Université Louis Pasteur
I R E M

Institut
de Recherche
sur l'Enseignement
des Mathématiques
10 rue du général Zimmer
67084 Strasbourg Cedex

LOGO

4. Récursivité :
Présentation et Analyse de situations

Claire Dupuis
Marie-Agnès Egret
Dominique Guin

Ce travail a été mené dans le cadre du Groupement de Recherches G0071 "Didactique et Acquisition des Connaissances Scientifiques", du Centre National de la Recherche Scientifique. Siège administratif : 46 rue Saint-Jacques 75005 Paris

1990

Avant - propos

Cette brochure contient la description et l'analyse d'une expérimentation d'enseignement de la récursivité menée avec des élèves de troisième. Ces élèves ont suivi en 4 ème un enseignement de base du langage LOGO orienté vers l'apprentissage de la programmation structurée dont la description et l'analyse figurent dans une autre publication de l' I.R.E.M. de Strasbourg (C. Dupuis , M.A. Egret , D. Guin , 1987).

Les enseignants pourront trouver, s'ils le souhaitent, des thèmes d'activités dans la partie " Présentation des différentes séances " (p. 7 à 46) avec leurs objectifs et nos commentaires. La partie " Analyse des productions des élèves " (p. 47 à 72) met en évidence les acquis et les difficultés des élèves au cours de cet apprentissage : ces éléments pourront permettre ultérieurement d'améliorer les situations proposées dans cette expérimentation.

Introduction

Cette étude s'intègre dans un ensemble de travaux de l'Institut de Recherche sur l'Enseignement des Mathématiques de Strasbourg liés à l'introduction de l'informatique dans le système scolaire. Elle a pour cadre le Groupement de Recherches "Didactique et Acquisition des Connaissances Scientifiques" (du Centre National de la Recherche Scientifique) qui coordonne le travail de plusieurs équipes françaises de recherche sur la Didactique des Mathématiques et de l'Informatique[1].

La méthodologie de cette expérimentation est analogue à celle développée dans l'expérimentation précédente, nous en rappelons succintement les idées principales[2]:

Il n'existe pas actuellement d'enseignement de l'informatique obligatoire au niveau secondaire. S'il n'y a pas d'enseignement, il n'y a pas de **difficultés** spécifiques connues, repérées, pas de **conceptions spontanées** identifiées. La mise au point d'un enseignement prenant en compte ces éléments nécessite une expérience d'enseignement et une analyse de **l'activité de programmation** des élèves, de leurs difficultés.

Un des résultats essentiels de la psychologie cognitive consiste à avoir mis en évidence le rôle éminent de la **résolution de problèmes** dans le processus d'acquisition des connaissances. Dans cette perspective, la connaissance **s'acquiert** et se **construit** par résolution de problèmes. C'est dans cet esprit que nous avons conçu nos séquences d'enseignement.

Le changement de l'enseignement des mathématiques, et de son contenu, à partir d'une **intervention** de **l'informatique**, nous semble nécessiter une étude sur les concepts **informatiques** de variable, de récursivité, de procédure, **en relation** avec les concepts **mathématiques** pour avoir une certaine maîtrise sur les **représentations** des objets mathématiques que les élèves se construiront. Si notre perspective est centrée sur l'informatique comme **"auxiliaire de pensée"**, nous devons étudier le **fonctionnement cognitif** dans ce domaine de connaissance.

---

1 Ont participé à ce groupe de travail : N. Balacheff, C. Laborde, P. Mendelsohn, J. Rogalski, A. Rouchier et R. Samurçay.

2 Pour plus de détails, il est possible de consulter (C. Dupuis, M.-A. Egret et D.Guin 1987, 1988 1 et 2).

L'étude de **l'acquisition des concepts élémentaires** propres à l'informatique et l'**analyse de leur enseignement** présentent donc une importance non seulement du point de vue de la connaissance didactique, mais du point de vue de la **pratique pédagogique.**

L'activité de programmation contraint les élèves à expliciter, par un programme écrit, leurs procédures de résolution. Cette explicitation reflète les connaissances acquises et les représentations qu'ont les élèves du fonctionnement du dispositif informatique. L'analyse des procédures de résolution implique donc la définition de **critères** d'analyse fondés, non seulement sur la ressemblance du résultat obtenu et du résultat demandé, mais encore sur les connaissances et les représentations sous-jacentes.

Nous avons donc poursuivi un triple objectif :

1 - mettre au point un **enseignement de la récursivité** accessible à des élèves de collège, basé sur la construction de situations-problème.

2 - proposer une **méthode d'analyse** par la définition de critères qui s'appuient sur une analyse a priori des situations et a posteriori des procédures de résolution. L'explicitation de méthodes d'analyse nous paraît suceptible d'améliorer la communication et la comparaison des recherches expérimentales en didactique de l'informatique. " Plus que les résultats de recherche, ce sont sans aucun doute les **méthodes** qui peuvent être les plus utiles pour les enseignants, non pas les méthodes de " preuves " des résultats mais plutôt les **méthodes d'analyse des situations** proposées aux élèves, à la fois lorsqu'il s'agit d'introduire des notions et quand il s'agit d'évaluer des acquisitions " (J. Rogalski , 1986).

3 - mettre en évidence les **compétences acquises** par les élèves, les **difficultés** rencontrées et les **procédures mises en oeuvre** par les élèves. Les résultats tirés de l'analyse permettront ensuite de modifier l'enseignement pour tenter de remédier aux difficultés apparues.

Conditions de l'expérimentation

L'expérimentation rapportée ici a été menée dans une classe de 3 ème composée de 19 élèves de 15 ans, avec la participation de l'enseignante de mathématiques.

Ces élèves ont tous suivi l'année précédente un enseignement de base du langage LOGO graphique et non graphique basé sur la programmation structurée, qui constituait la première phase de notre expérimentation.

Les 7 séances présentées ici constituent les deuxième et troisième phases de notre expérimentation. Rappelons notre projet (C. Dupuis, M.A. Egret, D. Guin, 1985, p. 44) :

- première phase :

Enseignement de base du langage LOGO.

A ce sujet, on pourra consulter la brochure LOGO 3 : programmation structurée (C. Dupuis, M.A. Egret, D. Guin, 1987) .

- deuxième phase :

Enseignement de la récursivité

Cette phase est celle qui doit donner aux élèves du sens à la récursivité. L'enseignement prévu a pour but de mettre la récursivité à leur disposition pour la réalisation de projets. Pour cela, nous estimons nécessaire de leur fournir différents modèles afin que chacun d'eux se construise une représentation de la récursivité.

- troisième phase :

Approfondissement

Il nous paraît important de ne pas nous restreindre à des projets graphiques, les élèves ayant reçu un enseignement de base du langage LOGO graphique et non

graphique. Cette phase qui n'avait pas eu lieu pendant la préexpérimentation nous semble essentielle pour évaluer l'acquisition de la récursivité.

Ces 7 séances se sont déroulées sur six mois environ. Elles ont, bien sûr, été de durée inégale. Les séances 2 à 5 constituent la deuxième phase. Nous avons choisi de faire précéder cette phase d'une séance (séance 1) motivant l'enseignement de la récursivité.

Cette alphabétisation informatique en Logo nous a permis, par la suite, de construire un enseignement des fonctions en mathématiques s'appuyant sur l'outil informatique (M.A. Egret, D. Guin, 1990).

La salle informatique comprenait 7 MICRAL 80 22 G et nous avons travaillé avec un langage LOGO qui ne connait que les nombres entiers (Matériel Education Nationale, dotation 1981) .

Le travail a eu lieu le samedi matin, à raison de deux heures par quinzaine. Bien que les heures d'informatique aient été des heures supplémentaires dans l'emploi du temps des élèves, ceux-ci étaient motivés pour participer à cette expérience une deuxième année.

Première partie : Présentation des différentes séances

Séance 1 : LES COURBES

Objectifs de la séance :

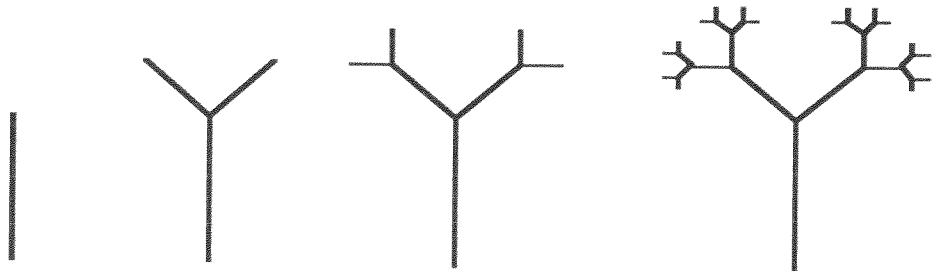
- situations - problèmes motivant l'introduction de la récursivité.
- travail sur l'invariance d'emboîtement. (cf. p. 8)

Consigne :

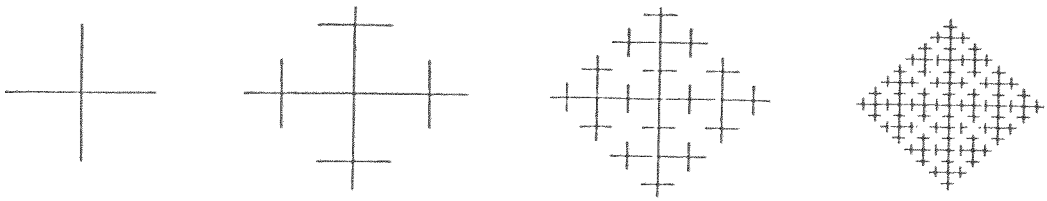
****

Voici trois familles de courbes. Sur une même ligne sont dessinées quatre courbes de la même famille. Choisissez une famille, puis écrivez les programmes correspondant à chaque courbe, en utilisant à chaque fois le programme précédent.

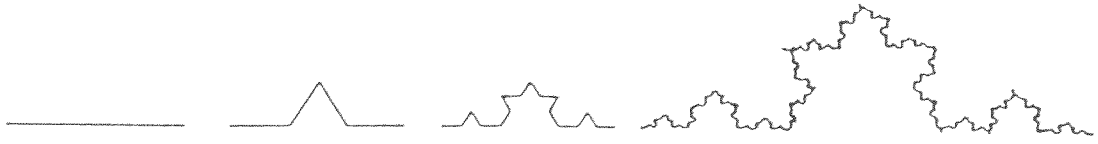
ARBRES :



CROIX :



VON KOCH :

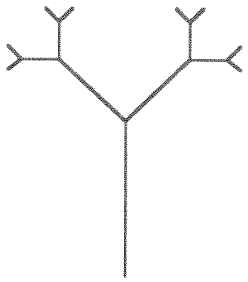


****

Analyse de la situation :

Ces courbes ont été choisies pour motiver l'introduction de l'outil récursivité : l'activité est complexe et comporte des difficultés de structuration et de coordination, points essentiels dans la première phase de notre expérimentation ; mais ces courbes plaisent esthétiquement aux élèves, ils ont d'autant plus d'ardeur au travail ! Bien entendu, l'écriture ne sera pas récursive, étant donné qu'ils n'ont eu, à ce moment, aucun enseignement sur la récursivité. Cependant, tous les ingrédients nécessaires à la récursivité sont présents dans ces situations, sauf **l'auto-référence**[1]. Chaque procédure peut faire appel au programme de la courbe précédente, toujours **de la même façon** : c'est ce que nous appelerons **invariance d'emboîtement**. Ce phénomène d'invariance est commun aux trois familles choisies. L'emboîtement de procédures différentes est le prolongement "naturel" des connaissances acquises en programmation structurée. Ce travail est donc consacré à la recherche de l'**invariance d'emboîtement** qui permettra l'écriture récursive.

Chaque famille de courbes comprend quatre étapes explicites qui ne correspondent pas forcément aux quatre premiers niveaux au sens récursif du terme[2]. La présentation des trois premiers niveaux et la consigne sont destinées à faire en sorte que les élèves structurent leurs programmes : le niveau inférieur étant écrit sous forme procédurale, il est économique de l'utiliser pour le niveau suivant. Le niveau 4 a été volontairement omis dans les familles des arbres et des courbes de Von Koch. Par exemple, dans la famille des arbres, le dessin de l'arbre **fantôme** qui a sa place entre la troisième et la quatrième étape explicitée est le suivant :



L'écriture d'une procédure de la courbe fantôme est utile dans une perspective de structuration et de régularité du programme ; elle est indispensable pour que soit réalisée l'invariance d'emboîtement d'une courbe à la suivante dans une même famille. Cette situation évite la difficulté liée à la **suspension de l'exécution** (cf. p. 21) tout en préparant à la représentation statique du programme, nécessaire à l'écriture de procédures

---

[1] il y a auto-référence lorsque la procédure s'appelle elle-même.

[2] Un niveau au sens récursif est le nombre d'exécutions de l 'appel récursif.

récursives : c'est un premier pas vers l'élaboration d'un schéma fonctionnel statique (J. Rogalski, G. Vergnaud, 1987) du type : pour construire l'objet de niveau n, on suppose construit l'objet de niveau n - 1.

Commentaires :

Le premier réflexe des élèves a été : " c'est beaucoup trop dur, on n'y arrivera jamais!" Cependant tous ont écrit au moins une procédure par étape et, suivant la consigne, ils ont tous réutilisé, à chaque étape, la procédure écrite à l'étape précédente. La programmation des courbes qui est une activité complexe les a beaucoup intéressés : tout au long de l'année scolaire, dès qu'ils en avaient l'occasion, ils ont continué cette activité.

Pour parvenir à écrire le dernier niveau de chaque famille, il était nécessaire de prendre en compte les régularités de la figure et de choisir un retour systématique de la tortue à une position "standard" à la fin de la procédure (C. Dupuis, M.- A. Egret, D. Guin, 1987) : cette nécessité n'a pas été perçue par tous et nombreux sont les élèves qui ont beaucoup tâtonné pour écrire le programme du dernier niveau.

a) La CROIX

Des élèves, ayant explicitement utilisé les régularités de La CROIX, ont proposé l'ensemble de procédures suivant :

```
POUR CROIX :R
REPETE 4 [ AV  :R   RE  :R   TD 90]
FIN

POUR CROIX1 :R
REPETE 4 [ AV :R*2   CROIX :R    RE :R*2   TD 90 ]
FIN

POUR CROIX2 :R
REPETE 4 [ AV :R*4    CROIX1 :R   RE :R*4   TD 90 ]
FIN

POUR CROIX3 :R
REPETE 4 [ AV :R*8    CROIX2 :R    RE :R*8   TD 90 ]
FIN
```

Cette solution ne respecte pas l'invariance d'emboîtement et ne permet donc pas le passage immédiat à une écriture récursive. C'est pourquoi nous avons proposé le corrigé suivant pour la famille de CROIX :

```
POUR CROIX1 : L
REPETE 4 [ AV  : L RE  : L TD 90 ]
FIN

POUR CROIX2  : L
REPETE 4 [ AV : L  CROIX1 : L / 2  RE  : L  TD 90 ]
FIN

POUR CROIX3  : L
REPETE 4 [ AV : L  CROIX2 : L / 2  RE  : L  TD 90 ]
FIN

POUR CROIX4 : L
REPETE 4 [ AV : L  CROIX3 : L / 2  RE  : L  TD 90 ]
FIN
```

Une discussion avec les élèves qui ont utilisé une stratégie multiplicative a été nécessaire pour les convaincre de l'équivalence des deux méthodes.

b) la COURBE DE VON KOCH

La mauvaise qualité graphique de la quatrième courbe de Von Koch empêche toute mesure d'angle ou de longueur. Il est donc nécessaire d'avoir admis que la courbe à l'écran doit être " semblable " aux précédentes. Pour écrire le dernier niveau de la courbe de VON KOCH, il faut trouver la courbe fantôme (C. Dupuis, M.- A. Egret, D. Guin, 1990), ce qui conduit à écrire l'ensemble de procédures suivant :

```
POUR VON1 : D
AV : D
FIN
```

```
POUR VON2: D
VON1 : D   TG 60   VON1 : D
TD 120
VON1 : D   TG 60   VON1 : D
FIN
```

```
POUR VON3: D
VON2 : D   TG 60   VON2 : D
TD 120
VON2 : D   TG 60   VON2 : D
FIN
```

POUR VON4: D
VON3 : D TG 60 VON3 : D
TD 120
VON3 : D TG 60 VON3 : D
FIN

Ce programme est celui de la courbe fantôme.

POUR VON5: D
VON4 : D TG 60 VON4 : D
TD 120
VON4 : D TG 60 VON4 : D
FIN

c) les ARBRES

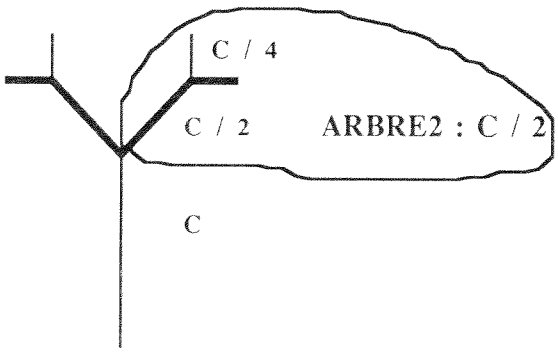
Le programme d'un arbre quelconque s'écrit, en y appelant **toujours de la même façon** le programme de l'arbre précédent :

POUR ARBRE (i+1) : C
AV : C
TG 45
ARBRE (i) : C / 2
TD 90
ARBRE (i) : C / 2
TG 45
RE : C
FIN

POUR ARBRE1 :C
AV :C
RE :C
FIN

A titre d'exemple, voici le programme de l'arbre de niveau 3 :

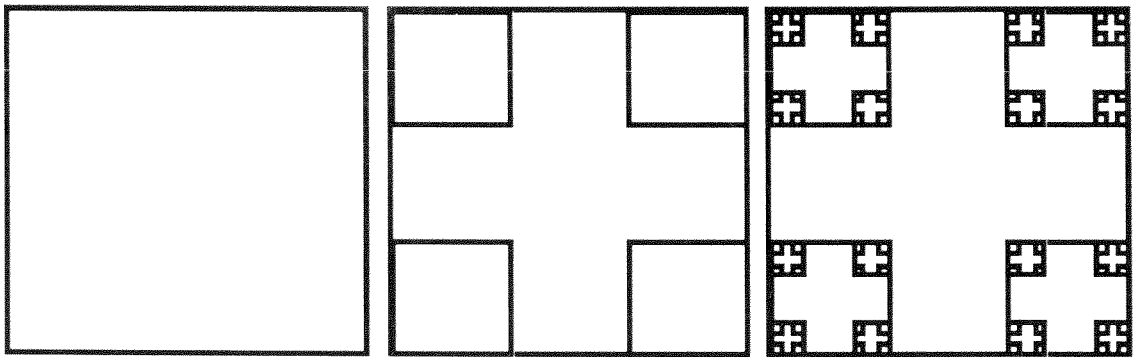
POUR ARBRE3 : C
AV : C
TG 45
ARBRE2 : C / 2
TD 90
ARBRE2 : C / 2
TG 45
RE : C
FIN



Une particularité de cette famille a rajouté une difficulté supplémentaire : la présence de deux, trois puis cinq longueurs différentes dans le même arbre. Les élèves préférant, nous semble-t-il, ne pas expliciter la relation entre ces longueurs, ont introduit différentes variables. L'analyse des difficultés et des stratégies des élèves est détaillée dans (C. Dupuis, D. Guin, 1989).

d) Les CARRES

Nous avons décidé de proposer une nouvelle famille de courbes en explicitant oralement une relation fonctionnelle. La détermination de la relation n'était donc plus à la charge de l'élève. Par contre, l'expression de cette relation en termes de variables restait à faire. Nous avons proposé la famille de carrés suivante :



(étape 1) (étape 2) (étape 3)

avec la même consigne que précédemment, plus l'indication :
" Chaque carré a pour côté le tiers du côté du carré qui l'entoure."

On remarque que les trois étapes explicitées correspondent aux niveaux 1, 2 et 4, au sens récursif du terme. Il y a donc un carré fantôme.

Le programme du premier carré s'écrit :

```
POUR CARRE1 : C
REPETE 4 [ AV : C   TD 90]
FIN
```

Le programme d'un carré quelconque peut s'écrire ainsi :

```
POUR CARRE ( i + 1) : C
REPETE 4 [ CARRE ( i ) : C / 3   AV : C   TD 90]
FIN
```

Séance 2 : ESSAIS

Objectif de la séance:

- introduction de la récursivité : manipulation de procédures récursives.

Consigne :

****

Voici un exemple de schéma récursif :

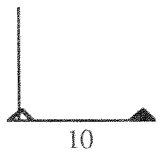
POUR ESSAI :N
SI :N = 0 ALORS [STOP]
SPA
ESSAI :N-10
SPB
FIN

Remarque : N est supposé multiple de 10.
SPA, SPB sont des sous-procédures ou de simples instructions.

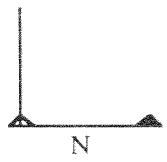
SPA et SPB seront modifiées à chaque exercice. Ecrivez, avant chaque exercice, votre prévision de l'exécution de cette procédure. Testez ensuite le programme sur l'ordinateur et essayez de comprendre les résultats.

DESSIN de A

△ position initiale
▲ position finale



DESSIN de B
(longueur de chaque branche : N)



Exercices :

1- SPA : dessin de A
SPB : rien

2- SPA : rien
SPB : dessin de A

3- SPA : dessin de A
SPB : dessin de A

4- SPA : dessin de B
SPB : rien

5- SPA : rien
SPB : dessin de B

6- SPA : dessin de B
SPB : dessin de B

7- SPA : instruction ECRIS :N
SPB : rien

8- SPA : rien
SPB : instruction ECRIS :N

9- SPA : instruction ECRIS :N
SPB : instruction ECRIS :N

10- SPA: dessin de B
SPB: instruction ECRIS :N

11- SPA: dessin de B,
instruction ECRIS :N
SPB: rien

D'autres exemples de schéma récursif...

* On modifie l'appel de procédure dans la procédure récursive ESSAI. On remplace l'instruction ESSAI :N - 10 par ESSAI :N - 1. Réalisez des essais de ce schéma dans les cas suivants :

12 - SPA : instruction ECRIS ITEM :N :L
SPB : rien
où L est la phrase : [UN DEUX TROIS QUATRE]

Question : Comment doit-on choisir N pour que cette procédure tourne bien ?

13 - SPA : (ECRIS :N ITEM :N :L)
SPB : Idem

14 - SPA : instruction ECRIS LETTRE :N :MOT
SPB : instruction ECRIS :N

où LETTRE est la procédure suivante :

```
POUR LETTRE :N :M
SI :N =1  ALORS [RENDS PREM :M ]
          SINON [RENDS LETTRE :N -1 SP :N ]
FIN
```

* On remplace l'instruction ESSAI :N - 1 par ESSAI :N + 1. Réalisez des essais de ce schéma dans les cas suivants :

15 - SPA : ECRIS :N
SPB : rien

Question : *Comment doit-on choisir N pour que cette procédure tourne bien ?*

16 - SPA : ECRIS :N
SPB : ECRIS :N

* On remplace l'appel de procédure ESSAI :N-10 par les deux instructions :

```
RELIE "N :N+10  ESSAI :N
```

17 - SPA : ECRIS :N
SPB : ECRIS :N

Question : Quelle est la différence avec l'exercice 9 ?

****

Corrigé :

Les "listings" qui suivent ont été donnés aux élèves après qu'ils aient effectué leurs prévisions et réalisé eux-mêmes les exercices.

```
POUR DESSIN
AV 10  RE 10  TD 90  AV 10  TG 90
FIN
```

```
POUR EXERCICE1 :N
SI :N = 0 ALORS [STOP]
DESSIN
EXERCICE1 :N - 10
FIN
```

N = 40

```
POUR EXERCICE2 :N
SI :N = 0 ALORS [STOP]
EXERCICE2 :N - 10
DESSIN
FIN
```

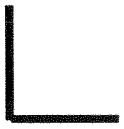
N = 40

```
POUR EXERCICE3 :N
SI :N = 0 ALORS [STOP]
DESSIN
EXERCICE3 :N - 10
DESSIN
FIN
```

N = 40

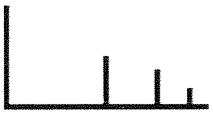
```
POUR DESSINVAR :L
AV :L RE :L TD 90 AV :L TG 90
FIN
```

L = 30



```
POUR EXERCICE4 :N
SI :N = 0 ALORS [STOP]
DESSINVAR :N
EXERCICE4 :N - 10
FIN
```

N = 40

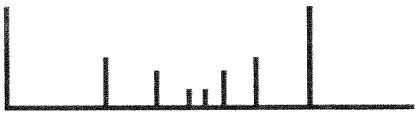


```
POUR EXERCICE5 :N
SI :N = 0 ALORS [STOP]
EXERCICE5 :N - 10
DESSINVAR :N
FIN
```

N = 40

```
POUR EXERCICE6 :N
SI :N = 0 ALORS [STOP]
DESSINVAR :N
EXERCICE6 :N - 10
DESSINVAR :N
FIN
```

N = 40



```
POUR EXERCICE7 :N
SI :N = 0 ALORS [STOP]
ECRIS :N
EXERCICE7 :N - 10
FIN
```

EXERCICE7 40
40
30
20
10

```
POUR EXERCICE8 :N
SI :N = 0 ALORS [STOP]
EXERCICE8 :N - 10
ECRIS :N
FIN
```

EXERCICE8 40
10
20
30
40

```
POUR EXERCICE9 :N
SI :N = 0 ALORS [STOP]
ECRIS :N
EXERCICE9 :N - 10
ECRIS :N
FIN
```

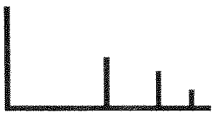
EXERCICE9 40
40
30
20
10
10
20
30
40

```
POUR EXERCICE10 :N
SI :N = 0 ALORS [STOP]
DESSINVAR :N
EXERCICE10 :N - 10
ECRIS :N
FIN
```

N = 40

10
20
30
40

N = 40



```
POUR EXERCICE11  :N
SI :N = 0 ALORS [STOP]
DESSINVAR :N  ECRIS :N
EXERCICE11 :N - 10
FIN
```

```
40
30
20
10
```

```
EXERCICE12 4 [UN DEUX TROIS QUATRE]
QUATRE
TROIS
DEUX
UN
```

```
POUR EXERCICE12 :N :L
SI :N = 0 ALORS [STOP]
ECRIS ITEM :N :L
EXERCICE12 :N - 1 :L
FIN
```

```
EXERCICE12 2 [UN DEUX TROIS QUATRE]
DEUX
UN
```

```
POUR LETTRE :N :M
TESTE :N = 1
SIVRAI [RENDS PREM :M]
SIFAUX [RENDS LETTRE :N - 1 SP :M]
FIN
```

```
ECRIS LETTRE 2 "NICOLE

I
```

```
POUR EXERCICE13 :N :L
SI :N = 0 ALORS [STOP]
(ECRIS :N  ITEM :N :L)
EXERCICE13 :N - 1  :L
(ECRIS :N  ITEM :N :L)
FIN
```

```
EXERCICE13  3  [UN DEUX TROIS ]
3 TROIS
2 DEUX
1 UN
1 UN
2 DEUX
3 TROIS
```

```
POUR EXERCICE14 :N :MOT
SI :N = 0 ALORS [STOP]
ECRIS LETTRE :N :MOT
EXERCICE14 :N - 1 :MOT
ECRIS :N
FIN
```

```
EXERCICE14 4 "NOIX
X
I
O
N
1
2
3
4
```

```
POUR EXERCICE15 :N
SI :N = 0 ALORS [STOP]
ECRIS :N
RELIE "N  :N - 10
EXERCICE15 :N
ECRIS :N
FIN
```

```
EXERCICE15 40
40
30
20
10
0
10
20
30
```

```
POUR EXERCICE16 :N
SI :N = 0 ALORS [STOP]
ECRIS :N
EXERCICE16 :N + 1
FIN
```

```
EXERCICE16 -4
- 4
- 3
- 2
- 1
```

```
POUR EXERCICE17 :N
SI :N = 0 ALORS [STOP]
ECRIS :N
EXERCICE17 :N + 1
ECRIS :N
FIN
```

```
EXERCICE17 -2
- 2
- 1
- 1
- 2
```

Analyse de la situation :

Nous avons fourni aux élèves le schéma récursif ESSAI afin qu'ils manipulent cette procédure en modifiant de manière systématique les sous-procédures SPA et SPB. La récursivité terminale correspond au cas où SPB est vide.

La conception spontanée de l'auto-référence, qui est le "retour au début du programme", correspond parfaitement au GO TO du BASIC et n'est pas contredite lors d'un appel terminal. Ce modèle s'apparente au modèle de fonctionnement spontané de l'itération " faire une suite d'actions et recommencer ", que nous avons vu fonctionner au cours de la première phase d'enseignement : en effet, lorsque les élèves avaient des projets graphiques à réaliser qui consistaient à répéter n fois une certaine figure, avec un programme d'interface entre chaque répétition, nous avons observé des solutions consistant à écrire la liste des instructions à répéter, puis l'instruction REPETE n-1 fois.

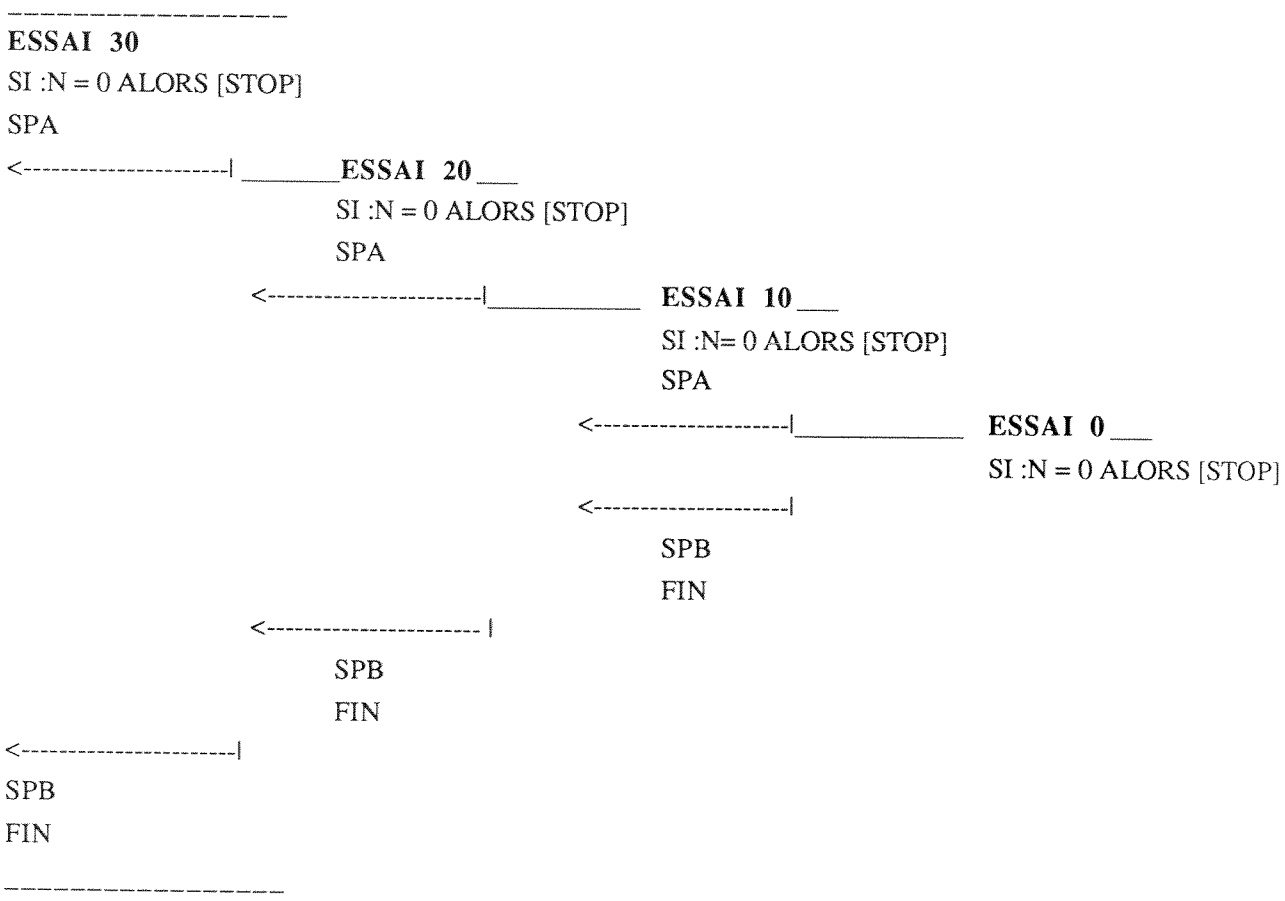
Cette conception spontanée de "retour au début du programme" est incompatible avec l'appel non terminal. Dans les quatre premiers exercices, cette incompatibilité ne peut pas être perçue ; l'exécution du cinquième exercice contredit de manière évidente cette conception spontanée. Nous créons ainsi un conflit qui remet en cause la conception spontanée des élèves et nous leur proposons alors d'autres modèles de fonctionnement.

Ce modèle d'apprentissage est fondé sur le développement d'une interaction entre deux fonctions essentielles : la compréhension du problème et l'élaboration d'une solution. D'où le concept de "système de représentation et de traitement" pour exprimer la liaison indissociable entre les **représentations** et les **traitements** : les élèves se construiront un système de représentation et de traitement par la confrontation entre leurs prévisions et les résultats produits lors de variations systématiques des types d'appels.

Commentaires :

Les élèves ont manipulé cette procédure ESSAI pour différentes sous-procédures SPA et SPB avec des variations systématiques (ce sont les exercices 1 à 11). Comme notre préexpérimentation (C. Dupuis , M.A. Egret , D. Guin , 1985) l'avait montré, les quatre premiers exercices n'ont pas posé de difficulté puisqu'une représentation exacte du fonctionnement de la récursivité n'est pas nécessaire pour faire des prévisions d'exécutions correctes jusque-là. C'est l'exercice 5 qui a provoqué la surprise : aucun élève n'est parvenu à comprendre pourquoi la taille du dessin de B augmentait. Nous avons alors proposé aux élèves différents modèles de représentation du fonctionnement des procédures récursives afin qu'ils s'approprient celui qui leur permettrait le mieux de comprendre le phénomène ; en voici deux exemples :

Le modèle d'insertion de lignes :



Ce modèle d'insertion de lignes met en évidence la **suspension de l'exécution** de la procédure appelante jusqu'à la fin de l'exécution de la procédure appelée.

Un schéma qui simule l'exécution des instructions :

Pour la lecture du schéma, on suit les flèches ; sinon, on lit les instructions de gauche à droite. Les chiffres soulignés indiquent l'ordre d'affichage des instructions.

```
POUR ENTIERA :N
SI :N = 0 ALORS [STOP]
ECRIS :N
ENTIERA :N+1
FIN
```

ENTIERA -2

NIVEAU	N	TEST	ECRIS :N	ENTIERA :N + 1
0	- 2	CONT	ECRIS -2 (1)	ENTIERA _1
1	- 1	CONT	ECRIS -1 (2)	ENTIERA 0
2	0	STOP		

```
POUR ENTIERB :N
SI :N = 0 ALORS [ STOP]
ENTIERB :N+1
ECRIS :N
FIN
```

ENTIERB -2

NIVEAU	N	TEST	ENTIERB :N+1	ECRIS :N
0	- 2	CONT	ENTIERB -1	ECRIS -2 (2)
1	- 1	CONT	ENTIERB 0	ECRIS -1 (1)
2	0	STOP		

Ce schéma permet l'introduction de la notion de **niveau** (cf. p. 8).

Dans les exercices 12 à 17, nous avons repris un travail analogue en modifiant l'appel de procédure et en utilisant pour SPA et SPB des instructions non graphiques. Les élèves ont du choisir la valeur de N pour l'exécution. Ils ont donc été amenés à prévoir le fonctionnement de la procédure, de manière à en contrôler l'arrêt.

Remarquons que dans l'ensemble, à l'issue de cette phase, les prévisions sont correctes et que le maniement des primitives texte ne paraît pas poser de problèmes supplémentaires (il est moins aisé, car moins bien connu). Certains groupes font des erreurs jusqu'à l'exercice 13, mais toutes les prévisions sont correctes à partir de là, avec des commentaires du style "ouf! on a enfin compris". Ces élèves n'ont fait une analyse du fonctionnement interne de la procédure qu' **après** une accumulation d'échecs. Auparavant, les commentaires sont souvent des descriptions des réalisations : " 5, c'est l' exercice inverse de 4 " ; "6, c'est 4 puis 5 ";" 8 est le contraire de 7 ".

Séance 3 : LES COURBES ET LA RECURSIVITE

Objectif de la séance:

- réinvestissement de la récursivité.

Consigne :

****

Ecrivez récursivement le programme CROIX afin d'obtenir à l'éxécution une courbe quelconque de la famille.

****

Commentaires :

Après quelques minutes de réflexion, une bonne partie des élèves arrive à écrire le programme suivant (rappelons que nous leur avions fourni un corrigé pour la famille de CROIX , page 10) :

```
POUR CROIX : N : L
REPETE 4 [ AV : L CROIX : N : L / 2 RE : L TD 90 ]
FIN
```

Les élèves font exécuter le programme. La présence de la variable N traduit la volonté d'exprimer le type de courbe : CROIX2, CROIX3. Ils constatent qu'il manque un test d'arrêt : ils proposent de rajouter une condition sur la longueur de L que l'on précisera par la suite. Nous leur conseillons de tester leur programme pour plusieurs valeurs de L et nous écrivons au tableau :

```
POUR CROIX : L
TEST
REPETE 4 [ AV : L CROIX : L / 2 RE : L TD 90 ]
FIN
```

Le test le plus fréquemment proposé par les élèves est le test d'arrêt de ESSAI :
SI :L = O ALORS [STOP]. Il y a une discussion avec les élèves sur les différents types de test :

- Pourquoi sur un appareil qui connait les décimaux, ce test ne convient-il pas ?
- Et si l'on remplace L = 0 par L = 1 ?

- Pourrait-on utiliser une inégalité ? Laquelle ?

La conclusion de la discussion est que, si l'on veut travailler avec d'autres nombres que des entiers, il vaut mieux utiliser une inégalité, d'où le corrigé :

POUR CROIX : L
SI : L < 2 ALORS [STOP]
REPETE 4 [AV : L CROIX : L / 2 RE : L TD 90]
FIN

Une nouvelle discussion commence dans la classe :

- Comment doit-on choisir L (en MICRAL, uniquement des quotients entiers!) pour que le test fonctionne bien ?

- Au premier appel L doit être divisible par 2, au second encore par 2, donc divisible par 4, etc... L doit être une puissance de 2.

- Comment pourrait-on choisir autrement que par la longueur de L, le type de courbe qu'on veut obtenir ?

- Il faut introduire le niveau, mais comment ?

Nous cherchons ensemble et choisissons le niveau 4 (instruction RELIE " NIVEAU 4)

- Il faut un compteur, un ascenseur qui descend jusqu'au niveau choisi, par exemple, N.

- De quel étage doit partir l'ascenseur ?

Les élèves proposent que l'ascenseur parte de 0 : l'éxécution se fera à partir de 0.

POUR CROIX : L : N
SI : N = : NIVEAU ALORS [STOP]
REPETE 4 [AV : L CROIX : L / 2 : N+ 1 RE : L TD 90]
FIN

- Peut - on prendre n'importe quelle valeur de L, une fois le niveau choisi ?

- Non.

- Par exemple, si on veut un niveau 3, comment choisir L ?

- L devra être divisible par 2 et encore par 2 encore par 2 : divisible par 8!

- Alors, si L = 2^{11} quelle est la longueur de la plus petite branche pour la courbe de niveau 3 ?

- Elle sera égale à $2^{11} / 2^3$.

- Mais puisque L = 2^n, comment exprimer le test d'arrêt ?

- Il faudra choisir n supérieur ou égal au niveau.

Remarquons que les difficultés durant cette discussion sont davantage d'ordre mathématique qu'informatique!

Séance 4 : PROJETS

Objectif de la séance:

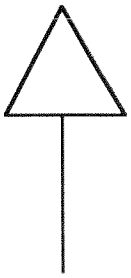
- écriture de procédures récursives (récursivité linéaire[1]).

Consigne :

****

Voici des exécutions de procédures récursives. Ecrivez, dans chaque cas, le programme récursif correspondant [2] .

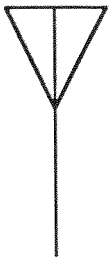
MOTIFA 40



```
POUR MOTIFA :L
AV :L
TRIANGLE
RE :L
FIN

POUR TRIANGLE
TG 90 AV 10
REPETE 2 [TD 120 AV 20]
TD 120 AV 10 TD 90
FIN
```

MOTIFB 40



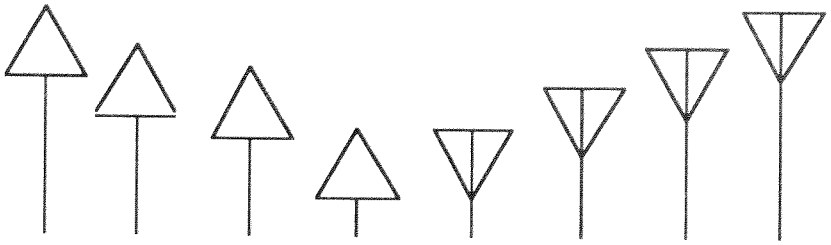
```
POUR MOTIFB :L
AV :L
BIDULE
RE :L
FIN

POUR BIDULE
TG 30 AV 20
REPETE 2 [TD 120 AV 20]
TD 150 AV 17 RE 17
FIN
```

[1] récursivité avec un seul appel dans la procédure.

[2] le corrigé proposé n'a, bien entendu, pas été distribué

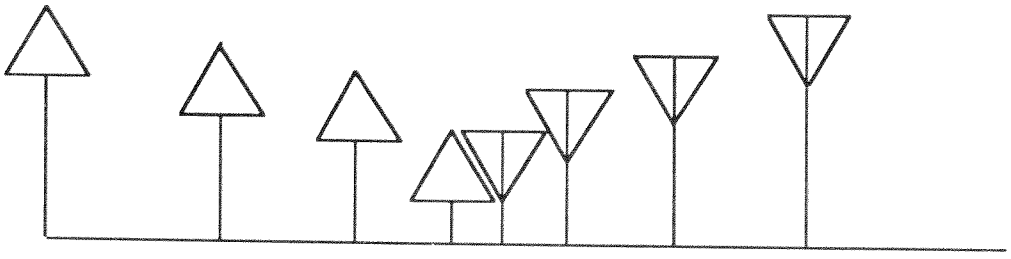
PROJ1 40



POUR ECART
TD 90 LC AV 20 TG 90 BC
FIN

POUR PROJ1 :L
SI :L = 0 ALORS [STOP]
MOTIFA :L ECART
PROJ1 :L - 10
MOTIFB :L ECART
FIN

PROJ2 40



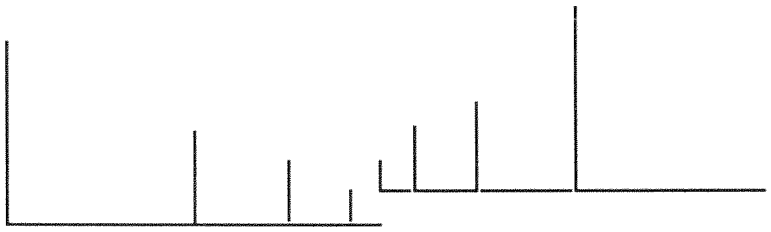
POUR POSI :L
TD 90 AV :L TG 90
FIN

POUR PROJ2 :L
SI :L = 0 ALORS [STOP]
MOTIFA :L POSI :L
PROJ2 :L - 10
MOTIFB :L POSI :L
FIN

```
PROJ3 4 [UN DEUX TROIS QUATRE]
        4 Q
        3 T
        2 D
        1 U
        UUN
        DDEUX
        TTROIS
        QQUATRE
```

```
POUR PROJ3 :N :L
SI :N = 0 ALORS [STOP]
(ECRIS :N  PREM ITEM :N :L)
PROJ3 :N - 1 :L
(ECRIS MOT  PREM ITEM :N :L  ITEM :N :L)
FIN
```

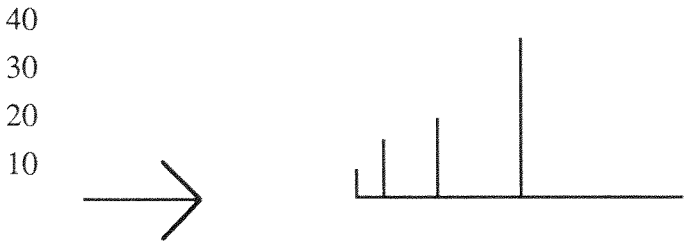
PROJ4 40



```
POUR DESSINVAR :L
AV :L  RE :L
TD 90
AV :L  TG 90
FIN
```

```
POUR PROJ4 :L
SI :L = 0 ALORS [LC  AV 10  BC  STOP]
DESSINVAR :L
PROJ4 :L - 10
DESSINVAR :L
FIN
```

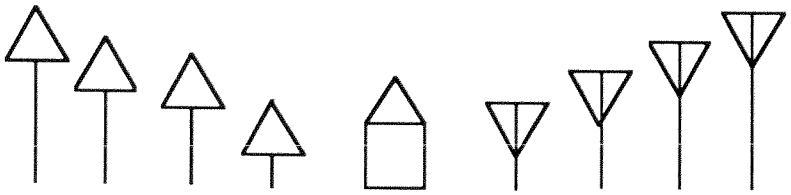
PROJ5 40



```
POUR FLECHE
TD 90  AV 20
REPETE 2 [TG 120  AV 5 RE 5]
TD 60  LC  RE 20  BC  TD 90
FIN
```

```
POUR PROJ5 :N
SI :N = 0 ALORS [FLECHE STOP]
ECRIS :N
PROJ5 :N - 10
DESSINVAR :N
FIN
```

PROJ6 40



```
POUR TOIT
AV 14  TD 90 AV 14
FIN

POUR FACADE
REPETE 4 [AV 20  TD 90]
FIN

POUR POSITI
AV 20  TD 45
FIN

POUR POSIT
TG 135  RE 20
FIN
```

```
POUR MAISON
FACADE
POSITI
TOIT
POSIT
FIN

POUR ECART
TD 90  LC  AV 20  TG 90  BC
FIN

POUR PROJ6 :L
SI :L = 0 ALORS  [MAISON  STOP]
MOTIFA :L  ECART
PROJ6 :L - 10
ECART  MOTIFB :L
FIN
```

```
PROJ7  4  [UN DEUX TROIS QUATRE]
QUATRE
TROIS
DEUX
UN
TU BEGAIES :
UUN
DDEUX
TTROIS
QQUATRE
```

```
POUR PROJ7 :N :L
SI :N = 0 ALORS  [ECRIS [TU BEGAIES :] STOP]
ECRIS ITEM :N :L
PROJ7 :N - 1 :L
(ECRIS MOT PREM ITEM :N :L ITEM :N :L)
FIN
```

****

Analyse de la situation:

Jusqu'à cette séance, les élèves ont manipulé la récursivité de deux manières. Ils ont exécuté des procédures récursives fournies (séance 2) : la seule tâche demandée était une prévision du résultat ; ils ont transformé les programmes des courbes en programmes récursifs (séance 3). Nous voulions dans cette séance observer si la compréhension du phénomène était suffisante pour qu'ils puissent eux-mêmes écrire des procédures récursives. Nous leur avons donc fourni sept de ces projets qui correspondent aux réalisations de programmes récursifs à récursivité centrale, graphiques ou non graphiques, incluant parfois une procédure avant le STOP. Les programmes de ces projets sont sans doute plus faciles à écrire que ceux des courbes : cependant, les élèves doivent aussi découvrir de nouvelles formes de test d'arrêt.

Commentaires :

Les élèves ont moins aimé cette séance que la précédente : les résultats sont moins spectaculaires. Certains ne respectent pas la consigne d'écrire un seul programme récursif par projet et réalisent des projets en écrivant deux programmes qu'ils ajustent ensemble ensuite. La possibilité d'écrire une procédure avant le STOP dans le test d'arrêt n'a pas posé de grandes difficultés. Tous les groupes ont écrit au moins un programme correct, certains très rapidement.

Séance 5 : TESTS

Objectif de la séance:

- évaluation des séances précédentes .

Consigne :

****

Test 1

1. Donnez une prévision d'exécutions des procédures récursives fournies.

1 - ***TOTO 9*** avec :

```
POUR TOTO :N
SI :N = 1 ALORS [ STOP ]
ECRIS :N
TOTO :N-2
ECRIS :N-1
FIN
```

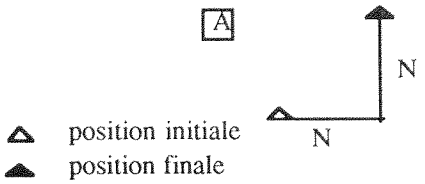
2 - ***ESSAI 30*** avec :

SPA : rien
SPB : Dessin A

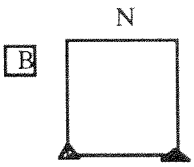
rappel :

```
POUR ESSAI :
SI :N = 0 ALORS [STOP]
SPA
ESSAI :N-10
SPB
FIN
```

Dessin A



Dessin B



3 - ***ESSAI 30*** avec : **SPA** : Dessin B
SPB : rien

4 - ***ESSAI 30*** avec : **SPA** : Dessin A
SPB : Dessin B

5 - ***TOTI "BON*** avec : **SPB** : ECRIS PREM :MOT

```
POUR TOTI :MOT
SI VIDE? :MOT ALORS [STOP]
ECRIS PREM :MOT
TOTI SP :MOT
SPB
FIN
```

6 - ***TOTI "BON*** avec : **SPB** : ECRIS DER :MOT

7 - ***TATI 3***, puis ***TATI 5*** , puis ***TATI 7***

```
POUR TATI :N
SI :N =1 ALORS [STOP]
TATI :N-2
ECRIS :N
TATI :N-2
FIN
```

2. Complétions de procédures récursives.

On donne :
- une procédure récursive

- une exécution de cette procédure

Il manque une instruction dans la procédure pour que l'exécution soit celle qui vous est donnée ; indiquez l'instruction manquante et sa place dans la procédure.

Procédure	***Exécution***
POUR TOTO :PHRASE	TOTO [UN DEUX TROIS]
SI VIDE? :PHRASE ALORS [STOP]	UN
ECRIS PREM :PHRASE	DEUX
TOTO SP :PHRASE	TROIS
ECRIS PREM :PHRASE	TROIS 1
FIN	DEUX 2
	UN 3

Utilisez la primitive COMPTE :
exemple: ECRIS [Le chat] COMPTE [La pluie tombe]
Le chat 3

Corrigé de la partie 2.

Il faut ajouter COMPTE :PHRASE à l'avant-dernière ligne, après
ECRIS PREM :PHRASE

Test 2

1. - ***TOTI "SAC*** avec: **SPA** : ECRIS PREM :MOT
SPB : rien

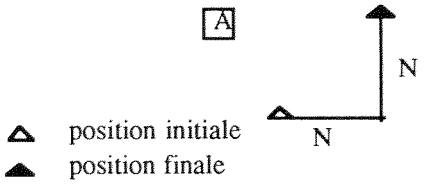
POUR TOTI :MOT
SI VIDE? :MOT ALORS [STOP]
SPA
TOTI SD :MOT
SPB
FIN

2. - ***TOTI "SAC*** avec: **SPA** : rien
SPB : ECRIS DER :MOT

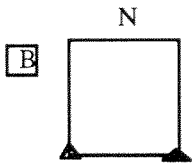
3. - ***TOTI "SAC*** avec: **SPA** : ECRIS DER :MOT
SPB : ECRIS PREM :MOT

4 - ***ESSAI 30*** avec : **SPA** : Dessin A
SPB : Dessin B

Dessin A



Dessin B



rappel :

```
POUR ESSAI :N
SI :N =0 ALORS [STOP]
SPA
ESSAI :N-10
SPB
FIN
```

5 - ***TOTO 8*** avec :

```
POUR TOTO :N
SI :N = 0 ALORS [ STOP ]
ECRIS :N - 1
TOTO :N - 2
ECRIS :N
FIN
```

6 - ***TATI 4*** , puis ***TATI 6*** , puis ***TATI 8***

```
POUR TATI :N
SI :N =2 ALORS [STOP]
TATI :N - 2
ECRIS :N
TATI :N - 2
FIN
```

****

Analyse de la situation:

Le but de ce test est de repérer les conceptions que les élèves se sont construites du **fonctionnement d'une procédure récursive**, plus particulièrement lorsque la récursivité est **centrale**. Pour cela, nous avons choisi de faire passer un test individuel papier-crayon (donc sans accès à l'ordinateur) en demandant pour chaque exercice, une prévision d'exécution de la procédure fournie. Contrairement à ce qui peut se produire lorsque l'on demande l'écriture de procédures récursives, il n'est pas possible de contourner les difficultés liées au fonctionnement du dispositif informatique.

Nous avons choisi de ne pas nous limiter à des situations où la réussite est possible avec une règle d'action globale (lorsqu'il s'agit de procédures récursives avec appel central) de la forme : "avant l'appel, ça décroit et après l'appel, ça croit". Dans de telles situations, les conceptions des élèves sont difficilement identifiables. Cette règle d'action globale était suffisante pour écrire bon nombre des procédures récursives que nous avions demandées dans les "projets".

Les énoncés sont construits suivant deux variations du schéma ESSAI :

a) Type de récursivité :

récursivité terminale, c'est à dire absence de SPB et absence d'opérations sur l'appel
récursivité avec appel au début, c'est à dire absence de SPA
récursivité centrale, c'est à dire présence de SPA et SPB.

b) Type de sous-procédures SPA et SPB :

écriture de valeurs de variables
graphique (dessin dont les dimensions varient en fonction du niveau)
liste (écriture de lettres d'un mot)

Pour permettre une meilleure observation, les énoncés choisis ne combinent pas les difficultés. SPA et SPB sont toujours du même type dans ces tests, sans être identiques.

Commentaires :

Ce test s'est déroulé en deux séances, espacées d'une semaine, sans corrigé entre les deux séances. Les élèves étaient invités à consulter leurs cahiers d'informatique contenant les fiches de cours et les programmes qu'ils avaient réalisés. Ils ont été un peu déroutés lors de la première séance par la forme des questions posées. Nous avons constaté que la première séance a eu un effet d'entraînement à ce type d'activité et que certains types de prévisions fausses ont disparu complètement lors de la deuxième séance. Des travaux d'élèves sont analysés dans la deuxième partie de la brochure (cf. p. 63).

Séance 6 : RENDS

Objectif de la séance:

- instruction RENDS

La séance commence par des exemples de procédures utilisant RENDS que l'on teste et discute avec toute la classe.

Exemples :

1 - POUR DOUBLE :X
RENDS 2 * :X
FIN

ECRIS DOUBLE 4
8

L'instruction RENDS produit un résultat qui peut être un nombre, un mot, ou une liste :

2 - POUR LETTRE2 :MT
RENDS PREM SP :MT
FIN

ECRIS LETTRE2 "BONJOUR
O

3 - POUR LISTOUMOT :X
SI MOT? :X ALORS [RENDS " MOT]
SINON [RENDS " LISTE]
FIN

ECRIS LISTOUMOT [LE CHAT]
LISTE

L'instruction RENDS peut aussi produire le résultat VRAI ou FAUX :

4 - POUR PLONG? :MOT1 :MOT2
SI (COMPTE :MOT1) > (COMPTE :MOT2) ALORS [RENDS "VRAI]
SINON [RENDS "FAUX]
FIN

ECRIS PLONG? "BON "MAUVAIS
FAUX

SI PLONG? 23 2 ALORS [ECRIS "NATURELLEMENT!]
NATURELLEMENT!

****

Exercices sur l'instruction RENDS :

1 - Prévoyez une exécution de chaque procédure, puis écrivez celle-ci :

1 - CARRE :X qui rend le carré d'un nombre.

2 - REP :MT qui rend le mot entré en répétant la première lettre.

3 - SUPP :MT qui rend le mot entré en supprimant l'avant-dernière lettre.

4 - DOUBLELISTE :X qui rend la liste doublée :
DOUBLELISTE [DO RE MI] produit [DO RE MI DO RE MI] .

5 - INFEGAL? :X :Y qui rend VRAI si le nombre X est inférieur ou égal au nombre Y, FAUX sinon.

6 - ABS :X qui rend la valeur absolue d'un nombre X.

7 - EQUAT1 :A :B qui rend la solution de l'équation $A X + B = 0$.

8 - EQUAT2 :A qui rend les solutions de l'équation $X^2 = A^2$.

9 - EQUAT3 :A qui rend les solutions de l'équation $|X| = A$.

10 - PAIR? :X qui rend VRAI si le nombre X est pair et FAUX sinon.

11 - INFECART? :X :Y :L qui rend VRAI si $|X - Y| < L$ et FAUX sinon.

12 - INFEGCART? :X :Y :L qui rend VRAI si $|X - Y| \leq L$ et FAUX sinon.

2 - Réfléchissons :

a) Ces procédures sont-elles toujours définies ?

b) Etudiez le domaine de définition de ces procédures (c'est-à -dire les conditions pour qu'elles soient définies).

c) Modifiez ensuite vos procédures pour qu'elles tournent bien !

****

Analyse de la situation:

Cette situation n'est pas une situation-problème. Il s'agit ici d'apprendre à manier une nouvelle instruction. Cette instruction sera ensuite utilisée durant la séance suivante et durant l'apprentissage des fonctions pendant le cours de mathématiques (M.-A. Egret, D. Guin, 1990).

Commentaires :

Les difficultés rencontrées par les élèves sont bien sûr, d'ordre mathématique : par exemple, la valeur absolue d'un nombre n'est pas une notion maîtrisée en troisième! Mais il n'y a pas de difficulté majeure à comprendre l'utilisation de cette instruction, même si son aspect fonctionnel n'est pas perçu par tous.

Séance 7 : LOGO NON GRAPHIQUE ET RECURSIVITE

Objectif de la séance:

- récursivité manipulant des mots et des listes

La séance commence par des exemples de procédures que l'on discute en classe. Nous proposons aux élèves de décomposer l'écriture d'une procédure récursive en deux étapes :

```
**********************************************************
*1 - test d'arrêt .                                      *
*2 - étape de réduction d'un objet LOGO à un objet LOGO plus petit .*
**********************************************************
```

Exemples de procédures récursives manipulant des objets LOGO (mots ou listes)

a) - Ecrire une procédure LONGUEUR :MOT qui rend la longueur d'un mot.

1 - test d'arrêt : Si le mot est vide , la longueur du mot est 0 :

```
SI VIDE? :MOT ALORS [ RENDS 0 ]
```

2 - étape de réduction : on se ramène à un mot plus petit, en supprimant la première lettre. Si le mot est de longueur n, on se ramène à un mot de longueur n - 1. On essaye de mettre en pratique la maxime :

" Si l'on sait faire pour un objet de longueur n - 1, on sait faire pour un objet de longueur n ".

L'étape de réduction consiste à expliciter la relation entre le traitement de l'objet de longueur n et celui de l'objet de longueur n - 1, ici la relation s'écrit facilement :

LONGUEUR :MOT = 1 + LONGUEUR SP :MOT

D'où :

```
POUR LONGUEUR :MOT
SI VIDE? :MOT ALORS [ RENDS 0 ]
                SINON [ RENDS 1 + LONGUEUR SP :MOT ]
FIN
```

b) - Ecrire une procédure CHOISIS :N :X qui rend l'élément N de la liste X (On suppose X non vide et 0 < N < longueur de X) .

1 - test d'arrêt : Si N est égal à 1, la procédure CHOISIS doit rendre le premier élément de la liste :

SI :N = 1 ALORS [RENDS PREM :X]

2 - étape de réduction : Il faut expliciter la relation entre le traitement de la liste X et le traitement de la liste SP :X. Si N est différent de 1, trouver l'élément N dans la liste X , c'est trouver l'élément N - 1 dans la liste SP :X, autrement dit :

CHOISIS :N :X = CHOISIS :N- 1 SP :X

D'où :

```
POUR CHOISIS :N :X
SI :N = 1 ALORS [ RENDS PREM :X ]
        SINON [ RENDS CHOISIS :N - 1 SP :X ]
FIN
```

c) - Ecrire une procédure INVERSE :MT qui rend le mot inversé.

1 - test d'arrêt : Si le mot est vide, son inverse est vide :

SI VIDE? :MT ALORS [RENDS "]

2 - étape de réduction : Si l'on sait inverser SP :MT, comment inverser MT ?

Autrement dit : INVERSE :MT = MOT (INVERSE SP :MT)(PREM :MT)

D'où :

```
POUR INVERSE :MT
SI VIDE? :MT ALORS [ RENDS " ]
        SINON [ RENDS MOT ( INVERSE SP :MT ) ( PREM : MT ) ]
FIN
```

****

Exercices sur les procédures récursives manipulant listes et mots :

1 - En vous inspirant de CHOISIS, écrivez une procédure LETTRE :N :MT qui rend la n ième lettre d'un mot :

ECRIS LETTRE 3 "CRI
I

Question : conditions sur N et MT ?

2 - En vous inspirant de LONGUEUR, écrivez une procédure SOMMER :L qui rend la somme des nombres d'une liste (si la liste est vide, la procédure rend 0) :

ECRIS SOMMER [3 4 5]
12

3 - En vous inspirant de SOMMER, écrivez une procédure PRODUIT :L qui rend le produit des nombres d'une liste (si la liste est vide, la procédure rend 1) :

ECRIS PRODUIT [3 4 5]
60

4 - Ecrivez une procédure EPELLE :MT qui écrit successivement chaque lettre du mot : EPELLE "CRI
C
R
I

5 - En vous inspirant de EPELLE, écrivez une procédure LISTEMOTS :PH qui écrit successivement chaque mot de la phrase :

LISTEMOTS [UN DEUX TROIS]
UN
DEUX
TROIS

6 - En vous inspirant de LISTEMOTS, écrivez une procédure BEGAIE :PH qui écrit successivement chaque mot de la phrase en répétant la première lettre :

BEGAIE [UN DEUX TROIS]
UUN
DDEUX
TTROIS

7 - En vous inspirant de INVERSE, écrivez une procédure INVPH :L qui inverse tous les mots d'une phrase :

ECRIS INVPH [UN DEUX TROIS]
TROIS DEUX UN

8 - En vous inspirant de INVPH et en utilisant la procédure INVERSE, écrivez une procédure INVTOUT :PH qui inverse tous les mots d'une liste et chaque mot :

ECRIS INVTOUT [UN DEUX TROIS]
SIORT XUED NU

9 - Ecrivez une procédure RELIETOUT :PH qui donne comme valeur à chaque mot de la phrase le mot suivant :

RELIETOUT [A 1 B BONJOUR C [IL PLEUT]]
ECRIS :A
1
ECRIS :B
BONJOUR
ECRIS :C
IL PLEUT

Remarque : La liste doit avoir une longueur paire !

10 - Ecrivez une procédure PRESENCE? :CAR :MT qui teste la présence d'une lettre dans un mot. La procédure doit rendre VRAI si la lettre est dans le mot, FAUX sinon :

ECRIS PRESENCE? "E "EUGENIE
VRAI
ECRIS PRESENCE? "E "ANON
FAUX

11 - Ecrivez une procédure PUISSANCE :N :X qui calcule récursivement x^N .

Question : conditions sur N et X ?

12 - Ecrivez une procédure SUPPRIME :CAR :MT qui supprime (éventuellement) une lettre donnée dans un mot :

ECRIS SUPPRIME "E "VENDANGE
VNDANG
ECRIS SUPPRIME "E "ANON
ANON

13 - Ecrivez une procédure ADDITION :A :B qui rend la somme de deux nombres en supposant que l'ordinateur ne sait que ajouter 0 ou 1 à un nombre donné.

14 - Ecrivez une procédure PRODUIT :A :B qui rend le produit de deux nombres en supposant que l'ordinateur ne sait que multiplier par 1 ou 2, diviser par 2 et additionner.

15 - Ecrivez une procédure CHEMIN :N qui rend le nombre de possibilités d'obtenir N en partant de 0 et en n'ajoutant que 1 ou 2.

****

Corrigé des exercices sur les procédures récursives manipulant listes et mots :

```
1 - POUR LETTRE :N :MT
    SI :N = 1 ALORS [ RENDS PREM :MT ]
              SINON [ RENDS LETTRE :N -1 SP :MT ]
    FIN

2 - POUR SOMMER :L
    SI VIDE ? :L ALORS [ RENDS 0 ]
                 SINON [ RENDS PREM :L + SOMMER SP :L ]
    FIN

3 - POUR PRODUIT :L
    SI VIDE ? :L ALORS [ RENDS 0 ]
                 SINON [ RENDS PREM :L * PRODUIT SP :L ]
    FIN

4 - POUR EPELLE :MT
    SI VIDE ? :MT ALORS [ ECRIS " ]
                  SINON [ ECRIS PREM :MT EPELLE SP :MT ]
    FIN

5 - POUR LISTEMOTS :PH
    SI VIDE ? :PH ALORS [ ECRIS [ ] ]
                  SINON [ ECRIS PREM :PH LISTEMOTS SP :PH ]
    FIN
```

```
6 - POUR BEGAIE :PH
    SI VIDE ? :PH ALORS [ ECRIS [ ] ]
                  SINON [ ECRIS  MOT ( PREM PREM :PH ) PREM :PH
                          BEGAIE SP :PH ]
    FIN

7 - POUR INVPH :PH
    SI VIDE? :L ALORS [ RENDS [ ] ]
                SINON [ RENDS PH ( INVPH SP :PH ) ( PREM :PH ) ]
    FIN

8 - POUR INVTOUT :PH
    SI VIDE? :L ALORS [ RENDS [ ] ]
                SINON [ RENDS PH ( INVTOUT SP :PH )
                         ( INVERSE PREM :PH ) ]
    FIN

 9 - POUR RELIETOUT :L
    SI VIDE? :L ALORS [ STOP ]
    RELIE PREM :L PREM SP :L
    RELIETOUT SP SP :L
    FIN

10 -POUR PRESENCE? :CAR :MT
    SI VIDE? :MT ALORS [ RENDS "FAUX ]
    SI PREM :MT = :CAR ALORS [ RENDS "VRAI ]
                       SINON [ RENDS PRESENCE? :CAR SP :MT ]
    FIN

11 - POUR PUISSANCE :N :X                N entier  > 0 et X > 0
    SI :N =1 ALORS [ RENDS :X ]
             SINON [ RENDS :X * PUISSANCE :N - 1 :X ]
    FIN
```

```
12 - POUR SUPPRIME :CAR :MT
   SI VIDE? :MT ALORS [ RENDS [ ] ]
   SI PREM :MT = :CAR ALORS [ RENDS SUPPRIME  :CAR SP :MT ]
                          SINON [ RENDS MOT ( PREM :MT )
                                    ( SUPPRIME  :CAR SP :MT )]
   FIN
```

Analyse de la situation:

Dans cette séance, nous avons proposé aux élèves d'approfondir la récursivité en travaillant sur les mots et les listes. Il nous paraissait important de ne pas nous restreindre aux projets graphiques pour lesquels souvent une compréhension partielle du fonctionnement est suffisante pour obtenir le résultat. L'étape de réduction est ici essentielle et peut paraître hors de portée d'élèves de ce niveau.

Commentaires :

Les élèves ont paru surpris du travail demandé : ils sont moins habitués à travailler avec les mots et les listes, le résultat est moins accrocheur. Mais ils ont apprécié la découverte de l'étape de réduction et, pris au jeu, ont réussi tous à faire quelques exercices.

Deuxième partie : Analyse des productions des élèves

Nous avons retenu trois situations, que nous avons analysées plus particulièrement :

Situation 1 : les courbes, à savoir :
les arbres, les croix, la courbe de Von Koch et les carrés,

Situation 2 : les projets,

Situation 3 : le test de prévisions d'exécutions de procédures récursives.

Pour cela, après une analyse a priori des situations, des observations antérieures et une analyse a posteriori des procédures de résolution des élèves, nous proposons de définir des critères d'analyse.

La complexité des activités de programmation, ou de prévision d'exécution de procédures, rend nécessaire la définition de plusieurs critères d'analyse. Chaque critère reflète un des aspects de l'activité.

Nous présentons ensuite les résultats des observations de ces critères, pour mettre en évidence les ***difficultés*** *rencontrées et les* ***compétences acquises****. Le but est de dégager de ces observations des éléments d'information sur les représentations que les élèves se sont construites, afin de pouvoir améliorer l'enseignement.*

Situation 1 : LES COURBES [1]

1 LES ARBRES.

Les élèves travaillent par groupes de deux ou trois. Huit groupes d'élèves (sur neuf) ont choisi de programmer la famille des arbres (deux séances de travail, c'est-à-dire 4 heures).

L'analyse de l'activité sur les arbres ayant été détaillée par ailleurs (C. Dupuis, D. Guin, 1989), nous ne reviendrons ici que sur les différents codages observés.

Choix de codage.

Aucune relation entre les différentes longueurs qui apparaissent dans un même arbre n'était explicitée dans la consigne et nous n'avons donné aucune indication orale. Trois modalités de codage des longueurs sont apparues : le codage descriptif, le codage analytique et l'absence de codage.

codage descriptif : à chaque étape, il y a **autant de variables** d'entrée dans la procédure que de **longueurs différentes** dans l'arbre (ce qui permet d'éviter l'expression d'une quelconque relation entre longueurs). De plus, **les noms des variables changent d'une étape à l'autre.**

codage analytique : une même variable est utilisée dans toutes les étapes ; des **relations entre les longueurs sont alors explicitées en termes de variables** (C et C/2, C et 2*C ou C et C-10). L'exemple de programme donné dans la première partie page 11 est en codage analytique.

absence de codage : **aucune variable** d'entrée n'est utilisée. Les longueurs données a priori sont modifiées par approximations successives en fonction de l'aspect de la courbe dessinée sur l'écran.

---

[1] Les dessins fournis aux élèves ainsi que les consignes de travail sur les courbes se trouvent dans la "Séance 1", page 7.

Un exemple de procédures en codage descriptif.

Un groupe (ALI+STé+SAB), qui travaillait en codage descriptif, a défini les procédures suivantes :

POUR ARBRE1 : COTE
.....
POUR ARBRE2 : COTE1 : COTE2
.....
POUR ARBRE3 : C1 : C2 : C3
.....

POUR ARBRE4 : C4 : C5 : C6 : C7 : C8
ARBRE3 : C4 : C5 : C6
TG 45
ARBRE3 : C5 : C6 : C7
TG 45
ARBRE3 : C6 : C7 : C8
RE : C6
TD 90
ARBRE3 : C6 : C7 : C8
RE : C6
TG 45
RE : C5
TD 90
ARBRE3 : C5 : C6 : C7
TD 45
ARBRE3 : C6 : C7 : C8
RE : C6
TG 90
ARBRE3 : C6 : C7 : C8
RE : C6
TD 45
RE : C5
TG 45
FIN

Au moment de l'exécution, ces élèves ont respecté les inégalités visibles en donnant des valeurs de plus en plus petites aux branches d'un même arbre.

Deux remarques sur le programme ci-dessus :

- ce programme peut paraître complexe, mais on peut y déceler une certaine utilisation des symétries de la figure en faisant apparaître les blocs en italique ci-dessus (où l'on retrouve les mêmes procédures et la même structure).

- ARBRE4 est le quatrième arbre dessiné donc en fait l'arbre de niveau 5. Dans la situation proposée, nous n'avions donné aux élèves que les dessins des niveaux 1, 2, 3 et 5, le niveau 4 ayant été supprimé pour observer si la régularité des programmes permettrait aux élèves de "voir" cet arbre fantôme (cf. p. 8). Le groupe d'élèves qui a écrit le programme ci-dessus ne l'a pas vu.

Le nombre de variables utilisées pourrait faire croire que ces élèves possèdent à fond la notion de variable en Logo. En fait, ces élèves ne font ici que **désigner des objets différents par des noms différents.** L'activité de désignation est à la portée d'élèves bien plus jeunes (D. Guin, J.G. Helm, 1984).

Réussite et choix de codage.

Si nous définissons la réussite comme l'écriture et la coordination des procédures pour l'obtention des quatre arbres demandés, celle-ci est obtenue par **les deux groupes qui ont travaillé en codage descriptif** et un groupe qui, après avoir écrit les procédures des trois premiers arbres en codage analytique, a écrit la dernière procédure en codage descriptif. **Aucun des quatre groupes qui ont choisi de travailler en codage analytique n'a obtenu les quatre arbres demandés.** Un groupe avait choisi de travailler sans variable et n'a pas non plus obtenu les quatre arbres.

Nous avons développé par ailleurs (C. Dupuis, D. Guin, 1989) les raisons pour lesquelles la procédure de résolution par codage descriptif est plus simple à mettre en oeuvre que celle par codage analytique ; une des raisons est qu'elle constitue une stratégie de décomposition d'un problème complexe en sous-problèmes pouvant être résolus séparément. Mais il faut signaler que, même si le codage descriptif constitue la procédure de résolution la plus efficace dans cette situation pour nos élèves, ce type de codage est un obstacle à l'acquisition de la notion de récursivité. En effet, le codage analytique est nécessaire pour l'écriture récursive des procédures.

2 LES CROIX

Observations.

Six groupes d'élèves ont écrit des programmes pour la famille des croix. Un seul groupe (DAV+REG) a réussi à exploiter les régularités des croix : ses programmes ont été reproduits dans la présentation de la séance 1 (cf. p. 9).

Pour les cinq autres groupes d'élèves, quel que soit le choix de codage, leurs programmes relèvent plus du "bricolage" que de l'analyse, et n'exploitent absolument pas les régularités de la figure. L'emboîtement de leurs procédures n'est pas du tout invariant.

Commentaires.

Contrairement à ce qui s'est passé pour les arbres, le choix du codage descriptif n'a pas entraîné la réussite. En effet, **la difficulté principale** ici n'est pas d'exprimer une relation entre des longueurs mais plutôt d'**analyser la figure en exploitant ses régularités** pour rendre simple le programme Logo correspondant.

Cinq groupes d'élèves sur six n'ont pas perçu la figure comme pouvant être décomposée en 4 sous-figures identiques. Pour voir cette décomposition, il aurait été nécessaire d'opérer une **réorganisation perceptive**. En effet, la première perception de la croix est : "un ensemble de segments se coupant à angle droit". Cette perception est adaptée à une tâche de reproduction de la croix à la main, avec règle et équerre. Elle n'est pas opératoire pour obtenir les croix par l'exécution d'un programme Logo, dont l'écriture nécessite l'**identification de l'invariant**. A la suite de R. Duval (1988), nous appellons **appréhension opératoire** cette capacité d'opérer une réorganisation perceptive en fonction du registre d'exécution que constitue la programmation en Logo.

La simplicité apparente des dessins des croix ne facilite pas la réorganisation perceptive. De plus, le travail sur la recherche d'invariants géométriques réalisé auparavant avait porté essentiellement sur des figures fermées ; il a donc sans doute plus favorisé la réorganisation perceptive pour la famille des carrés que pour cette famille de courbes. Nous verrons que l'invariance d'emboîtement y est réalisée par une plus forte proportion d'élèves.

3 LA COURBE DE VON KOCH

La courbe de Von Koch est la plus simple en ce qui concerne le nombre de longueurs différentes en jeu à chaque étape : il n'y a qu'une longueur !

Observations.

Sept groupes ont travaillé plus ou moins longtemps sur la courbe de Von Koch. Deux groupes (DAV+REG, ALI+STé+SAB) ont écrit les cinq programmes correspondant aux quatre courbes dessinées et au niveau manquant (la courbe fantôme). L'un d'entre eux a, comme à son habitude, travaillé en codage descriptif et donné un nom différent à chaque étape à la variable (ALI+STé+SAB, cf. p. 49), tandis que l'autre (DAV+REG, cf. p. 9) a conservé le même nom de variable partout. Le programme de ce groupe est identique au corrigé.

Un seul groupe (MAR+KAR) a eu des difficultés de coordination dès la troisième étape. Dans les programmes des étapes 2 et 3, ces élèves ont intégré l'instruction initiale TD 90 (tourne à droite de 90°) à la procédure. Après divers essais, ils l'ont supprimée à la deuxième, mais pas à la troisième étape. De sorte que, en emboîtant cette procédure dans celle de la quatrième étape (sans voir le niveau manquant), ils sont arrivés à une figure incompréhensible et ont abandonné.

Les programmes écrits par les quatre autres groupes donnent, à l'exécution, le résultat attendu. Deux groupes (ALE+FLO, KLA+CAR) n'ont écrit que le programme de la deuxième courbe. Les deux autres groupes (NAT+RAC, CYR) sont arrivés jusqu'au niveau 3. A chaque étape, ils utilisent la procédure de l'étape précédente de la même manière. L'emboîtement est donc invariant.

Commentaires.

La première difficulté est l'estimation des angles de rotation de la tortue à la deuxième étape (et le professeur de mathématique peut en profiter pour rappeler la somme des angles d'un triangle !). Dès la troisième étape, se pose aux élèves un problème de coordination des procédures qu'ils ont écrites : les élèves doivent prendre en compte, simultanément, l'état de la tortue à la fin et au début des procédures à emboîter, pour estimer l'angle de rotation de la tortue.

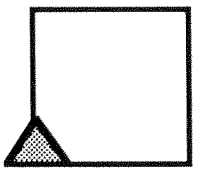
Il n'y a pas, dans cette famille, de problème d'expression d'une relation entre des longueurs comme dans la famille des arbres ou de réorganisation perceptive à opérer comme pour les croix. C'est sans doute la plus simple des trois familles présentées jusqu'ici. En effet, à chaque étape, il n'y a qu'une seule longueur en jeu. De plus, il n'y a pas de choix à opérer dans l'ordre du tracé qui se fait naturellement de gauche à droite. Enfin, il n'y a pas de superposition puisque la courbe d'une étape est seulement emboîtée "en plus petit" à l'étape suivante.

4 LES CARRES [1]

1 Une solution optimale

Deux groupes ont choisi de répéter la plus petite séquence possible. Voici les programmes de l'un d'eux :

POUR CARRE1 :N
REPETE 4 [AV :N TD 90]
FIN

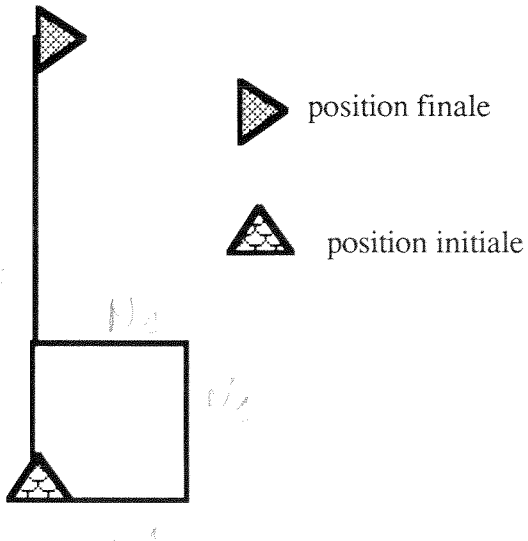


 Positions initiale et finale de la tortue

POUR CARRE2 :N
REPETE 4 [CARRE1 :N / 3 AV :N TD 90]
FIN

CARRE2 réalise le deuxième carré dessiné.

La séquence entre crochets dans le programme CARRE2 correspond au schéma ci-contre :

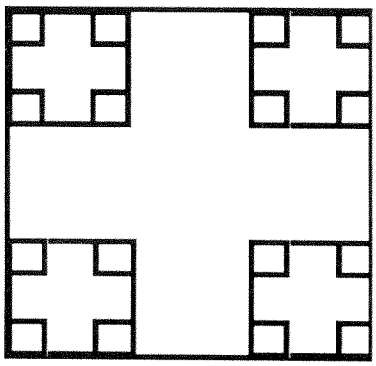


CARRE3 réalise le carré fantôme, c'est à dire celui qui aurait sa place entre le deuxième et le troisième carré dessiné.

---

[1] cf. p. 12

le carré fantôme



POUR CARRE3 :N
REPETE 4 [CARRE2 :N / 3 AV :N TD 90]
FIN

CARRE4 réalise le troisième carré dessiné dans la consigne donnée aux élèves.

POUR CARRE4 :N
REPETE 4 [CARRE3 :N / 3 AV :N TD 90]
FIN

2 Définition de critères d'analyse.

Etat de la tortue.

L'état de la tortue se caractérise par sa position et son orientation. Nous présenterons ici deux modalités[1] de **prise en compte de l'état de la tortue** à la fin des procédures graphiques (ou d'une liste d'instructions) destinées à être intégrées à un projet.

*** retour à une position "standard"** à la fin de toutes les procédures destinées à être intégrées. Une position "standard" est une position à partir de laquelle la coordination de cette procédure avec les autres est économique. Il s'agit souvent de la position initiale (modalité **PS**).

*** * aucun retour à une position "standard"** ne figure dans une procédure (au moins) destinée à être intégrée. Aucun mouvement de la tortue n'est effectué en plus de ceux qui sont nécessaires au tracé (modalité **sans PS**).

---

[1] Pour l'analyse des activités de programmation pour d'autres courbes, nous avons été amenées à définir d'autres modalités du critère Etat de la tortue (Dupuis, Egret, Guin, 1987).

Régularités

Les programmes écrits utilisent les régularités internes de la courbe : oui / non. La non-utilisation des régularités dans la programmation peut conduire à des programmes très embrouillés.

Fantôme

Une procédure est écrite pour le carré fantôme : oui / non. L'écriture d'une procédure pour le carré fantôme est nécessaire pour que soit respectée l'invariance d'emboîtement d'une procédure dans la suivante.

Coordination

Ecriture et coordination correctes des procédures pour les trois carrés dessinés : oui /non.[1]

Gestion de la composition de la relation.

L'emboîtement de chaque procédure dans la suivante pose le problème de la composition de la relation N $\longrightarrow$ N / 3 avec elle-même. Les élèves peuvent choisir de confier la gestion de la composition au dispositif informatique en utilisant une procédure CARRE(i+1) :N qui appelle une procédure CARRE(i) :N / 3, comme c'est le cas dans les programmes figurant ci-avant (cf. p. 54). Mais ils peuvent aussi décider de gérer directement la composition en écrivant par exemple les procédures CARRE2 :N et CARRE3 :N suivantes :

```
POUR CARRE2 :N
REPETE 4 [ CARRE :N  AV 3 * :N  TD 90 ]
FIN

POUR CARRE3 :N
REPETE 4 [ CARRE2 :N  AV 9 * :N  TD 90 ]
FIN
```

[1] Dans d'autres situations, nous avons distingué plusieurs types d'erreurs de coordination (Dupuis, Egret, Guin, 1987).

et ainsi de suite. Dans ce cas, N reste toujours la longueur du côté du plus petit carré dessiné ; les longueurs des autres carrés (3*N et 9*N), qui résultent de la composition de la relation avec elle-même, sont exprimées directement par l'élève dans son programme. Ce critère n'a évidemment de sens que si l'élève choisit une procédure de résolution par codage analytique, c'est-à-dire s'il choisit d'exprimer une relation (ce qui est le cas pour tous les élèves dans cette situation). En fait, ce critère prendra ici trois modalités :

directe : la composition est gérée **directe**ment par l'élève

dispositif : la composition est gérée par le **dispositif** informatique.

débat : les élèves d'un groupe (ALI + STé + SAB) ont débattu du problème de la gestion sans pouvoir se mettre d'accord.

Economique

Les élèves ont-ils choisi de répéter la plus petite séquence possible ? : OUI / NON / 0. La modalité 0 signifie que les élèves n'ont pas réalisé la troisième étape.

3 Observation des critères : tableau de résultats.

	ETAT	Régularité	Fantôme	Coordination	Gestion	économique
ALE + FLO	PS	OUI	OUI	OUI	dispositif	OUI
MAR	PS	OUI	OUI	OUI	dispositif	OUI
SEB	PS	OUI	OUI	OUI	directe	NON
DAV + REG	PS	OUI	OUI	OUI	directe	NON
GIL + CHR	PS	OUI	NON	NON	directe	0
SAR	sans PS	NON	NON	NON	dispositif	0
ALI+STé +SAB	sans PS	NON	NON	NON	débat	0

4 Commentaires.

Codage analytique pour tous !

Tous les élèves ont utilisé la relation fonctionnelle qui avait été donnée : ils ont tous travaillé en codage analytique. La mention du codage ne figure donc pas dans le tableau.

Etat et régularité

La situation avait été choisie suffisamment complexe pour que, sans retour à une position standard et sans utilisation des régularités de la figure, il soit quasiment impossible d'obtenir les trois carrés. C'est ce qui se passe pour les deux groupes du bas du tableau.

Réussite

La réussite complète, c'est-à-dire l'écriture et la coordination des procédures pour l'obtention des trois carrés, se traduit par le code OUI dans la colonne Coordination. Quatre groupes ont donc réussi à programmer les trois figures, deux d'entre eux ayant choisi de répéter la séquence la plus économique et de confier la gestion de la relation au dispositif, les deux autres ayant choisi la gestion directe et une séquence moins économique.

L'invariance d'emboîtement

L'écriture d'une procédure pour le carré fantôme est une nécessité absolue pour que soit respectée l'invariance d'emboîtement. Bien entendu, le respect de l'invariance d'emboîtement ne figurait pas parmi les consignes. D'ailleurs, l'invariance au sens strict du terme n'est pas respectée par ceux qui choisissent la gestion directe. Néanmoins, ils n'en sont pas loin et, pour eux, le passage à une écriture récursive ne sera pas trop difficile.

Situation 2 : LES PROJETS.

Sept projets ont été proposés aux élèves (séance 4, page 26). Ceux-ci ont travaillé seuls ou en groupes pendant deux à trois séances de travail de deux heures. Le projet 4 est le premier projet nécessitant un traitement lié au test d'arrêt. Il s'agit d'un déplacement, crayon levé. Ce déplacement a été négligé par tous les élèves (cette erreur est signalée dans le tableau ci-après par la mention "OD" pour Oubli du Déplacement). On peut penser qu'ils n'ont pas "vu" le déplacement à effectuer. Dans le projet 5, l'instruction avant le STOP correspond à un dessin, beaucoup plus "visible" qu'un déplacement. Les élèves ont alors découvert que l'on peut mettre des instructions dans les crochets, avant le STOP.

1 Définition des critères d'analyse

Structure du programme écrit

RC : le programme écrit est une procédure avec appel Récursif Central

DAT : le programme est décomposé en deux procédures récursives avec appel terminal.

DA : le programme est décomposé en deux procédures récursives, l'une avec appel terminal et la deuxième avec appel au début.

REP : le programme utilise l'instruction REPETE.

Résultat obtenu

C : le résultat obtenu à l'écran est celui qui était demandé

Le résultat obtenu à l'écran n'est pas celui qui était demandé à cause de l'erreur suivante:

E : le programme écrit comporte des erreurs de latéralisation ou des erreurs sur la valeur d'un angle.

OD : Oubli du Déplacement central : c'est une erreur typique du projet 4.

2 Tableaux de résultats

Type	1 graph.	2 graph.	3 liste	4 graph. Stop	5 ecr+gra Stop	6 graph. Stop	7 liste Stop	nombre de séances	prc[1]
SEB	RC C	RC C	RC C	RC OD	RC C	RC C	RC C	3	6
RAC+CHR	RC C	RC C	RC C	RC OD	0	RC C	0	3	4
SAR+NAT	RC C	RC C	RC C	DA puis RC C puis OD	RC C	0	0	3	4
KLA+CAR	RC (2) E (2)	RC C	RC C	RC OD	RC E puis C	RC C	RC C	3	4
DAV+MAR	RC C	RC C	RC C	0	0	0	0	2	3
CYR	RC C	RC C	RC C	0	0	0	0	2	3
REG+GIL	RC C	RC C	RC C	RC OD	0	0	0	3	3
KAR	RC E puis C	RC C	0	RC OD	0	0	0	2	2
MAG+FLO +ALE	REP puis RC C	RC C	0	0	0	0	0	2	2
ALI+STé +SAB	DAT (2) C (2)	RC C	0	0	0	0	0	2	1

[1] prc pour : procédures récursives écrites

Dans le tableau précédent, pour chaque groupe d'élèves, les modalités du premier critère figurent en première ligne et les modalités du deuxième critère figurent en deuxième ligne.

Dans les deux dernières colonnes figurent le nombre de séances de travail de 2 heures que les élèves ont consacré à cette activité, puis le nombre de procédures récursives correctes écrites pendant ces séances.

3 Commentaires

Les deux groupes qui ont écrit le moins de procédures récursives correctes ont, en fait, essayé pendant toute une séance d'obtenir le résultat attendu pour le projet 1 en utilisant des structures qui leur étaient plus familières. Le groupe MAG+FLO+ALE a tenté d'écrire un programme utilisant l'instruction REPETE. Le groupe ALI+STE+SAB a eu, dans cette situation, l'attitude qu'il a souvent eu devant une difficulté nouvelle : décomposer un problème complexe (écriture d'une procédure avec appel récursif central) en sous-problèmes plus simples (écrire de procédures avec appel récursif terminal).

Tous les élèves ont écrit au moins une procédure récursive dont à la fois la structure et le résultat sont corrects. Ces résultats montrent qu'il est possible que **des élèves de collège conçoivent et écrivent des procédures récursives avec appel récursif central**. Cependant, ces compétences ne peuvent être acquises qu'après une solide alphabétisation informatique : en effet, pour que des élèves puissent parvenir au maniement de la structure récursive, il est nécessaire qu'ils ne soient pas arrêtés, par exemple, par des problèmes de coordination de procédures graphiques (à ce stade, un seul élève (KAR) a eu des problèmes de coordination de ses procédures).

Situation 3 : TEST DE PREVISIONS D'EXECUTIONS DE PROCEDURES RECURSIVES [1].

1 Définition des critères d'analyse

prévision d'exécution en récursivité centrale.

S - exécution séquentielle

Les procédures sont exécutées dans l'ordre où elles sont écrites, une seule fois. C'est le type de prévision le plus "loin" de la récursivité. On n'y voit aucune trace de l'enseignement de la récursivité.

R - l'appel récursif central ne déclenche pas la suspension de l'exécution de la procédure appelante.

La conception spontanée de l'auto-référence est le retour au début du programme[2]. Les différentes modalités R présentées ci-dessous reflètent cette conception. Elle est d'autant plus difficile à mettre en échec qu'elle permet, par ailleurs, des prévisions d'exécution correctes pour les programmes en récursivité terminale.

R 0 et R 0 AT Les procédures situées après l'appel ne sont jamais exécutées,

R 0 AT : et la prévision d'exécution de SPA est correcte

R 0 : et la prévision d'exécution de SPA n'est pas correcte

R 1 et 2 Les procédures situées après l'appel sont prises en compte lorsque le test d'arrêt est vrai et sont alors exécutées :

R 1 : soit pour la seule valeur initiale de la variable

R 2 : soit pour la dernière valeur de la variable avant le STOP.

RFin 1, 2 et 3 Toutes les procédures sont exécutées dans l'ordre où elles sont écrites et pour toutes les valeurs de la variable. Mais la variable change de valeur :

RFin 1 soit après chaque exécution de SPB

RFin 2 soit avant chaque exécution de SPB

---

[1] cf. p. 31
[2] cf. p. 20

RFin 3 soit avant et après chaque exécution de SPB

Les modalités classées sous S et R traduisent la **méconnaissance de l'aspect suspensif** dans l'exécution d'une procédure récursive.

D A T - La procédure fonctionne comme une succession de Deux programmes récursifs avec Appel Terminal.

On voit ici en oeuvre la conception spontanée d'un retour à un début ; c'est, sans doute, de plus le "souvenir" qu'en récursivité centrale il y a autant d'exécutions de la procédure SPA que de la procédure SPB. Un pas est fait dans la bonne direction. Une partie de **l'aspect suspensif** de la récursivité a été perçue.

C - prévision correcte.

Même dans le cas d'une prévision correcte isolée, il n'est pas certain que le modèle de fonctionnement que l'élève s'est construit soit correct. Un modèle global "miroir", tel qu'il est décrit ci-après, permet des prévisions correctes dans de nombreuses situations.

Prévisions pour ESSAI 30

S	R0	R1	R2	Rfin1	RFin2	RFin3	DAT	Correct
SPA 30	SPA 30	SPA 30	SPA 30	SPA 30	SPA 30	SPA 30	SPA 30	SPA 30
SPB 30	SPA 20	SPA 20	SPA 20	SPB 30	SPB 20	SPB 20	SPA 20	SPA 20
	SPA 10	SPA 10	SPA 10	SPA 20	SPA 20	SPA 10	SPA 10	SPA 10
		SPB 30	SPB 10	SPB 20	SPB 10		SPB 30	SPB 10
				SPA 10	SPA 10		SPB 20	SPB 20
				SPB 10			SPB 10	SPB 30

MIROIR - la prévision est faite suivant un modèle global "miroir".

L'observation des exécutions de procédures récursives amène un certain nombre d'élèves à se construire un modèle global du fonctionnement d'une procédure récursive que l'on peut exprimer ainsi : << Avant l'appel, ça diminue ; après l'appel, ça augmente.>>[1]. Ce modèle est renforcé par le fait qu'il permet des prévisions d'exécution

[1] cf. p. 35

correctes dans des situations présentant une certaine symétrie. Il ne peut donc être mis en évidence qu'à propos d'exercices ne présentant pas cette symétrie. Dans ce cas, on verra l'élève simuler l'exécution des procédures avant l'appel, puis compléter "par symétrie, comme dans un miroir". Ce modèle "miroir" n'est pas le résultat des acquis antérieurs à l'enseignement de la récursivité mais bien d'une analyse du fonctionnement de la récursivité.

Prenons l'exemple de la procédure suivante :

```
POUR        TOTI  :MOT
SI VIDE? :MOT  ALORS [STOP]
ECRIS  DER  :MOT
TOTI  SD  :MOT
ECRIS  PREM  :MOT
FIN
```

La prévision d'exécution était demandée pour TOTI "SAC. La prévision correcte est :

C A S S S S.

Suivant un modèle " miroir " , l'élève fournit la réponse :

C A S S A C.

l'interprétation du test d'arrêt

Certains élèves exécutent **une dernière fois** la procédure pour la valeur de la variable pour laquelle le test d'arrêt est VRAI. La position de ce test, au début du programme récursif, explique cette erreur. La structure du programme : test puis action, est analogue à celle que l'on rencontre dans les structures itératives du type TANT QUE....FAIRE, dont J.Rogalski et G. Vergnaud (1987) soulignent qu'elles posent plus de problèmes que la structure REPETE.....JUSQU'A qui est plus proche du modèle spontané.

Cette interprétation erronée du test est indépendante de la modalité d'interprétation de la récursivité centrale. On notera qu'elle n'est pas repérable lorsque le test d'arrêt est du type : SI :N = 0 ALORS [STOP] et que N désigne la dimension d'un dessin à exécuter.

A titre d'exemple, dans la modalité R1 du critère précédent, on observera la prévision ci-contre pour ESSAI 30 :

```
SPA  30
SPA  20
SPA  10
SPA   0
SPB  30
```

Cette interprétation est repérée dans le tableau par la mention "+T" (T comme Test) et se retrouve surtout dans le Test 1. Aucune mention n'est faite dans le cas contraire.

Les deux critères qui suivent sont propres aux procédures non graphiques :

Ecriture de la valeur de la variable au moment de l'appel

Dans les procédures où interviennent des listes ou des écritures de valeurs de variables, certains élèves écrivent aussi, dans leurs prévisions d'exécution, la valeur de la variable ou de la liste au moment de l'appel. Il ne semble pas que cela vienne d'une réelle confusion entre ce qui apparaîtra ou n'apparaîtra pas à l'écran, mais plutôt de la nécessité de noter " où l'on en est". En effet, cette écriture, qui est assez fréquente dans le Test 1 surtout dans les exercices 5 et 6, a pratiquement disparu dans le Test 2 avec le rappel de la consigne.

Cette écriture est repérée dans les tableaux par la mention "+V" (V comme Variable). Aucune mention n'est faite dans le cas contraire.

Modification de la valeur de la variable par l'instruction ECRIS

Dans les exercices demandant l'écriture de valeurs de variables (Test 1 ex. 1 et Test 2 ex. 5), on constate que pour certains élèves l'instruction ECRIS : N - 1 modifie la valeur de la variable N, comme s'il s'agissait d'un appel récursif (N prend la valeur de N - 1).

Cette interprétation est repérée dans les tableaux par la mention "+E" (E comme Ecris). Aucune mention n'est faite dans le cas contraire.

2 Les tableaux de résultats.

Test 1

	ex.1	ex.2	ex.3	ex.4	ex.5	ex6
Réc	centrale	appel au début	terminale	centrale	centrale	centrale
Type	écriture de valeurs de variables	graphique	graphique	graphique	liste	liste
CYR	Correct	Correct	Correct	Correct	Correct	Correct
SEB	Correct	Correct	Correct	Correct	non identifié	non identifié
STé	RFin2+E+V	Correct	Correct	Correct	RFin2 +V	RFin2 +V
CHR	Correct	Correct	Correct	RFin2 +V	RFin2 + V	RFin2 +V
SAB	R0 + T	AT	Correct	DAT	S + V	0
RAC	R2 + T	AT	Correct	DAT	S + V	S +V
KLA	R2 + T	AT	Correct	DAT	S + V	S +V
REG	RFin1+T+E	AT	Correct	RFin1	S+V	S+V
MAR	R0+T	Correct	Correct	DA	S+V	S+V
ALE	RFin2+E	AT	Correct	RFin2	R0	0
ALI	R1+T	R1	Correct	R1	R1	R1
SAR	RFin3+T+V	AT	Correct	RFin2	RFin2+V	non identifié
NAT	R0+T	AT	Correct	R1	RFin2+V	RFin2+V
CAR	R0+T	AT	Correct	DAT	S+V	S+V
GIL	DAT+T	AT	Correct	Correct	S+V	S+V
MAG	R1+T	AT	Correct	DAT	R1	R1
FLO	RFin2+E	AT	Correct	RFin2	R0	0
KAR	R0+T	R1	Correct	RFin3	S+V	S+V
DAV	R0+T	AT	Correct	RFin2	R0	RFin1

Test 2

	ex.1	ex.2	ex.3	ex.4	ex.5
Récursivité	terminale	appel au début	centrale	centrale	centrale
Type	liste	liste	liste	graphique	écriture de valeurs de variables
CYR	Correct	Correct	Correct	Correct	Correct
SEB	Correct	Correct	Correct	Correct	Correct
STé	Correct	Correct	Correct	Correct	Miroir+E+V
CHR	Correct	Correct	Correct	Correct	Miroir+E+V
SAB	Correct	Correct	Correct	Correct	R0ATE+V
RAC	Correct	Correct	Correct	Correct	R0
KLA	Correct + T	AT	Miroir+C	Correct	Miroir+C
REG	non identifié	non identifié	Miroir+I	Correct	R0
MAR	non identifié	AT	Miroir+C	Correct	0
ALE	Correct	Correct	RFin1 ou2	Correct	0
ALI	Correct	Correct	R1	Correct	Miroir +I
SAR	Correct + V	RFin2 + V	RFin2 + V	Correct	R0AT sur ECRIS:N-1
NAT	Correct + V	non identifié	RFin2	Correct	R0
CAR	Correct	RFin2	RFin1 ou 2	Correct	non identifié
GIL	Correct	R1	DA	DAT+C	DAT+C
MAG	non identifié	Correct	RFin1 ou 2	Correct	Miroir +I
FLO	Correct	RFin2	RFin2	RFin2	0
KAR	non identifié	R0 + V	RFin1 ou 2	Correct	Miroir +C

nota bene

La modalité "non identifié" signifie qu'il ne nous a pas été possible d'identifier la procédure qui peut amener à la prévision fournie par l'élève.

La mention "0" correspond à la non-réponse.

La modalité DA "Deux Appels" est un phénomène local. Quelques élèves ont considéré le programme avec appel récursif central comme la succession des 2 programmes récursifs déjà simulés (SPA ou SPB vides) aux deux exercices précédents. La situation avait été construite pour rendre visible cette interprétation non pertinente.

Nous n'avons pas défini de critère particulier pour analyser les prévisions d'exécution en récursivité avec appel au début (ex. 2 Test 1 et 2). Elles peuvent, en effet, être analysées avec une partie des modalités du critère "prévisions d'exécution en récursivité centrale" (celles qui ont encore un sens dans ce cas) plus une modalité particulière AT. Cette modalité signifie que les élèves ont prévu l'exécution exactement comme s'il s'agissait d'un programme avec appel terminal. Cette modalité est beaucoup plus fréquente dans la première séance du test que dans la deuxième. La conception spontanée de l'auto-référence est donc toujours présente ; on peut remarquer que l'entraînement par quelques exercices fait diminuer le nombre de prévisions de ce type sans les faire disparaître complètement.

relations entre les critères

Les trois critères : "interprétations du test d'arrêt ", "écriture de la valeur de la variable au moment de l'appel" et "modification de la valeur de la variable par l'instruction ECRIS" apparaissent comme **indépendants** du premier critère "prévision d'exécution en récursivité centrale". Ils correspondent à des difficultés qui ne sont pas directement liées à la conception de la récursivité.

3. Evolution des conceptions sous-jacentes.

La lecture du tableau met en évidence trois phénomènes :

Les exercices les mieux réussis concernent la récursivité terminale, l'un d'entre eux étant même réussi par tous les élèves.

En ce qui concerne les prévisions d'exécution en récursivité centrale, le phénomène le plus marquant est la grande variété des erreurs et l'instabilité des comportements de réponses, sauf pour deux élèves : l'un répond correctement à tous les exercices et le deuxième à presque tous.

Les trois derniers exercices concernent tous trois la récursivité centrale (leurs énoncés figurent dans la première partie, page 34). Pourtant l'un d'entre eux est beaucoup mieux réussi que les deux autres.

Dans les paragraphes qui suivent, nous allons développer l'interprétation de ces phénomènes en termes de conceptions des élèves.

conception spontanée

La conception spontanée de l'auto-référence, où l'appel récursif déclenche le retour au début du programme, permet une prévision d'exécution correcte pour les programmes en récursivité terminale. Cette conception spontanée existe avant tout enseignement de la récursivité. Nous l'avons d'ailleurs observée, comme prévu, lors de la séance 2 d'introduction de la récursivité, où elle a été contredite par les exemples dont la récursivité n'était pas terminale. En fait, cette conception persiste chez certains élèves, ce qui permet d'expliquer la réussite massive pour les exercices concernant la récursivité terminale.

retour à une conception antérieure ou plus primitive

En ce qui concerne les prévisions d'exécution en récursivité centrale, tout se passe comme si, lors de difficultés, les élèves revenaient à des conceptions antérieures ou plus primitives.

Les prévisions de type S correspondent à une conception antérieure à tout enseignement de la récursivité. La modalité S ne se rencontre que dans deux exercices (Test 1, exercices 5 et 6) qui sont les premiers de type "liste" et dans lesquels la récursivité est centrale. Ce sont des exercices plutôt difficiles et qui ont surpris les élèves. Est-ce la raison pour laquelle certains reviennent à une conception antérieure à tout enseignement de la récursivité ? C'est possible, d'autant plus que cette modalité a complètement disparu vers la fin du test, avec l'entraînement et le rappel des consignes.

La modalité DAT (Deux Appels Terminaux) peut être vue comme une tentative de décomposer le problème que pose la récursivité centrale en deux sous-problèmes de récursivité terminale, dont la solution est connue ou peut être trouvée en faisant référence uniquement à la conception spontanée de l'auto-référence. La modalité DAT a quasiment disparu lors de la deuxième séance du test (on ne l'observe plus que chez un élève).

Ce phénomène de retour à une conception antérieure "dans la difficulté" a déjà été observé au cours de l'expérience. Ainsi, à la fin de la phase d'enseignement de la programmation structurée, nous avions écrit : << on peut affirmer (...) que les éléments de base d'une programmation graphique structurée sont en place chez ces élèves. La structuration des programmes est devenue une méthode efficace de résolution de problèmes>> (Dupuis, Egret, Guin, 1987). Force a été de constater, lors de l'écriture des programmes pour les **arbres,** que deux groupes d'élèves avaient écrit des procédures graphiques non structurées (Dupuis, Guin, 1989). Ce phénomène n'est évidemment pas propre à la programmation en Logo.

conception "miroir"

La conception "miroir" est apparue dans les exercices 3 et 5 du test 2. Pour bien préciser le contexte, nous allons détailler les résultats des exercices 3, 4 et 5 du Test 2.

1 Résultats marginaux

Les prévisions de la majorité des élèves sont correctes pour l'exercice 4 (seize réussites sur dix-huit élèves présents). En revanche, ils ne sont que six à faire une prévision correcte pour l'exercice 3 et deux pour l'exercice 5. On observe trois prévisions correspondant à un modèle global "miroir" pour l'exercice 3 et six pour l'exercice 5.

2 Résultats croisés

On constate que tous les élèves qui donnent ce type de prévision pour l'exercice 3 ou l'exercice 5 donnent une prévision correcte pour l'exercice 4. Mais la réciproque n'est pas vraie, c'est-à-dire qu'il existe quelques élèves qui donnent une prévision correcte pour l'exercice 4 et une réponse de type R dans l'un des deux autres exercices. On peut soupçonner qu'ils ont une conception "miroir" mais que les difficultés de l'écriture de valeurs de variables ou du traitement des listes les font retourner à une conception plus primitive, n'incluant pas l'aspect suspensif.

3 Caractéristiques des situations expliquant ces différences

La différence de réussite entre les trois exercices s'explique par la conjonction de caractéristiques qui rendent l'exercice 4 plus facile que les deux autres :

- l'analogie avec des exemples programmés auparavant,

- **un modèle global "miroir" permet des prévisions correctes,** contrairement à ce qui se passe pour les deux autres exercices. En effet, l'exercice 4 est le seul des trois à être **"symétrique".** Certes, le dessin A et le dessin B ne sont pas identiques mais ils sont de même taille N. Dans l'exercice 5, par exemple, la "symétrie" a été cassée par le fait que l'on demande ECRIS : N avant l'appel et ECRIS : N-1 après l'appel. Les erreurs observées pour l'exercice 5 permettent de supposer que l'exercice 4 n'a pas été mieux réussi uniquement parce qu'il était "graphique" mais plutôt parce que **la gestion des valeurs de la variable y était beaucoup plus simple.**

Conclusion

L'objectif de cet enseignement, qui pouvait apparaître trop ambitieux avec des élèves de collège, a été atteint :

Durant la phase d'écriture de programmes récursifs, l'écriture récursive des familles de courbes a été bien réussie. De plus, tous les élèves ont été capables d'écrire au moins une procédure récursive correcte pour un "projet", en interaction avec l'ordinateur.

Cependant, l'étude de leurs prévisions d'exécution montre une grande diversité d'erreurs (si l'on considère l'ensemble des exercices que nous leur avions proposés), des procédures incorrectes et des erreurs instables pour la plupart d'entre eux, l'accroissement des difficultés induisant souvent un **retour** à des **conceptions antérieures** :

- Les exercices les mieux réussis concernent la **récursivité terminale**, l'un d'entre eux étant même réussi par tous les élèves. La **conception spontanée de l'auto-référence**, où l'appel récursif déclenche le retour au début du programme, permet une prévision d'exécution correcte pour les programmes en récursivité terminale.

- En ce qui concerne les prévisions d'exécution en **récursivité centrale**, tout se passe comme si, lors de difficultés, les élèves revenaient à des conceptions **antérieures ou plus primitives** : la modalité DAT (Deux Appels Terminaux), par exemple, peut être vue comme une tentative de se ramener à deux sous-problèmes de récursivité terminale. De même, dans les exercices sur les listes, on voit ressurgir des conceptions antérieures à tout enseignement de la récursivité. Ce phénomène de retour à une conception antérieure "dans la difficulté" a déjà été observé dans l'expérimentation précédente (Dupuis, Egret, Guin, 1987) .

Nous avons constaté que les élèves se dégagent **progressivement** d'une représentation d'une procédure exclusivement en termes d'exécution, et accèdent à une **représentation statique** nécessaire à l'écriture récursive au fur et à mesure qu'ils intègrent les caractéristiques du traitement par le dispositif informatique. Toutefois, une représentation même erronée, tel le modèle global "miroir", est suffisante pour écrire correctement bon nombre de procédures récursives (possédant une certaine **symétrie**).

Ainsi, si l'objectif de l'enseignement est une bonne compréhension du **traitement** d'une procédure récursive par le dispositif informatique, l'enseignement devra être modifié de manière à casser cette conception fausse : il faudra construire des situations spécifiques contraignant les élèves à modifier leur modèle. Par contre, si l'objectif est plutôt l'écriture récursive, on peut émettre l'hypothèse qu'une conception **partielle**, même erronée, est suffisante dans un premier temps.

Enfin, il est important de souligner qu'un tel enseignement ne peut être abordé que s'il a été précédé d'une **alphabétisation informatique** solide développant à la fois des **compétences de programmation** graphique et traitement de listes impliquant :

- l'**appréhension opératoire** des figures,
- la **structuration** ou **modularité** du travail (décomposition du problème),
- la gestion des **variables**,

Ces compétences, qui contribuent au développement intellectuel, seront consolidées par un enseignement de plus haut niveau comme celui de la récursivité. Elles pourront être alors réinvesties en dehors du **contexte** de la **programmation**, et plus particulièrement en Mathématiques. Par exemple, la représentation statique nécessaire à l'écriture récursive pourra être une préparation à l'étude des suites numériques récurrentes.

Il est clair qu'un enseignement de la récursivité présentant les caractéristiques développées précédemment n'est envisageable que si l'on dispose d'horaires suffisamment importants pour pouvoir proposer une diversité de situations de difficulté croissante : on ne peut espérer obtenir des résultats analogues à ceux que nous avons présentés en travaillant pendant quelques heures facultatives au sein d'un club informatique.

REFERENCES

DUPUIS C., EGRET M.- A., GUIN D. (1985) : Récursivité et Logo - 1 : Préexpérimentation, Brochure I.R.E.M de Strasbourg .

DUPUIS C., EGRET M.-A., GUIN D. (1987) : Logo 3. Programmation structurée : Présentation et Analyse de situations, Brochure I.R.E.M. de Strasbourg .

DUPUIS C., EGRET M.-A., GUIN D. (1988-1) : Pour une analyse multi-critères de programmation en Logo, Annales de Didactique et de Sciences Cognitives, vol .1, pp 111-130, IREM de Strasbourg.

DUPUIS C., EGRET M.-A., GUIN D. (1988-2) : Présentation et analyse d'activités de programmation en LOGO, Petit X, n° 18, pp. 47-69, IREM de Grenoble .

DUPUIS C., EGRET M.-A., GUIN D. (1990) : Une expérience d'enseignement de la récursivité en LOGO, Annales de Didactique et de Sciences Cognitives, vol .3, pp.143-162, IREM de Strasbourg.

DUPUIS C. , GUIN D. (1988) : " Découverte de la récursivité en LOGO dans une classe ", Actes du premier colloque franco-allemand de Didactique des Mathématiques et de l'Informatique, La Pensée Sauvage, F-38002 Grenoble. pp 267 - 274

DUPUIS C., GUIN D. (1989) : Gestion des relations entre variables dans un environnement de programmation LOGO, Information Technology and Mathematics Education, Educational Studies in Mathematics, vol 2 n° 3, pp. 293-316.

EGRET M.-A., GUIN D. (1990) : Une introduction de la notion de fonction utilisant le langage LOGO, bulletin de l' EPI , n° 57.

GUIN D., HELM J.G. (1984) : Logo 2. Rapport d'expérimentation en CM2, Brochure IREM de Strasbourg .

BIBLIOGRAPHIE

DUVAL R. (1988) : Approche cognitive des problèmes de géométrie en termes de congruence, Annales de Didactique et de Sciences Cognitives,vol. 1, pp. 57-74, IREM de Strasbourg.

ROGALSKI J. (1986) : Pour une pédagogie de l'informatique, E P I, vol. 42, pp.105-109.

ROGALSKI J. , VERGNAUD G. (1987) : Didactique de l'informatique et acquisitions cognitives en programmation, Psychologie Française, vol. 32-4, pp. 267-273.

Table des matières