Rubrique itérative

A bas l'impérialisme de l'Itératif ! Place au récursif !!!

Il s'agit bien d'une fronde contre le titre qualificatif de cette rubrique, contre le titre de ce numéro contre tous ces algorithmes itératifs qui risquent de nous enfermer dans une seule vision des choses :

l'itération est une démarche constructive qui part du connu (données initiales) pour aller vers l'inconnu (le résultat que l'on cherche). D'une certaine manière, on est assez proche des raisonnements inductifs (il serait intéressant de bien distinguer l'une et l'autre démarche et leurs inter-actions dans quelques situations typiques qu'il faudrait produire).

En revanche la récursion part de l'inconnu pour aller au connu en réduisant le problème, c'est l'analyse récurrente qui consiste :

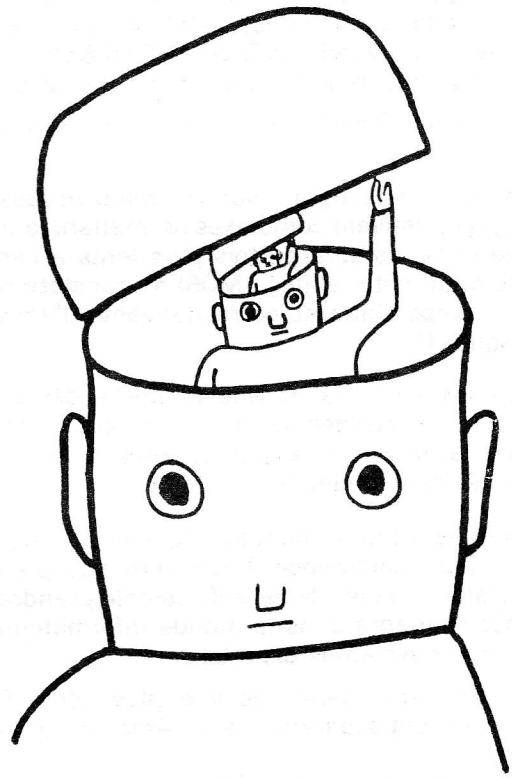
- à repérer l'information par son **type** et sa **taille**
- à la formaliser en montrant les aspects qualitatifs (type-propriété) et quantitatifs (taille),
- à faire apparaître un «découpage» de cette information (séquentialité) qui correspond à la réduction du problème en conservant le type et en réduisant la taille.

L'exemple classique de la factorielle permet de mettre en évidence le type : produit d'entiers consécutifs à partir de 1, et la taille n (pour n !). La séquentialité et la réduction du problème apparaissent avec l'écriture :

$$n ! = (n - 1) ! \times n$$

Connaître la factorielle de n revient donc à connaître la factorielle de n-1. La difficulté avec cette analyse récurrente (comme d'ailleurs avec la mise en évidence de la formule d'itération) est de bien concevoir ce passage comme une démarche générique qui engendre un processus qui ne prendra fin que lorsqu'on arrivera à quelque chose de connu (on rappelle qu'une démarche récursive conduit de l'inconnu au connu).

Pour cet exemple le connu peut être 1 ! (ce peut être aussi 10 ! si vous connaissez ce résultat et aussi les précédents car vous ne pourrez calculer n !, que pour $n > 10$).



On écrira donc la procédure récursive suivante :

RECURSIF

```
Fact : (naturel n) ⟶ (naturel)
|
|  choix :
|  | n = 1⟶1 ..... cas connu
|  | n ≠ 1⟶[Factoriel (n - 1)] * n   cas inconnu
|  Fin choix
|
Fin fact.
```

La démarche itérative conduit à l'algorithme suivant

ITERATIF

```
Factoriel (naturel n) → (naturel)
| F ← 1
| x ← 1
| TANT QUE (x < n) FAIRE (x ← x + 1 ; F ← F * x)
| FIN TANT QUE
| → F
Fin factoriel
```

On peut améliorer ce programme pour avoir une seule variable F.

Ici, il est plus important de noter que la forme itérative permet de bien mettre en évidence l'**algorithme** de calcul :

La lecture d'une écriture itérative est donc «proche» du traitement que l'on effectue (en particulier on peut facilement utiliser une calculette - programmable ou non -).

En revanche la forme **récursive** est plus nettement **déclarative** et il est plus difficile de lui associer «spontanément» un traitement. Dans ses travaux didactiques sur la récursivité André ROUCHIER pose le problème en ces termes : comment passer de l'analyse de la situation (la factorielle pour notre exemple) qui repose sur un traitement de forme **conjonctive** (pour avoir Fact... 10 il faut avoir la suite ordonnée des antécédents Fact 9 **et** Fact 8 **et** ... et Fact 1) à l'écriture du programme de forme **disjonctive** :

$n = 1$ (on donne Fact 1) ou $n \neq 1$ (on calcule avec Fact n-1)?

D'autre part, les programmes récursifs nécessitent des langages souvent complexes permettant le traitement de piles, ils sont parfois plus lents à l'emploi (avec du petit matériel (logo M05) on constate nettement un temps d'attente correspondant à l'empilage de n jusqu'à 1).

Pour ces raisons, certains pensent que les démarches itératives plus rapides d'emploi et plus faciles à comprendre (en particulier pour les élèves) sont mieux adaptées à l'enseignement.

Il convient de rester prudent alors que la didactique de l'informatique commence à produire ses premiers travaux, alors que le «déclaratif» semble prendre une place déterminante dans le monde informatique (de multiplan aux systèmes-experts).

D'autre part, d'un point de vue plus cognitif, les heuristiques font apparaître des intéractions constantes entre :
connu → inconnu : démarche ascendante,
inconnu → connu : démarche descendante

en particulier dans les démonstrations. Ces distinctions sont d'ailleurs souvent étrangères à la recherche des problèmes et n'apparaissent que dans la formulation (ou la formalisation) des solutions. On montre en particulier qu'il y a équivalence entre itération et récursivité avec appel récursif terminal (nous reviendrons sur ce problème on peut déjà se reporter à [1] ou [2], la distinction itératif-algorithmique / récursif-déclaratif demeure toutefois).

Enfin, signalons que la démarche récursive est moins étrangère aux enfants qu'on veut bien le croire (voir [3]) et qu'elle prend tout son sens quand on a à traiter des problèmes un peu complexes qui conduisent à un appel récursif non terminal (on met des actions en réserve dans une pile qui seront traitées par la suite au dépilage) ou qui nécessitent à plusieurs appels récursifs : on peut citer les constructions fractales dont voici un exemple (facilement programmable en logo)

```
POLYFRACTAL (nombre de côté N, côté L)    image fractale

   choix : L < 5 → rien
   |       L ⩾ 5 REPETE N [POLYFRACTAL (N ; L/2)
   FIN CHOIX                              AVANCE L
                                          TOURNEDROITE 360/N]
FIN POLYFRACTAL
```

Le nombre d'appels récursifs varie avec la longueur L du côté de départ. Le fait de prendre 5 dans le test est lié au matériel (définition de l'écran) à l'image que l'on veut obtenir.

Pour terminer on peut reprendre l'approximation de $\sqrt[n]{A}$ proposée dans l'article «De Héron à Newton», en constatant qu'une analyse récurrente du calcul ne convient pas, en effet pour réduire le problème (type racine, taille n) il est nécessaire d'avoir $n = a \times b$ pour écrire $\sqrt[n]{A} = \sqrt[a]{A} \times \sqrt[b]{A}$

(avec $1 < a < n$ et $1 < b < n$).

Il faut donc connaître $\sqrt[n]{A}$ pour tout n premier !

Cela ne disqualifie pas pour autant la démarche récursive et permet d'insister sur le caractère itératif de l'approximation, le titre de ce numéro est donc bien justifié, un pléonasme en quelque sorte.

C'est tout pour aujourd'hui.

M.C.

Référence [1] : LUCAS - PEYRIN - SCHOLL Algorithmique et traitement des données - MASSON ed.
Référence [2] : Démarches algorithmiques en classe de mathématiques IREM Orléans.
Référence [3] : Piaget. Construction des raisonnements récurrentiels PUF. éd.

■

Pour le savoir,
lisez **"Démarches algorithmiques en informatique"**
Pub n° 25 - Irem d'Orléans