TP 1 : Table de caractères ASCII

Objectifs du TP

1. travailler la mise en forme d'un affichage.

1 Le code ASCII et le code Latin-1

Le code ASCII (American Standard Code for Information Interchange), créé en 1963, a longtemps été utilisé pour le codage des caractères alphanumériques. C'est un code qui définit 128 caractères, codés en binaire sur 7 bits de 0 (0000000_2) à 127 (1111111_2).

Ce code, qui ne contient aucun caractère accentué, a été étendu en différents codes dont les caractères sont codés sur 8 bits. L'un d'entre eux est le code ISO 8859-1, connu aussi sous le nom de Latin-1. Ce code contient les principaux caractères des langues d'Europe occidentale.

L'affichage des caractères du type CHAR utilise le codage ISO 8859-1.

Dans tout ce qui suit, il va s'agir de réaliser des affichages sous diverses formes de l'ensemble des caractères du type CHAR accompagnés de leur code, en se limitant à ceux dont le code est compris entre 0 et 127 (c'est–à–dire au code ASCII).

Exercice 1 : *À la découverte d'une nouvelle fonction*

Q 1 . Testez l'instruction

writeln(CHAR(k));

avec k entier compris entre 65 et 90.

Q 2 . Que vaut CHAR(k) ?

2 Affichage en une colonne

Exercice 2 : *Le programme `table1`*

Réalisez un programme (sauvegardé dans un fichier nommé `table1.pas`) qui affiche tous les caractères ASCII accompagnés de leur valeur ordinale ou code.

Voici un extrait de l'affichage que votre programme doit produire :

```
...

   . :  46
   / :  47
   0 :  48
   1 :  49
   2 :  50
   3 :  51
   4 :  52
   5 :  53
   6 :  54
   7 :  55
   8 :  56
   9 :  57
   : :  58
   ; :  59
   < :  60
   = :  61
   > :  62
   ? :  63
   @ :  64
   A :  65
```

```
   B :  66
   C :  67

...
```

Vous pourrez utiliser
– l'instruction d'affichage

write (c : k) ;

qui affiche la valeur de la variable c (de type entier ou caractère) en lui réservant k emplacements. Suggestion du choix de la valeur de k : $k = 5$ pour l'affichage des caractères, et $k = 4$ pour les codes.
– et la fonction prédéfinie CHAR.

Exercice 3 : *Gestion des caractères non affichables*

Certains caractères ne sont pas affichables. Ce sont les caractères dont le code est compris entre 0 et 31, et le caractère de code 127.

Q 1 . Faites une copie de votre fichier `table1.pas` dans un fichier `table2.pas`, et réalisez ce qui suit dans ce nouveau fichier.

Q 2 . Réalisez une procédure qui affiche le caractère passé en paramètre s'il est affichable, et qui affiche `NA` sinon. L'affichage se faisant sur cinq emplacements.

Q 3 . Utilisez cette procédure pour réaliser l'affichage de tous les caractères accompagnés de leur code.

Extrait de l'affichage

```
 ...

NA :  27
NA :  28
NA :  29
NA :  30
NA :  31
   :  32
 ! :  33
 " :  34
 # :  35
 $ :  36
 % :  37
 & :  38
 ' :  39
 ( :  40

 ...
```

3 Affichage en plusieurs colonnes (1)

On souhaite réaliser un affichage sous la forme

```
NA :   0   NA :   1   NA :   2   NA :   3   NA :   4   NA :   5   NA :   6   NA :   7
NA :   8   NA :   9   NA :  10   NA :  11   NA :  12   NA :  13   NA :  14   NA :  15
NA :  16   NA :  17   NA :  18   NA :  19   NA :  20   NA :  21   NA :  22   NA :  23
NA :  24   NA :  25   NA :  26   NA :  27   NA :  28   NA :  29   NA :  30   NA :  31
   :  32    ! :  33    " :  34    # :  35    $ :  36    % :  37    & :  38    ' :  39
 ( :  40    ) :  41    * :  42    + :  43    , :  44    - :  45    . :  46    / :  47
 0 :  48    1 :  49    2 :  50    3 :  51    4 :  52    5 :  53    6 :  54    7 :  55
 8 :  56    9 :  57    : :  58    ; :  59    < :  60    = :  61    > :  62    ? :  63
 @ :  64    A :  65    B :  66    C :  67    D :  68    E :  69    F :  70    G :  71
 H :  72    I :  73    J :  74    K :  75    L :  76    M :  77    N :  78    O :  79
 P :  80    Q :  81    R :  82    S :  83    T :  84    U :  85    V :  86    W :  87
 X :  88    Y :  89    Z :  90    [ :  91    \ :  92    ] :  93    ^ :  94    _ :  95
 ` :  96    a :  97    b :  98    c :  99    d : 100    e : 101    f : 102    g : 103
```

```
 h : 104    i : 105    j : 106    k : 107    l : 108    m : 109    n : 110    o : 111
 p : 112    q : 113    r : 114    s : 115    t : 116    u : 117    v : 118    w : 119
 x : 120    y : 121    z : 122    { : 123    | : 124    } : 125    ~ : 126   NA : 127
```

La table est donc présentée sur 16 lignes de huit colonnes.

Exercice 4 : *Le programme `table3`*

Q 1 . Faites une copie de votre fichier `table2.pas` dans un fichier `table3.pas`, et réalisez ce qui suit dans ce nouveau fichier.

Q 2 . Définissez une constante `NBC` pour le nombre de colonnes dont la valeur est fixée à 8.

Q 3 . Définissez une constante `NBL` pour le nombre de lignes en utilisant cette expression précédente.

Q 4 . Déterminez les instructions qui permettent d'obtenir l'affichage souhaité.

4 Affichage en plusieurs colonnes (2)

On veut toujours produire un affichage sur 4 colonnes et 32 lignes, les caractères étant rangés par colonne cette fois-ci.

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

Exercice 5 : *Le programme `table4`*

Q 1 . Faites une copie de votre fichier `table3.pas` dans un fichier `table4.pas`, et réalisez ce qui suit dans ce nouveau fichier.

Q 2 . Que faut-il modifier au programme `table3` pour obtenir l'affichage voulu ?

5 Affichage au format HTML

Il s'agit de produire maintenant un affichage au format HTML afin de pouvoir visualiser la table produite avec un navigateur internet, comme par exemple cette page.

Cette partie est facultative. La présentation (même sommaire) du langage HTML qui y est faite ne fait pas partie du cours d'API1.

Présentation du format HTML Le langage HTML (HyperText Markup Language) est le langage informatique créé et utilisé pour écrire les pages Web. C'est un langage de description de pages qui utilise des *balises*.

Tout document HTML est écrit sur le schéma

```
<html>
<head>

  <!-- Ici on trouve quelques éléments d'entête -->

</head>
<body>

  <!-- Ici on trouve le contenu de la page -->

</body>
</html>
```

Les éléments entre chevrons (<, >) sont appelés *balises*. La plupart des balises vont par couple : une balise ouvrante (comme <**html**>) et une balise fermante (comme </**html**>) qui se distingue par la barre oblique (/).

Exercice 6 : Q 1 . Examinez le code source de cette page que vous lisez actuellement (avec FIREFOX, il suffit de choisir l'option `Code source de la page` du menu contextuel obtenu à l'aide d'un clic droit de la souris.).
Q 2 . Repérez les balises <**html**>, </**html**>, <**body**>, </**body**>, <**head**> et </**head**>. Il y a bien évidemment beaucoup d'autres balises que nous ne présenterons pas.
Q 3 . Recherchez l'élément donné entre les balises <**title**> et </**title**> situé dans l'entête. Que vaut-il ? Comparez avec ce qu'on peut lire dans le bandeau supérieur de FIREFOX.

Présentation des outils Voici quelques procédures écrites en PASCAL qui permettent de réaliser un programme produisant le code source d'une page HTML.

- Début d'un document HTML

```
// debuterHTML(titre) écrit sur la sortie standard
// le debut d'un document HTML.
// le paramètre titre donne le titre du document.
procedure debuterHTML(const nomPage : STRING);
begin
    writeln('<html>');
    writeln('<head>');
    writeln('<title>',nomPage,'</title>');
    writeln('</head>');
    writeln('<body>');
end {ouvrirHTML};
```

  Cette procédure est à utiliser pour débuter un document HTML. Le titre passé en paramètre donne le titre de ce document (celui qui apparaît dans le bandeau supérieur du navigateur lors de l'affichage du document).

- Fin d'un document HTML

```
// finirHTML() écrit sur la sortie standard
// la fin d'un document HTML.
procedure finirHTML;
```

```
begin
    writeln('</body>');
    writeln('</html>');
end {finirHTML};
```

Cette procédure est à utiliser pour terminer un document HTML.

- Titre de section.

```
// H1(libelle) écrit sur la sortie standard
// le libellé passé en paramètre entre les balises
// <h1> et </h1>.
procedure H1(const libelle : STRING);
begin
    writeln('<h1> ',libelle,'</h1>');
end {H1};
```

Cette procédure est à utiliser pour nommer une section, ou une partie, du document.

- Débuter une table.

```
// debuterTable(b,l) écrit sur la sortie standard
// le code HTML de début d'une table.
// b indique l'épaisseur des bords des cellules.
// l donne la largeur de la table en pourcentage
// de la largeur de la fenêtre du navigateur.
procedure debuterTable(const bordure : CARDINAL;
                       const largeur : REAL);
begin
    writeln('<table border="',bordure,'" width="',largeur:6:2,'%">');
end {debuterTable};
```

Cette procédure est à utiliser pour débuter une table.

- Finir une table.

```
// finirTable() écrit sur la sortie standard
// le code HTML de fin d'une table.
procedure finirTable;
begin
    writeln('</table>');
end {finirTable};
```

Cette procédure est à utiliser pour finir une table.

- Débuter une ligne d'une table.

```
// debuterLigne() écrit sur la sortie standard
// le code HTML du début d'une ligne de table.
procedure debuterLigne;
begin
    writeln('<tr>');
end {debuterLigne};
```

Cette procédure est à utiliser pour débuter une ligne.

- Finir une ligne d'une table.

```
// finirLigne() écrit sur la sortie standard
// le code HTML de fin d'une ligne de table.
procedure finirLigne;
begin
    writeln('</tr>');
end {finirLigne};
```

Cette procédure est à utiliser pour finir une ligne.

- Débuter une cellule d'une table.

```
// debuterCellule(c) écrit sur la sortie standard
// le code HTML de début d'une cellule.
```

```
// Le paramètre c décrit la couleur de fond de cette cellule.
procedure debuterCellule(const couleur : STRING);
begin
    write('<td bgcolor="',couleur,'">');
end {debuterCellule};
```

Cette procédure est à utiliser pour débuter une cellule. La couleur passée en paramètre peut être un nom de couleur à choisir parmi quelques noms prédéfinis (comme `black`, `white`, `blue`, `red`, `yellow`, ...), ou bien une description RVB en hexadécimal des composantes Rouge, Vert et Bleu de la couleur voulue (cf exemple plus bas).

– Finir une cellule d'une table.

```
// finirCellule() écrit sur la sortie standard
// le code HTML de fin d'une cellule.
procedure finirCellule;
begin
    writeln('</td>');
end {finirCellule};
```

Un exemple

Programme Pascal	Code HTML produit

begin
debuterHTML('Code ASCII');
H1('Table des caractères ASCII');
debuterTable(5,50);
// 1ère ligne
debuterLigne();
debuterCellule('#777777');
write('Case 1,1');
finirCellule();
debuterCellule('#AAAAAA');
write('Case 1,2');
finirCellule();
debuterCellule('#CCCCCC');
write('Case 1,3');
finirCellule();
finirLigne();
 	
// 2ème ligne
debuterLigne();
debuterCellule('#777777');
write('Case 2,1');
finirCellule();
debuterCellule('#AAAAAA');
write('Case 2,2');
finirCellule();
debuterCellule('#CCCCCC');
write('Case 2,3');
finirCellule();
finirLigne();
 	
finirTable();
finirHTML();
end.
	<**html**>
	<**head**>
	<**title**>Code ASCII</**title**>
	</**head**>
	<**body**>
	<**h1**> Table des caractères ASCII </**h1**>
	<**table border**="5" **width**="50.00%">
	<**tr**>
	<**td bgcolor**="#777777">Case 1,1</**td**>
	<**td bgcolor**="#AAAAAA">Case 1,2</**td**>
	<**td bgcolor**="#CCCCCC">Case 1,3</**td**>
	</**tr**>
	<**tr**>
	<**td bgcolor**="#777777">Case 2,1</**td**>
	<**td bgcolor**="#AAAAAA">Case 2,2</**td**>
	<**td bgcolor**="#CCCCCC">Case 2,3</**td**>
	</**tr**>
	</**table**>
	</**body**>
	</**html**>

Exercice 7 : Q 1 . Réalisez le programme `table5.pas` donné en exemple ci-dessus.
Q 2 . Éxécutez le programme en dirigeant la sortie vers un fichier nommé `tableASCII.html`.

Q 3 . Utilisez FIREFOX pour visualiser la page HTML produite par votre programme (menu FICHIER > OUVRIR UN FICHIER).

Q 4 . Modifier le programme `table5.pas` pour obtenir la table des caractères ASCII. (vous pouvez examiner le code HTML de cette page)