La bibliothèque **bigint**

Cette bibliothèque permet d'utiliser des nombres de plus grande taille et d'explorer les nombres premiers. Il s'agit en fait d'une interface pour la bibliothèque **largeint** de S. J. Schaper (réf. 1).

Les exemples se trouvent dans le dossier **exemples\fbcroco\bigint** qui contient aussi la documemtation de la bibliothèque (fichier **bigint.htm**). Nous nous contenterons de commenter l'un des programmes fournis.

Les nombres de la forme $2^n - 1$ sont appelés *nombres de Mersenne*. Si n est premier, $2^n - 1$ peut l'être aussi mais ce n'est pas garanti ! Le programme **test8.bas** étudie ces nombres pour $n = 60$ à $n = 70$ en essayant de les décomposer en facteurs premiers :

```
#include "bigint.bi"

' ------------------------------------------------------------------------
' Nombres de Mersenne - Decomposition en facteurs premiers
' ------------------------------------------------------------------------

enum
  N4 = 4, N5      ' Indices des nombres
end enum

dim nDigits%      ' Nombre de chiffres max.

nDigits = BigInit(N5 + 1)

' ------------------------------------------------------------------------
' Programme principal
' ------------------------------------------------------------------------

dim n%

SetNum N4, 2          ' (N4) = 2

for n = 60 to 70
  BigPow N4, n, N5    ' (N5) = 2^n
  BigInc N5, -1       ' (N5) = 2^n - 1

  print using "2^## - 1 = "; n;
  print using "\                      \ = "; GetStr(N5);
  print BigPrimDec(N5)
next n
```

La première ligne inclut le fichier d'en-têtes de la bibliothèque. L'initialisation de la bibliothèque se fait en deux temps :

1. A l'aide de l'instruction **enum**, on définit une série de constantes qui correspondent aux indices des nombres entiers que l'on veut utiliser (ces entiers sont stockés dans un tableau, dans l'espace mémoire de la bibliothèque). Nous commençons à l'indice 4 car la procédure de décomposition en facteurs premiers réserve les 4 premiers entiers, d'indices 0 à 3, pour son propre usage.

Cette étape n'est pas obligatoire : on aurait pu utiliser directement les valeurs numériques des indices ; les constantes ne sont là que pour la clarté.

2. On appelle ensuite la fonction **BigInit** avec comme paramètre le nombre total d'entiers que l'on veut utiliser : ici **N5 + 1** puisqu'on commence à zéro. Cette étape est indispensable, contrairement à la précédente.

La fonction retourne le nombre de chiffres maximum que peut avoir un entier, soit près de 40000 dans cet exemple ! On aurait pu ignorer cette valeur en utilisant la fonction comme une procédure, les parenthèses devenant alors facultatives :

```
BigInit N5 + 1
```

L'instruction **SetNum N4, 2** donne la valeur 2 à l'entier d'indice **N4** ; nous notons **(N4)** cet entier.

L'instruction **BigPow N4, n, N5** élève l'entier **(N4)** à la puissance **n** et dépose le résultat dans **(N5)** Notons que le même indice ne peut figurer qu'une fois dans les paramètres : on ne peut pas écrire **BigPow N4, n, N4**

L'instruction **BigInc N5, -1** incrémente l'entier **(N5)** de -1, c'est-à-dire qu'elle le décrémente de 1 ! (il n'y a pas d'instruction **BigDec**)

Les fonctions **GetStr** et **BigPrimDec** fournissent, sous forme de chaînes de caractères, la valeur d'un entier et sa décomposition en facteurs premiers.

Au prix d'un temps de calcul un peu long, on obtient les résultats suivants pour n = 60 à 70 :

```
2^60 - 1 =    1152921504606846975 = 3^2 * 5^2 * 7 * 11 * 13 * 31 * 41 * 61 * 151 * 331 * 1321
2^61 - 1 =    2305843009213693951 = 2305843009213693951
2^62 - 1 =    4611686018427387903 = 3 * 715827883 * 2147483647
2^63 - 1 =    9223372036854775807 = 7^2 * 73 * 127 * 337 * 92737 * 649657
2^64 - 1 =   18446744073709551615 = 3 * 5 * 17 * 257 * 641 * 65537 * 6700417
2^65 - 1 =   36893488147419103231 = 31 * 8191 * 145295143558111
2^66 - 1 =   73786976294838206463 = 3^2 * 7 * 23 * 67 * 89 * 683 * 20857 * 599479
2^67 - 1 =  147573952589676412927 = 193707721 * 761838257287
2^68 - 1 =  295147905179352825855 = 3 * 5 * 137 * 953 * 26317 * 43691 * 131071
2^69 - 1 =  590295810358705651711 = 7 * 47 * 178481 * 10052678938039
2^70 - 1 = 1180591620717411303423 = 3 * 11 * 31 * 43 * 71 * 127 * 281 * 86171 * 122921
```

Il n'y a qu'un seul nombre premier dans la série : $2^{61} - 1$. La primalité de ce nombre a été démontrée en 1883 (voir réf. 2). Le cas de n = 67 a donné lieu à une anecdote également rapportée dans la réf. 2 et reproduite ici :

La conférence du mathématicien silencieux

Au XVII^e siècle, le père Mersenne affirma que $2^{67} - 1$ est un nombre premier, ce qu'on crut vrai pendant 250 ans. Lors d'une réunion de la Société américaine de mathématiques en 1903, Franck Nelson Co1e, de l'Université Columbia, était annoncé pour une conférence intitulée Sur la factorisation des grands nombres. *Eric Temple Bell, qui assistait à la séance, raconte : « Cole - qui n'était pas un homme bavard - avança vers le tableau, écrivit soigneusement les puissances de deux jusqu'à la soixante-septième. Retira 1 à cette dernière, ce qui donna le monstrueux résultat 147573952589676412927. Puis il gagna une partie propre du tableau et, toujours sans un mot, calcula le produit 193707721 x 761838257287. Les deux résultats coincidaient. La conjecture de Mersenne, si c'en était une, tombait dans les limbes de la mythologie mathématique. Pour la première fois une assemblée de la Société américaine de mathématiques applaudit vigoureusement un conférencier. Cole retourna à son siège sans prononcer un seul mot. Personne ne lui posa de questions. »*

Vous remarquerez que FBCroco retrouve bien les résultats de Cole, et tout aussi silencieusement !

Pour les nombres $n > 70$ l'algorithme de factorisation utilisé par FBCroco est souvent pris en défaut mais peut donner de bons résultats si les facteurs ne sont pas trop grands. A titre de curiosité, voici le résultat pour $n = 210$:

```
2^210 - 1 = 1645504557321206042154969182557350504982735865633579863348609023
          = 3^2 * 7^2 * 11 * 31 * 43 * 71 * 127 * 151 * 211 * 281 * 331 * 337 *
            5419 * 29191 * 86171 * 106681 * 122921 * 152041 * 664441 * 1564921
```

C'est impressionnant, mais on est loin des résultats obtenus par les experts qui utilisent des algorithmes spécialisés et répartissent le temps de calcul sur de nombreux ordinateurs ! (voir les réf. 2 et 3).

Références

1. Big-integer arithmetic in BASIC : Page de la bibliothèque **largeint**
2. J. P. Delahaye, *Merveilleux nombres premiers*, Editions Belin, 2012
3. Page de Wikipedia sur les nombres premiers : https://fr.wikipedia.org/wiki/Nombre_premier