QCM2 - Mesure de l'information, codage du son, des images et des vidéos

Q1 – Indiquez les assertions correctes :

- ❑ **1 Kilo octet = 1000 octets**
- ❑ ~~1 Kibi octet = 1000 octets~~
- ❑ **1 Kibi octet = 1024 octets**
- ❑ ~~1 Kibi octet = 1024 bits~~
- ❑ ~~1 Ko = 1024 octets~~
- ❑ ~~1 Ko = 1000 bits~~
- ❑ ~~1 Kibi Octet = 1000 octet~~
- ❑ **symbole du kibi est « ki »**
- ❑ **symbole du Gibi est « Gi »**
- ❑ **symbole du kilo est « k »**
- ❑ **ko veut dire « kilo octet »**
- ❑ ~~ko veut dire « **kilo bit** »~~
- ❑ **kilo = 10^3 = 1000**
- ❑ **méga = 10^6 et géga = 10^9**
- ❑ **Téra = 10^{12}**
- ❑ ~~kibi = kilo~~
- ❑ **kibi = 2^{10} et gibi = 2^{20}**
- ❑ **Tibi = 2^{30},**

Q2 - Calculez en bits les valeurs suivantes

- ❑ 15 octets
  = 15x8 = 120 bits
- ❑ 3,2 Méga octets (ou 3,2 Mo)
  = 3,2*10^6x8 bits = 25,6 x 10^6 bits
- ❑ 30 Kibi octets (ou 30 KiO)
  = 30x2^{10} octets = 240x2^{10} bits
- ❑ 1 Tera octets (ou 1 To)
  = 1 x 10^{12} octets = 8 x 10^{12} bits
- ❑ 1 Gibi octets (ou 1 GiO)
  = 1x2^{30} octets = 8x2^{30} bits

Q3 - Le son est par définition

- ❑ **un signal analogique**
- ❑ ~~un signal numérique~~
- ❑ **est une vibration mécanique d'un fluide (de l'air notamment), qui se propage sous forme d'*ondes***

Q4 – Indiquez les bonnes réponses :

- ❑ ~~le format « **wav** » est un format numérique du son compressé~~
- ❑ **le format « mp3 » est un format numérique du son compressé**
- ❑ **le format « ogg » est un format numérique du son compressé**
- ❑ **le format « mp4 » est un format numérique de la vidéo**

Q5 – On a crée des codages du son, de l'image et de la vidéo **compressés** au lieu de les garder bruts (sans compression) ?

- ❑ ~~Juste pour s'amuser~~
- ❑ **pour réduire la taille des fichiers**
- ❑ ~~pour rendre rapide les programmes~~

Q6 – Pour calculer la définition d'une image, on utiliser deux valeurs :

- ☐ ~~la surface (en pixels)~~
- ☐ **la largeur (en pixels)**
- ☐ **la longueur (en pixel)**
- ☐ ~~la profondeur (24 bits)~~

Définie les couleurs

Q7 – La profondeur d'une image définie l'ensemble de ses couleurs. On utilise dans le codage RVB en général 24 bits:

- ☐ **Vrai**
- ☐ ~~Faux~~

Q8 – Lorsqu'on vous dit qu'une image est codée en RVB que veut dire ces lettres :

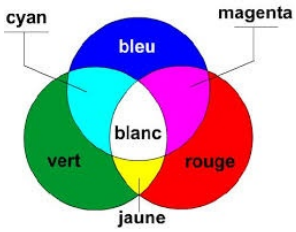
- ☐ **R : Rouge**
- ☐ **V : Vert**
- ☐ **B : Bleu**

Q9 – Sur combien de bits code-t-on ma couleur en « ***True color*** » ? Indiquez le nombre de bits par couleur

- ☐ ~~9 bits : 3 / couleur (vert, rouge et bleu)~~
- ☐ ~~15 bits : 5 / couleur (vert, rouge et bleu)~~
- ☐ **24 bits : 8 / couleur (vert, rouge et bleu)**

Q10 : En supposant que vous codez en ***True color*** (***RVB***), indiquez les couleurs représentées par les codes suivants :

- ☐ $(255, 255, 0)_{10}$ ↔**(R=255, V=255, B=0)$_{10}$**
  Intensité maximales pour le rouge et le vert ce qui donne le Jaune
- ☐ $(FF00FF)_{16}$ ↔**(R=255, V=0, B=255)$_{10}$**
  Intensité maximales pour le rouge et le bleu ce qui donne le magenta
- ☐ $(255, 255, 255)_{10}$↔**(R=255, V=255, B=255)$_{10}$**
  Intensité maximales pour les 3 couleurs de basece qui donne le blanc
- ☐ $(5, 255, 5)_{10}$↔**(R=5, V=255, B=5)$_{10}$**
  Intensité maximales pour le Vert et très faible intensité pour les autres couleurs, ce qui donne le Vert
- ☐ $(10, 0, 255)_{10}$↔**(R=10, V=0, B=255)$_{10}$**
  Intensité maximales pour le Bleu et très faible intersité des 2 autres couleurs ce qui donne le Bleu
- ☐ $(0, 255, 255)_{10}$↔**(R=0, V=255, B=255)$_{10}$**
  Intensité maximales pour le vert et le bleu ce qui donne le cyan
- ☐ $(0, 0, 0)_{10}$↔**(R=0, V=0, B=0)$_{10}$**
  aucune Intensité des 3 couleurs de base ce qui donne le noir
- ☐ $(200, 200, 200)_{10}$↔**(R=200, V=200, B=200)$_{10}$**
  forte Intensité équilibré des 3 couleurs ce qui donne le gris clair car proche du blanc
- ☐ $(80, 80, 80)_{10}$↔**(R=80, V=80, B=80)$_{10}$**
  Intensité équilibrée et relativement faible des 3 couleurs de base ce qui donne le gris foncé car proche du noir



Q11 – En supposant que vous avez une définition d'une image 400x300, calculez le poids (capacité mémoire) de cette image si sa profondeur est codée :

- ❑ en « ***True type*** » ***RVB*** (24 bits) :
  400x300x24 bits = 2 880 000 bits
- ❑ sur 8 bits (256 couleurs) :
  400x300x8 bits = 960 000 bits
- ❑ en noir et blanc : **400x300x1 bits = 120 000 bits**

Q12 – On considère qu'on a une animation lorsqu'on fait défiler combien d'image par secondes ?

❑3 ●10 ❑25 ❑30 ❑100

C'est à partir de 10 images par seconde que le cerveau humain n'arrive plus à distinguer les images. Ainsi, il va avoir l'impression de voir le mouvement (animation) dans les images.

Q13 – Lorsqu'on code une image, il est souhaitable de faire défiler le maximum d'images par secondes au moins 50 images par seconde pour avoir une très bonne qualité de l'animation ?

- ❑ Vrai
- ❑ Faux

Au delà de 25 images par seconde, notre cerveau ne percevra la différence , ainsi, si vous monter des vidéos avec 100 images par secondes ou 25 images par secondes il est fort possible que vous ne verrez pas la différence !

Q1 – Codage des entiers :

Si le codage est en binaire naturel (dit aussi binaire pure ou entier non signé), indiquez l'intervalle des valeurs pouvant être représentées sur n bits.

Donnez L'étendue des valeurs du codage S+VA, C1 et C2:

Codages	Indiquez l'étendue des valeurs si le codage est sur 3 bits (bit de signe compris)
S+VA	$[-(2^{3-1}-1), +(2^{3-1}-1)] = [-3, +3]$
C1	$[-(2^{3-1}-1), +(2^{3-1}-1)] = [-3, +3]$
C2	$[-2^{3-1}, +(2^{3-1}-1)] = [-4, +3]$

Codages	Étendue des valeurs si le codage est sur n bits (bit de signe compris)
S+VA	$[-(2^{n-1}-1), +(2^{n-1}-1)]$
C1	$[-(2^{n-1}-1), +(2^{n-1}-1)]$
C2	$[-2^{n-1}, +(2^{n-1}-1)]$

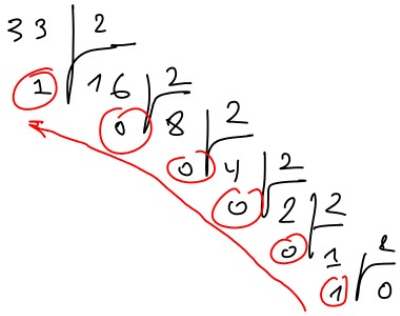
Q2 – Codage S+VA, C1 et C2 avantage et inconvénients

Complétez le tableau suivant :

	Avantages	Inconvénients
S+VA	Représentation	…… ……
	naturelle et simple	double représentation du zéro
C1	Représentation des nombres négatifs relativement simple calculs possibles	double représentation du zéro problème de performance du à l'addition de la ……
C2	…… …… …… calculs possibles	/

Q3 – Donnez sur 8 bits, en S+VA, C1 et C2 le codage des nombres suivants :

$(+33)_{10}$ = (?)$_{SVA}$ = (?)$_{C1}$ = (?)$_{C2}$



ce qui donne sur 8 bits :

$(+33)_{10}$ =(**0** 0100001)$_{SVA}$ =(**0** 0100001)$_{C1}$ = (**0** 0100001)$_{C2}$

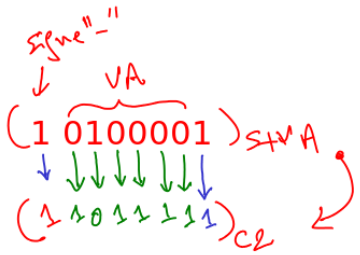
$(-33)_{10}$ = (?)$_{SVA}$ = (?)$_{C1}$ = (?)$_{C2}$

Ce qui donne : $(-33)_{10}$ = (1 0100001)$_{S+VA}$

en complément à 1 (C1) il suffit d'inverser tous les bits de la valeur absolue :

Ce qui donne : $(-33)_{10}$ = (1 1011110)$_{S+VA}$

en complément à 2 (C2) il suffit de prendre la valeur absolue, de recopier tous les 0 en commençant de la droite, arrivé au premier, il faut le recopier, puis inverser e reste. Ne pas oublier de mettre le signe à « 1 » car 'est un nombre négatif :

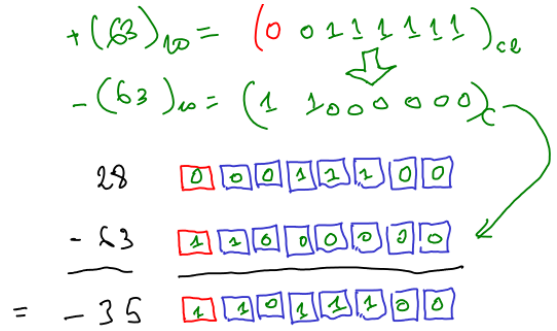


Ce qui donne : $(-33)_{10}$ = (1 1011111)$_{C2}$

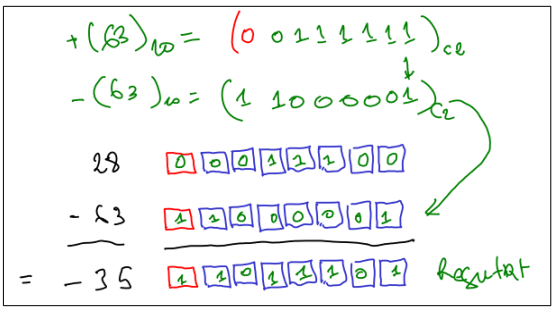
Q4 – Calcul arithmétique :

Effectuez les calculs suivants

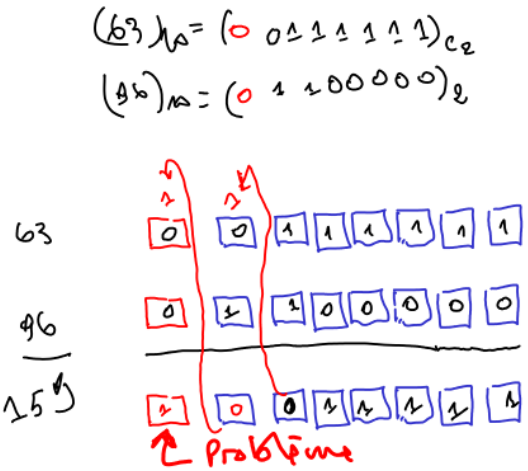
- 28+ (–63) en C1 sur 8 bits



- 28+ (–63) en C2 sur 8 bits



- 63 + 96 en C2 sur ~~7 bits~~ (ici il y a une erreur dans l'énoncé c'est **8 bits** au lieu de 7 bits)



Nous avons additionné 2 nombres positifs et nous avons obtenu un nombre négatif : Nous déduisons qu'il y a une situation de débordement. D'ailleurs, vérifions l'étendue des valeurs que l'on peut représenter sur 8 bits en C2 : $[-2^{8-1}, 2^{8-1}-1] = [-128, +127]$

En faisant le calcul de 63 et de 96 on obtient 159 qui est en dehors de l'intervalle des valeurs possibles en C2 sur 8 bits !

Q5 – Codage en virgule fixe :

Sur **9 bits** dont un bit de signe et 5 bits pour la partie entière et 3 bits pour la partie décimale, donnez les représentations des nombres suivants :

$(-12,25)_{10}$	(……..)$_{S+VA}$

Sur 9 bits dont 3 pour la partie décimale :

$(+12,25)_{10}$ = (0 01100 , 010)$_{S+VA}$

ce qui donne **$(+12,25)_{10}$ = (1 01100 , 010)$_{S+VA}$**

$(AC,8)_{16}$	(……..)$_{S+VA}$

$(AC,8)_{16}$= (1010 1100, 1000)$_2$

On voit bien que la partie entière de ce nombre nécessite au moins 9 bits (en intégrant le bits de signe). On déduit que l'on ne peut pas le représenter en S+VA sur 5 bits *comme demandé dans l'énoncé.* Nous devons donc prévoir au total **9+3 bits** (en supposant que nous prévoyant 3 bits pour la partie décimale). Voici ce que ça va donner :

$(AC,8)_{16}$= (1 10101100 , 100)$_{S+VA}$

Bit de signe | Partie entière sur 8 bits | Partie décimale sur 3 bits

$(-12,25)_{10}$	(……..)$_{C1}$

$(-12,25)_{10}$ = (?)$_{S+VA}$

Sur 9 bits dont 3 bits pour la partie décimale :

$(+12,25) = (0\ 01100{,}010)_{C1}$

ce qui donne $(-12,25) = (1\ 10011{,}101)_{c1}$.

$(-12,25)_{10}$	(……..)$_{C2}$

$(-12,25)_{10}$ = (?)$_{S+VA}$

Sur 9 bits dont 3 bits pour la partie décimale :

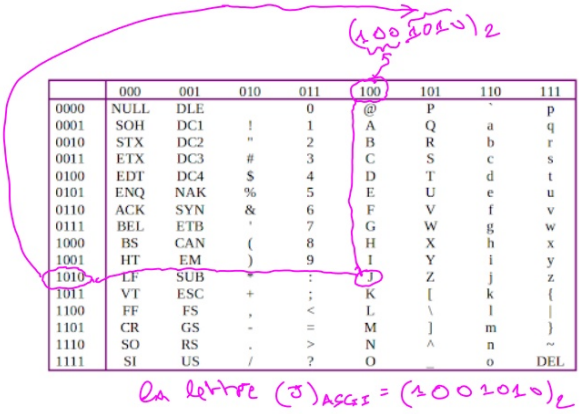
$(+12,25) = (0\ 1100{,}010)_{C2}$

ce qui donne $(-12,25) = (1\ 0011{,}110)_{c2}$.

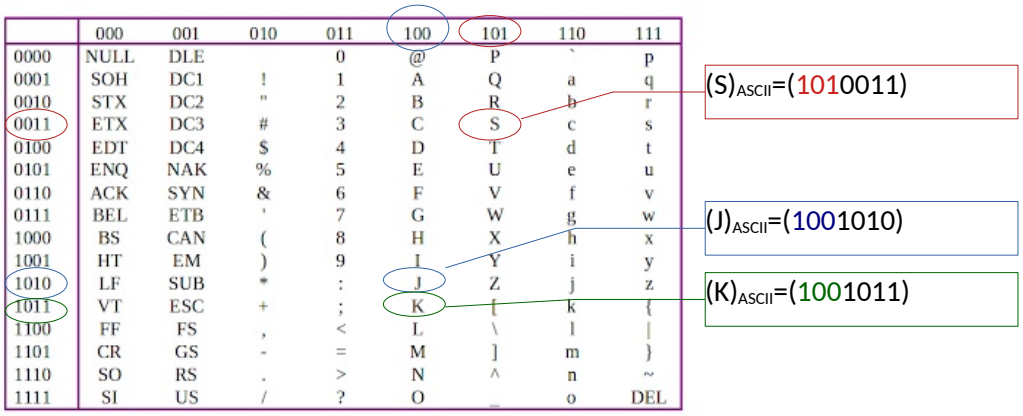
Q6 – Codage des caractères :

En vous référant à la table ASCII standard sur 7 bits donnez le code (en binaire) du mot « JSK »

Voici comment je trouve le code ASCII (7bits) de la lettre **J** :



	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EDT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL



	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EDT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

$(S)_{ASCII}=(1010011)$

$(J)_{ASCII}=(1001010)$

$(K)_{ASCII}=(1001011)$

Ce qui donne :

$(JSK)_{ASCII}$ = (1000011 1001010 1001011)