTUCE : https://linux.die.net/man/5/pgm, https://linux.die.net/man/5/ppm