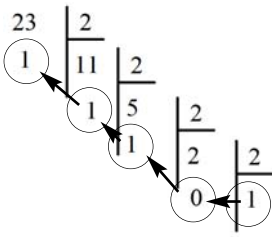


Numération binaire : la base 2

Le problème



L'écriture en base 2 (ou *écriture binaire*) permet d'écrire tout entier naturel avec uniquement des 0 et des 1. Le procédé pour transformer un entier naturel non nul en base 2 est simple. Prenons l'exemple de 23 : on effectue les divisions euclidiennes par 2 de 23, puis des quotients successifs obtenus, jusqu'à obtenir pour quotient 1 ; il reste à lire les restes successifs, de droite à gauche (à partir de 1, dernier quotient non nul).

Ainsi, l'écriture binaire de 23 est **10111**.

Cela signifie que $23 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2 + 1$.

Complément culturel

L'informatique repose sur le système binaire, car dans son fonctionnement physique un ordinateur ramène tout à des 0 et des 1 (le courant passe ou ne passe pas). Il travaille donc en base 2 (ou dans le *système binaire*).

Les programmes

Python



La longueur d'une liste est déterminée en lui appliquant la fonction « len ». Lorsqu'une suite L est définie, son premier terme est L[0] ; par conséquent, si l'on note d la longueur de la suite, le dernier est L[d - 1].

le programme

```
>>> def base2(N):
    Liste=[]
    ListeRésultat=[]
    q=int(N/2)
    while q!=1:
        Liste.append(N%2)
        N=q
        q=int(N/2)
    Liste.append(N%2)
    Liste.append(1)
    i=0
    d=len(Liste)
    while i<d:
        ListeRésultat.append(Liste[d-1-i])
        i=i+1
    return ListeRésultat
```

le résultat

```
>>> base2(23)
[1, 0, 1, 1, 1]
>>> base2(15100)
[1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0]
```

SCRATCH

Scratch



Les restes des divisions successives sont placés dans une liste nommée 'Liste' jusqu'à obtenir un quotient égal à 1, que l'on rajoute à la liste. À la fin du programme, on construit une liste nommée 'Listerésultat', obtenue en lisant 'Liste' à l'envers.

```

quand [ ] pressé
  répéter longueur de Liste fois
    supprimer 1 de Liste
    supprimer 1 de Listerésultat
  demander N=? et attendre
  à N attribuer réponse
  à q attribuer N - N mod 2 / 2
  répéter jusqu'à q = 1
    ajouter N mod 2 à Liste
    à N attribuer q
    à q attribuer N - N mod 2 / 2
  ajouter N mod 2 à Liste
  ajouter 1 à Liste
  à d attribuer longueur de Liste
  à i attribuer 0
  répéter jusqu'à i = d
    ajouter élément d - i de Liste à Listerésultat
    changer i par 1
  dire Listerésultat pendant 5 secondes
  
```



AlgoBox

Même début que sous Scratch pour créer la liste nommée 'list'. Il ne reste plus qu'à la lire en sens inverse. Le logiciel ne permettant pas une édition directe de cette liste inverse, on effectue la lecture pas à pas.

```

VARIABLES
  N EST_DU_TYPE NOMBRE
  q EST_DU_TYPE NOMBRE
  d EST_DU_TYPE NOMBRE
  i EST_DU_TYPE NOMBRE
  k EST_DU_TYPE NOMBRE
  List EST_DU_TYPE LISTE

DEBUT_ALGORITHME
  LIRE N
  q PREND_LA_VALEUR floor(N/2)
  i PREND_LA_VALEUR 1
  TANT_QUE (q!=1) FAIRE
    DEBUT_TANT_QUE
    List[i] PREND_LA_VALEUR N-2*floor(N/2)
    i PREND_LA_VALEUR i+1
    N PREND_LA_VALEUR q
    q PREND_LA_VALEUR floor(N/2)
    FIN_TANT_QUE
  List[i] PREND_LA_VALEUR N-2*floor(N/2)
  List[i+1] PREND_LA_VALEUR 1
  POUR k ALLANT_DE 1 A i+1
    DEBUT_POUR
    AFFICHER List[i+2-k]
    PAUSE
    FIN_POUR
  FIN_ALGORITHME
  
```

Prolongements

Nous vous proposons de créer vous-même le programme « Décimal » (plus facile), qui permet de réaliser l'opération inverse : donner la valeur dans le système décimal d'un nombre écrit en base 2. Voici deux exemples de résultats que vous devez obtenir :

le résultat

```

>>> Décimal (base2 (15100))
15100
>>> Décimal (base2 (28121964))
28121964
  
```