Travaux dirigés (D)

Le codage des caractères

À partir du moment (années 1940) où il est clairement apparu que les ordinateurs n'allaient plus servir à traiter que des informations numériques brutes, le problème du codage des caractères, s'est posé.

1 - ASCII - 7 bits

ASCII - American Standart Code for Information Interchange :

C'est la première norme, adoptée après de nombreuses révisions, par l'« American National Standards Institute » (ANSI) dans les années 1960.

À l'époque les ordinateurs fonctionnaient en 8 bits, et 2^7 = 128 caractères étaient suffisants pour coder les lettres majuscules, minuscules, chiffres et ponctuations... 1 bit de parité étant conservé pour la détection des erreurs de transmission.

Le document ci-dessous date de 1972 :

USASCII code chart

b_7 b_6 b_5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits b_4 b_3 b_2 b_1 ↓	Column → 0	1	2	3	4	5	6	7
Row ↓								
0 0 0 0	0 NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1 SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2 STX	DC2	"	2	B	R	b	r
0 0 1 1	3 ETX	DC3	#	3	C	S	c	s
0 1 0 0	4 EOT	DC4	$	4	D	T	d	t
0 1 0 1	5 ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6 ACK	SYN	&	6	F	V	f	v
0 1 1 1	7 BEL	ETB	'	7	G	W	g	w
1 0 0 0	8 BS	CAN	(	8	H	X	h	x
1 0 0 1	9 HT	EM	)	9	I	Y	i	y
1 0 1 0	10 LF	SUB	*	:	J	Z	j	z
1 0 1 1	11 VT	ESC	+	;	K	[	k	{
1 1 0 0	12 FF	FS	,	<	L	\	l	|
1 1 0 1	13 CR	GS	-	=	M	]	m	}
1 1 1 0	14 SO	RS	.	>	N	^	n	~
1 1 1 1	15 SI	US	/	?	O	_	o	DEL

Source : https ://en.wikipedia.org/wiki/ASCII#/media/File :US-ASCII_code_chart.png)

Par exemple, le caractère 'A', est le caractère numéro `100 0001`.
C'est-à-dire : $1\times2^6+0\times2^5+\ldots+0\times2^1+1\times2^0 = 65$ en décimal.
Le tableau permet d'en lire l'écriture hexadécimale (colonne ligne) : `41`.
On peut vérifier par le calcul : $4\times16^1+1\times16^0 = 65$.

On pourra donc afficher la lettre « A » directement en la spécifiant entre guillemets simples ou à l'aide de son code ASCII (écrit en décimal, binaire ou hexadécimal) avec une opération de cast vert le type **char** :

```
println('A');
println((char)65);
println((char)0x41);
println((char)0b1000001);
```

Ex. D1 *Écrire un programme pour vérifier que Java's Cool utilise encore les mêmes références pour coder ses 128 premiers caractères (faire afficher les 128 caractères dans la console).*

Réciproquement, on peut également utiliser un cast d'un caractère vers un entier pour obtenir son code ASCII.

Ex. D2 *Déterminer à l'aide d'un cast, le code ASCII du caractère* `'a'`.

2 - ASCII étendu

L'amélioration de la fiabilité des transmissions et les besoins des européens, par exemple d'accentuer les lettres, ont abouti à l'utilisation du $8^{\text{ème}}$ bit et donc à doubler le nombre de caractères codés.

latin-1

C'est la norme ISO 8859-1 pour l'encodage des caractères.
C'est celle utilisée par Java's Cool.

Ex. D3 *Vérifier cette affirmation en modifiant affichant 256 premier caractères avec Java's Cool. (Comparer avec ISO 8859-1 dont on cherchera une représentation sur internet.)*

3 - Unicode

Chaque langue ayant ses spécificités, de nombreuses normes ont vu le jour, compliquant de plus en plus la tache des développeurs pour se faire comprendre par le plus grand nombre. (Un texte écrit dans un standard devenant difficilement lisible dans un autre.)

Un consortium composé d'informaticiens, de chercheurs, de linguistes et de personnalités du monde de la politique... s'est attelé à unifier toutes les pratiques en un seul et même système : l'Unicode.

L'Unicode

Codé sur 16 bits (et toujours compatible avec l'ASCII) il permet de représenter les caractères spécifiques à toutes les langues.

```
println('\u0041'); //Encore et toujours notre A
```

Un bon compromis (qui s'impose sur internet) est l'UTF-8.

UTF-8

C'est un encodage des caractères qui utilise les codes ASCII par défaut (8 bits par caractère) plus un caractère spécial indiquant que le suivant est à lire en Unicode (en 16 bits).

On voit parfois apparaître ces caractères sur des pages ou courriels dont l'encodage est mal spécifié. La phrase « C'est très raté ! » par exemple devient « C'est trÃ¨s ratÃ© ! ».

Ex. D4 *En utilisant un navigateur internet, sur la page de votre site préféré,*

1. *chercher le menu relatif à l'« Affichage » et vérifier le type d'« Encodage des caractères ».*
2. *changer Unicode en Occidental (ou autre chose). Que constate-t-on ?*
3. *afficher le code source de la page et chercher la ligne <meta charset=...*

   De quel encodage s'agit-il ?

- Exercices d'entraînement

Ex. D5 *En majuscule*

Écrire la fonction `majuscule()` *qui prend un caractère entre* `a` *et* `z` *en paramètre et renvoie le caractère correspondant en majuscule (entre* `A` *et* `Z`*).*

Ex. D6 *Le chiffrage de Caesar*

En 58 avant Jésus-Christ, Jules César se lançait à la conquête de la Gaule. Pour communiquer avec ses généraux, il imagina des procédés de chiffrement. L'un d'eux consistait à permuter l'alphabet en remplaçant chacune des lettres par celle située un certain nombre de rangs plus loin.

Par exemple un décalage de 3 rangs remplacera la lettre A par D, B par E, C par F et ainsi de suite...

Écrire la fonction `code()` *qui prend un caractère entre* `A` *et* `Z` *en paramètre et renvoie le caractère correspondant au décalage proposé ci-dessus. Vérifier son fonctionnement avec* `A` *et* `Z`*.*

Ex. D7 *Type de caractère*

Écrire la fonction `typeCarac()` *qui prend un caractère de la table ASCII - 7 bits en paramètre et affiche un message pour dire si c'est une lettre, un chiffre ou un symbole.*

- Solutions des exercices

Retour sur l'exercice D1 128 caractères

```
void main(){
  for (int car=0;car<128;car++){
    print((char)car);
    // Retour à la ligne tous les 32 caractères :
    if (car%32==31) println();
  }
}
```

Retour sur l'exercice D2 cast `char` vers `int`

```
void main(){
  print("Le code de 'a' est : "+(int)'a');
}
```

Retour sur l'exercice D3 256 caractères

Il suffit de remplacer `128` par `256`. La comparaison valide l'affirmation selon laquelle c'est l'encodage utilisé par Java's Cool.

Retour sur l'exercice D4 Sur internet

1. La plupart des pages sont actuellement codées en Unicode.
2. Les caractères accentués par exemple ne s'affichent plus correctement si l'encodage sélectionné n'est pas le bon.
3. Sur la plupart des pages, on trouve `<meta charset="utf-8"/>`

Retour sur l'exercice D5 En majuscule

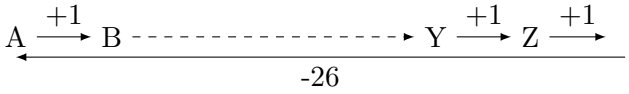
Comme `'A'` est codé `65` et `'a'` est codé `97`, il suffit d'opérer un décalage de 32.

```
char majuscule(char lettre) {
  int temp = (int)lettre;// Récupération du code ASCII
  temp = temp - 32;      // Décalage pour que a devienne A
  return (char) temp;    // Conversion par la table ASCII
}
```

Retour sur l'exercice D6 Chiffrage de Caesar

On a vu comment passer d'un code à un caractère et réciproquement à l'aide de casts. Pour coder un 'A' il suffit d'augmenter la valeur ASCII correspondante.

Mais pour décaler le 'Z', il faut revenir en arrière et repartir de 'A' dans le décalage.




```
char code(char lettre) {
  int c = (int)lettre; // Récupération du code ASCII
  c = c + 3;           // Chiffrage de Caesar
  if (c>(int)'Z') c = c - 26;  // gestion des dépassements
  return (char)c;      // Conversion par la table ASCII
}

void main(){
  println(code('A'));
  println(code('Z'));
}
```

Retour sur l'exercice D7 Type de caractère

Si l'on cherche à afficher directement le message selon le type testé, on se retrouve rapidement à imbriquer des structures conditionnelles.

On a choisi ci-dessous de coder préalablement le type de caractère sur un entier. Un **switch** sur ce codage permet de conclure plus facilement.

```
void typeCarac(char car) {
  int code = (int) car; // Code ASCII du caractère
  int type = 0;
  if (code>=48&&code<=57) type=2; // un chiffre
  if (code>=65&&code<=90) type=1; // une lettre (majuscule)
  if (code>=97&&code<=122) type=1;// une lettre (minuscule)
  print("Le caractère '"+car+"' est ");
  switch (type) {
    case 1 : println("une lettre."); break;
    case 2 : println("un chiffre."); break;
    default : println("un symbole.");
  }
}
```

Travaux dirigés (E)

Les chaînes de caractères

`String`

Ce n'est pas un type primitif, mais une *classe*, c'est-à-dire le regroupement sous un même nom de plusieurs données (appelées *attributs*) et fonctions (appelées *méthodes*)

Cette classe représente une chaine de caractères, un peu comme un tableaux de type **`char`**.

Les données sont les caractères (**`char`**) qui composent la chaîne.

1 - Construction

En tant que *classe*, un `String` peut être défini (on dit alors *instancié*) par le mot-clef new suivit d'une méthode de construction du même nom :

```
String chaine = new String("texte");
```

ou encore :

```
char lettres[] = {'t', 'e', 'x', 't', 'e'};
String mot = new String(lettres);
```

Mais cette classe est particulière. Cette construction peut également se faire comme pour un type primitif :

```
String chaine = "texte";
```

On a par ailleurs déjà eu l'occasion de *concaténer* des chaines de caractères, avec l'opération « **`+`** » :

```
String test ="Début"+" et "+"fin.";
```

2 - Méthodes

Utilisation de méthodes

On applique une méthode à une instance à l'aide d'un point « . »
`instanceDeLaClasse.nomDeLaMethode(parametres)`.

⋆ `length()`

Renvoie la longueur (**`int`**) de la chaîne de caractères :

```
int nombre = "texte".length()
```

On n'oublie pas les « `()` » puisque `length()` est une fonction (à ne pas confondre avec l'*attribut* `length` d'un tableau).

Ex. E1 *Mot de passe*
Écrire une fonction qui affiche un mot (passé en paramètre) en remplaçant chaque lettre par une étoile.

⋆ `charAt(int i)`
Renvoie le caractère (`char`) de rang `i` (Le premier caractère étant de rang 0 comme pour les tableaux).

```
char lettre = "texte".charAt(1);
// lettre vaut maintenant 'e'
```

Ex. E2 *Palindrome*

1. *Écrire une fonction qui affiche un texte à l'envers.*
   ("texte" → "etxet"*).*
2. *Écrire une fonction qui détermine si un texte est un palindrome.*

⋆ `substring(int d, int f)`
Renvoie la sous-chaîne de caractères (`String`) commençant au rang `d` (inclus) et finissant au rang `f` (exclu).

```
String milieu = "texte".substring(2,4)
// milieu vaut maintenant "xt".
```

Ex. E3 *Écrire une fonction qui n'écrit que la moitié de la chaine de caractères passée en paramètre et complète par des étoiles.*

⋆ `equals(Object unObjet)`
Renvoie `true` ou `false` (un `boolean`) selon si la chaine de caractères est ou non identique à `unObjet`.
On l'emploiera avec un `String` qui est bien un `Object` en Java.

```
boolean test = "test ".equals("test")
// test vaut maintenant : false (à cause de l'espace en trop
    sur la première chaîne de caractères).
```

Ex. E4 *Écrire une fonction `compte` qui calcule le nombre de fois où une chaine de caractères est comprise dans une autre. Exemple :*
`compte(`"Ton tonton tond ton tonton"*,*"ton"*) vaut 6.*

(La classe `String` possède de nombreuses autres méthodes. Il ne faut pas hésiter à aller en chercher la liste sur internet en cas de besoin.)