

TD Informatique n° 3

Le code bibinaire

I - Coder en bibinaire

a. Qu'est-ce que c'est ?



Boby LAPOINTE, célèbre chanteur français, était aussi mathématicien à ses heures. Ayant trouvé le code binaire trop compliqué à utiliser, il inventa le code... bibinaire. Il suffit de remplacer les chiffres par des lettres. On commence par couper le nombre écrit en binaire en paquets de 2. S'il y a un nombre impair de chiffres, on rajoute un zéro à gauche, ce qui ne modifie pas la valeur de nombre. On commence par le premier groupe de deux chiffres le plus à droite. On remplace 00 par O, 01 par A, 10 par E, 11 par I.

Puis on prend le paquet de deux chiffres suivants en se déplaçant de droite à gauche. On remplace 00 par H, 01 par B, 10 par K, 11 par D.

Pour le paquet suivant, on recommence avec les voyelles. S'il y a encore un groupe, on remplace par une consonne, etc.

b. Numération en base 2

Comptez vos doigts en base 2. Quel est le lien entre l'écriture d'un nombre en base 2 et les restes des divisions successives d'un nombre par 2 ?

Proposez une procédure `bine(n)` qui renvoie la liste des chiffres composant l'écriture binaire d'un nombre. Vous utiliserez à bon escient les commandes XCAS `iquo(a,b)` et `irem(a,b)` qui renvoient respectivement le quotient et le reste de la division euclidienne de a par b.



« Retournez une liste »

Soit `L:=[1,2,3,4,5]` une liste. Regardez ce que donne

```
revlist(L)
```

Peut-être en ferez-vous bon usage...

c. Évaluation booléenne

Pour tester une égalité, on utilise `==`. Cela nous permet de faire des tests à l'aide de `if then else`. Par exemple :

```
pair(n):={
  if (irem(n,2)==0){"Pair"}else{"Impair"}
};

pair(45);
```

ou en « français »

```
pair(n):={
  si (irem(n,2)==0) alors "Pair" sinon "Impair"
  fsi;
};

pair(45);
```

d. Liste ou séquence ?

Il s'agit de deux objets informatiques différents. Une séquence est une succession ordonnée d'objets séparés par des virgules. Une liste est une séquence entre crochets. Dit comme ça, on ne voit pas trop quel est le problème. En fait, on peut effectuer de nombreuses opérations sur les listes qui ne sont pas possibles sur les séquences. D'un autre côté, il est plus simple de rajouter des éléments à une séquence qu'à une liste.

Nous allons utiliser le meilleur des deux selon nos besoins. Ainsi, pour rajouter un élément à une séquence, il suffit de faire :

```
S:=1,2,3,4;
S:=S,37;
```

alors que pour une liste il faut écrire :

```
L:=[1,2,3,4];
L:=append(L,37);
```

Une séquence vide se note `NULL`. Regardez alors ce que fait ce programme :

```
T=NULL;
pour k de 0 jusque 10 faire
  T:=T,k;
fpour;
```

ou en « anglais »

```
T=NULL;
for(k:=0;k<=10;k:=k+1){T:=T,k}
```

e. Coder en binaire

Écrivez « à la main » les nombres de 0 à 25 en binaire. Pourquoi Bobby a-t-il finement choisi le H ?

Écrivez « à la main » les nombres de 255 à 257 en binaire. À votre avis, quel est la base du binaire ?

Pourquoi est-il pratique d'écrire les nombres en base 2 avec un nombre de chiffres multiple de 4 avant de les traduire en binaire ?

À quoi pourrait servir cette ligne de code, sachant que `!=` signifie \neq :

```
while(size(ib) mod 4!=0){ib:=append(ib,0)};
```

ou en « français » :

```
...
tantque size(ib) mod 4!=0 faire ib:=append(ib,0) ftantque;
...
```

À l'aide d'une série de tests `if then else` proposez une procédure `bibine(n)` qui donne l'écriture bibinaire d'un nombre entré sous forme décimale.

Testez alors :

```
bibine(2751);
```

II - Décodez du bibinaire

a. La commande table

On construit une table avec la commande... `table` :

```
note:=table("maths"=18,"physique"=17,"info"=7]);
note["maths"];
note["info"];
```

À quoi correspond alors le code suivant :

```
decod:=table(seq(bibine(k)=k,k=0..15));
```

b. Procédure de décodage

Proposez à présent une procédure `enibib(L)` qui renvoie un nombre sous forme décimal correspondant à la liste L des lettres de son écriture bibinaire.

Vous utiliserez une seule boucle `for` connaissant la base du bibinaire :

```
enibib(L):={
  local N,k;
  N:=0;
  p:=0;
  pour k de size(L)-2 jusque 0 pas -2 faire
  ...
```

À vous d'imaginer la suite...