Des bugs[1] riches d'enseignement

Sommaire

1 Un bug ou bogue est un défaut de conception d'un programme informatique à l'origine d'un dysfonctionnement (source Wikipédia).

1 Donkey Kong

Donkey Kong, jeu vidéo d'Arcade créé par Nintendo en 1981 est l'un des premiers jeux de plates-formes. Donkey Kong introduit le personnage de Mario qui deviendra une véritable icône du genre.
Ce jeu calcule pour chaque niveau la durée dont le joueur dispose pour finir le niveau en question.

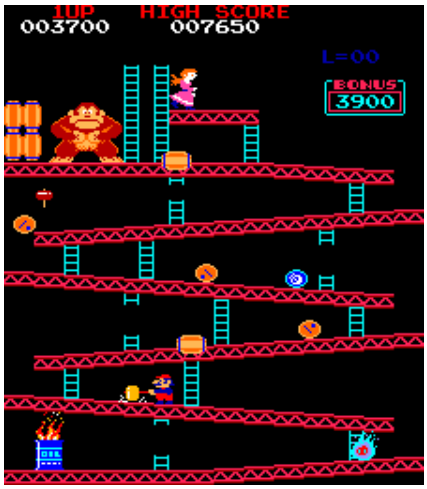

Figure 1 : Donkey Kong

Le calcul est simple et repose sur la formule suivante :

```
duree_niveau [s] = niveau x 10 + 40
```

La durée du niveau est stockée dans un octet et est codé en binaire naturel.
Lors de sa sortie aucun joueur n'a pu dépasser le niveau 22 car l'écran de fin apparaissait trop tôt.

Q1. Quelle est la durée du niveau 22 du « point de vue du jeu » ?

duree_niveau = **22** x 10 + 40 = 260 donc la durée du niveau 22 est égale à 260 s = 4 min 20 s

Un nombre entier codé sur n bits en binaire naturel ne peut représenter que les valeurs entières allant de 0 à $2^n - 1$.

Q2. En déduire pourquoi aucun joueur n'a pu dépasser le niveau 22 lors de la sortie de Donkey Kong.

260 ne peut être codé correctement sur un octet. On a :
$260)_{10} = 1\ 0000\ 0100)_2$ ce qui donne sur un octet $0000\ 0100)_2 = 4)_{10}$
Le joueur une fois arrivé au niveau 22 n'a que 4 secondes pour effectuer le niveau, ce qui est impossible !

Q3. Proposer une manière de résoudre ce bug.

Il faut, par exemple, coder le niveau sur 16 bits ou bien faire en sorte qu'une fois arrivé au niveau 22 la durée du jeu « tienne » sur un octet c'est-à-dire qu'elle soit égale à 255 s = 4 min et 15 s.

Pour les curieux voici un lien vers un dépôt github qui propose un correctif :

https://gist.github.com/guebe/e074b165ab6fd291b1fa3f424c18a893

2 Ariane 5

Le vol inaugural 501 a eu lieu le 4 juin 1996 à Kourou (Guyanne).

Le lanceur fut détruit au bout de 37 secondes de vol. L'échec était dû à une erreur informatique. Le coût estimé des pertes s'élève à 500 millions de dollars.

L'enquête a révélé que lors du vol une exception s'est produite. Cette exception est due à la conversion d'un nombre décimal codé sur 64 bits et représentant l'accélération horizontale du lanceur en un nombre entier signé codé sur 16 bits.

Figure 2 : Ariane 5

Cette exception a provoqué la panne du système de navigation puis celui du système de secours. Le pilote automatique a alors ordonné une violente correction de trajectoire. La fusée a alors dévié brutalement de sa trajectoire ce qui a déclenché une autodestruction préventive de la fusée.

Les nombres entiers signés sont codés :

- en binaire naturel pour les nombres positifs ;
- en complément à 2 pour les nombres négatifs.

Cela signifie par exemple qu'un octet représente en binaire signé les nombres compris entre -128 et +127.

Entiers naturels	Binaire				Entiers relatifs
16	1	0	0	0	0
15		1	1	1	1	−1
14		1	1	1	0	−2
13		1	1	0	1	−3
12		1	1	0	0	−4
11		1	0	1	1	−5
10		1	0	1	0	−6
9		1	0	0	1	−7
8		1	0	0	0	−8
7		0	1	1	1	7
6		0	1	1	0	6
5		0	1	0	1	5
4		0	1	0	0	4
3		0	0	1	1	3
2		0	0	1	0	2
1		0	0	0	1	1
0		0	0	0	0	0

Figure 3 : binaire signé sur 4 bits

Q1. Que vaut le MSB en binaire signé d'un entier positif codé sur 8 bits. Même question pour un entier négatif codé sur 8 bits.

On remarque d'après le tableau ci-contre qu'en binaire signé :

MSB = 0 ⇒ l'entier est positif

MSB = 1 ⇒ l'entier est négatif

Q2. Vérifier en binaire signé que l'addition de 5 et de -5 donne bien zéro sur 4 bits.

$5)_{10} = 0101)_{2\ signé}$

$-5)_{10} = 1011)_{2\ signé}$

Donc : 0101

\+ 1011

$10000)_{2\ signé} = 0000)_{2\ signé\ sur\ 4\ bits}$

Q3. Quelle valeur positive a dépassé le nombre décimal codé sur 64 bits pour que cela déclenche une exception lors de sa conversion ?

Sur 4 bits en binaire signé on observe que l'on passe « brutalement » de 7 à -8 (2^4) alors que l'on a seulement augmenter de 1 le nombre binaire.

Sur 16 bits cela a lieu pour la valeur 2^{15} -1 = 32767.

Donc lorsque l'accélération est passée de 32767 à 32768 sur 64 bits cela n'a posé aucun problème. Par contre la valeur convertie sur 16 bits, quant à elle, est passée brutalement de 32767 à -32768.

Pour voir un programme illustrant ce bug :
https://replit.com/@franckrivier1/ariannebug?v=1

3 Affichage bizarre de page html (« arrête ton charset »)

Avez-vous jamais remarqué des pages html qui affichent les accents de manière bizarre ?

Figure 4 : mauvais rendu d'une page html

Cela est dû à un mauvais décodage du texte par le navigateur.
Dans l'exemple précédent la page est encodée en UTF-8 or le navigateur pense traiter de l'ISO 8859-1 parce que le jeux de caractères (charset en anglais) est mal spécifié dans la page html.

Voici un extrait de chaque table de caractère (source Wikipédia):

ISO/CEI 8859-1															
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	*positions inutilisées*															
1x																
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8x	*positions inutilisées*															
9x																
Ax	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
Bx	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figure 5 : table ISO 8859-1

Type	Caractère	Point de code (hexadécimal)	Valeur scalaire		Codage UTF-8	
			décimal	binaire	binaire	hexadécimal
Contrôle (C0)	[NUL]	U+0000	0	0	00000000	00
	[US]	U+001F	31	1·1111	00011111	1F
Texte (US-ASCII)	[SP]	U+0020	32	10·0000	00100000	20
	0	U+0030	48	11·0000	00110000	30
	9	U+0039	57	11·1001	00111001	39
	?	U+003F	63	11·1111	00111111	3F
	@	U+0040	64	100·0000	01000000	40
	A	U+0041	65	100·0001	01000001	41
	Z	U+005A	90	101·1010	01011010	5A
	a	U+0061	97	110·0001	01100001	61
	z	U+007A	122	111·1010	01111010	7A
	~	U+007E	126	111·1110	01111110	7E
Contrôle (C0 et C1)	[DEL]	U+007F	127	111·1111	01111111	7F
	[PAD]	U+0080	128	1000·0000	11000010 10000000	C2 80
	[APC]	U+009F	159	1001·1111	11000010 10011111	C2 9F
Texte (PMB)	[NBSP]	U+00A0	160	1010·0000	11000010 10100000	C2 A0
	¿	U+00BF	191	1011·1111	11000010 10111111	C2 BF
	À	U+00C0	192	1100·0000	11000011 10000000	C3 80
	é	U+00E9	233	1110·1001	11000011 10101001	C3 A9
	ƀ	U+07FF	2047	111 1111·1111	11011111 10111111	DF BF
	ᚠ	U+0800	2048	1000 0000·0000	11100000 10100000 10000000	E0 A0 80
	€	U+20AC	8 364	10·0000 1010·1100	11100010 10000010 10101100	E2 82 AC
	퟿	U+D7FF	55 295	1101·0111 1111·1111	11101101 10011111 10111111	ED 9F BF

Figure 6 : table UTF-8

Q1. À partir des tables fournies expliquer l'affichage erroné du mot « Actualités » (indice il faut partir de la table UTF-8 dans un 1er temps) :

On sait que le mot actualité est encodé en UTF-8, c'est le décodage qui pose problème.

Le mot apparaît sous cette forme : ActualitÃ© or dans la table UTF-8 on relève :

'é' = C3 A9)$_{UTF-8}$ et on a bien C3)$_{ISO-8859-1}$ = 'Ã' et A9)$_{ISO-8859-1}$ = '©'.

Le texte est donc bien encodé en UTF-8 mais décodé en ISO-8859-1.

Q2. Donnez la valeur hexadécimale des caractères ' è ' (e accent grave) et ' ï ' (i tréma) codés en UTF-8 (indice il faut partir de la table ISO 8859-1 cette fois) :

Dans l'article on relève que :

- le 'è' apparaît sous la forme 'Ã¨' soit C3A8)$_{ISO-8859-1}$. Le code UTF-8 de 'è' est donc C3A8)$_{UTF-8}$;
- le 'ï' apparaît sous la forme 'Ã¯' soit C3AF)$_{ISO-8859-1}$. Le code UTF-8 de 'ï' est donc C3AF)$_{UTF-8}$;

De nos jours 95% des sites web sont encodés en UTF-8 et les erreurs d'affichage des pages html sont de plus en plus rares.

Pour convertir des caractères en UTF-8 :

https://www.cogsci.ed.ac.uk/~richard/utf-8.cgi?input=C3+A8&mode=bytes