Tools for Learning —
Insights for the Mathematics Educator from a Logo Programming Environment

CELIA HOYLES

Background
Researchers in mathematics education have long realised the wealth of insights that can be gleaned from detailed observation of children involved in mathematical activity. Interpretation of pupil behaviour in such studies, even from a basis of a well considered theoretical framework is, however, often problematic, given that the processes used by pupils whilst undertaking mathematical problems are not necessarily obvious. During the Logo Maths Project [Note 1] observation of pupils in collaboration with a partner in an exploratory Logo programming environment has been found to provide a very rich research environment for the study of the processes by which pupils grapple with problems. Pupil’s thinking is made accessible to the researcher through planning work and the way the Logo programs are constructed and used.

A powerful way for pupils to learn mathematics is for them to become involved in mathematical activity which is functional; that is activity in which the power of mathematics is used as a means to achieve a desired outcome (which may or may not be ‘useful’ in any broad sense). We know that individuals can apply mathematical ideas and operations as tools in situations which are meaningful to them in the restructuring of knowledge from its initial basis within the activity. A transition has therefore to be made to raise these mathematical structures to a “plane of conscious awareness” [Thom 1963]. This is a fundamental issue in mathematics education and one which is to be addressed in this paper.

With this background in mind, a model for learning mathematics is proposed (abbreviated to the UDGS) which involves the dynamically related components of Using, Discriminating, Generalising and Synthesising. The essence of the model is: first use mathematics in situations in which its power is appreciated, although there might, at this stage, be little more than a syntactical or procedural level of understanding of what is being used; after successful use (successful in terms of achievement of desired goals) learners can then be provoked to rewrite in explicit form what was previously implicit and break down and reflect upon procedures that previously only operated in totalities. They will also then have the confidence to apply structures in new domains and make links with other procedures they have learned.

The components of the model are as follows:

USING, where a concept is used as a tool for functional purposes to achieve particular goals.

DISCRIMINATING, where the different parts of the structure of a concept used as a tool are progressively made explicit.

GENERALISING, where the range of application of the concept used as a tool is consciously extended.

SYNTHESISING, where the range of application of the concept used as a tool is consciously integrated with other contexts of application; that is, where multiple representations of the same knowledge in different symbolic codes derived from different domains are reformulated into an integral whole.

Movement between all the parts of the UDGS model is achieved through the strategies of conjecturing, testing and debugging, although the semantic level at which pupils are prepared to engage in the activity is subject to cognitive, social and affective influences. Firstly, pupils have to be “cognitively ready” to move semantic level; that is, the move must be within their zone of proximal development [Vygotsky 1962 p. 102]. Secondly, they must perceive the need for any change in their use and perception of their mathematical tools (in terms of other desired outcomes). Thirdly, they must be ready emotionally; that is, they must be able to transcend any inhibitions built up from previous experiences and not be impeded by competitiveness or fear of failure.

Experimentation and willingness to try out ideas is at the heart of the UDGS model for learning mathematics. It fits particularly well in an interactive computer context which both engages a learners’ actions and provides informative feedback. In such an environment and together with teacher intervention a learner is helped to reflect upon structures used; that is transcend the gap between an understanding of the syntactic aspects of actions and their logical bases. The computer environment is therefore regarded as a special learning situation which involves the dynamically related components of Using, Discriminating, Generalising and Synthesising. The essence of the model is: first use mathematics in situations in which its power is appreciated, although there might, at this stage, be little more than a syntactical or procedural level of understanding of what is being used; after successful use (successful in terms of achievement of desired goals) learners can then be provoked to rewrite in explicit form what was previously implicit and break down and reflect upon procedures that previously only operated in totalities. They will also then have the confidence to apply structures in new domains and make links with other procedures they have learned.

The components of the model are as follows:

USING, where a concept is used as a tool for functional purposes to achieve particular goals.

DISCRIMINATING, where the different parts of the structure of a concept used as a tool are progressively made explicit.

GENERALISING, where the range of application of the concept used as a tool is consciously extended.

SYNTHESISING, where the range of application of the concept used as a tool is consciously integrated with other contexts of application; that is, where multiple representations of the same knowledge in different symbolic codes derived from different domains are reformulated into an integral whole.

Movement between all the parts of the UDGS model is achieved through the strategies of conjecturing, testing and debugging, although the semantic level at which pupils are prepared to engage in the activity is subject to cognitive, social and affective influences. Firstly, pupils have to be “cognitively ready” to move semantic level; that is, the move must be within their zone of proximal development [Vygotsky 1962 p. 102]. Secondly, they must perceive the need for any change in their use and perception of their mathematical tools (in terms of other desired outcomes). Thirdly, they must be ready emotionally; that is, they must be able to transcend any inhibitions built up from previous experiences and not be impeded by competitiveness or fear of failure.

Experimentation and willingness to try out ideas is at the heart of the UDGS model for learning mathematics. It fits particularly well in an interactive computer context which both engages a learners’ actions and provides informative feedback. In such an environment and together with teacher intervention a learner is helped to reflect upon structures used; that is transcend the gap between an understanding of the syntactic aspects of actions and their logical bases. The computer environment is therefore regarded as a special learning situation which involves the dynamically related components of Using, Discriminating, Generalising and Synthesising. The essence of the model is: first use mathematics in situations in which its power is appreciated, although there might, at this stage, be little more than a syntactical or procedural level of understanding of what is being used; after successful use (successful in terms of achievement of desired goals) learners can then be provoked to rewrite in explicit form what was previously implicit and break down and reflect upon procedures that previously only operated in totalities. They will also then have the confidence to apply structures in new domains and make links with other procedures they have learned.

The components of the model are as follows:

USING, where a concept is used as a tool for functional purposes to achieve particular goals.

DISCRIMINATING, where the different parts of the structure of a concept used as a tool are progressively made explicit.

GENERALISING, where the range of application of the concept used as a tool is consciously extended.

SYNTHESISING, where the range of application of the concept used as a tool is consciously integrated with other contexts of application; that is, where multiple representations of the same knowledge in different symbolic codes derived from different domains are reformulated into an integral whole.

Movement between all the parts of the UDGS model is achieved through the strategies of conjecturing, testing and debugging, although the semantic level at which pupils are prepared to engage in the activity is subject to cognitive, social and affective influences. Firstly, pupils have to be “cognitively ready” to move semantic level; that is, the move must be within their zone of proximal development [Vygotsky 1962 p. 102]. Secondly, they must perceive the need for any change in their use and perception of their mathematical tools (in terms of other desired outcomes). Thirdly, they must be ready emotionally; that is, they must be able to transcend any inhibitions built up from previous experiences and not be impeded by competitiveness or fear of failure.

Experimentation and willingness to try out ideas is at the heart of the UDGS model for learning mathematics. It fits particularly well in an interactive computer context which both engages a learners’ actions and provides informative feedback. In such an environment and together with teacher intervention a learner is helped to reflect upon structures used; that is transcend the gap between an understanding of the syntactic aspects of actions and their logical bases. The computer environment is therefore regarded as a special learning situation which involves the dynamically related components of Using, Discriminating, Generalising and Synthesising. The essence of the model is: first use mathematics in situations in which its power is appreciated, although there might, at this stage, be little more than a syntactical or procedural level of understanding of what is being used; after successful use (successful in terms of achievement of desired goals) learners can then be provoked to rewrite in explicit form what was previously implicit and break down and reflect upon procedures that previously only operated in totalities. They will also then have the confidence to apply structures in new domains and make links with other procedures they have learned.

The components of the model are as follows:

USING, where a concept is used as a tool for functional purposes to achieve particular goals.

DISCRIMINATING, where the different parts of the structure of a concept used as a tool are progressively made explicit.

GENERALISING, where the range of application of the concept used as a tool is consciously extended.

SYNTHESISING, where the range of application of the concept used as a tool is consciously integrated with other contexts of application; that is, where multiple representations of the same knowledge in different symbolic codes derived from different domains are reformulated into an integral whole.

Movement between all the parts of the UDGS model is achieved through the strategies of conjecturing, testing and debugging, although the semantic level at which pupils are prepared to engage in the activity is subject to cognitive, social and affective influences. Firstly, pupils have to be “cognitively ready” to move semantic level; that is, the move must be within their zone of proximal development [Vygotsky 1962 p. 102]. Secondly, they must perceive the need for any change in their use and perception of their mathematical tools (in terms of other desired outcomes). Thirdly, they must be ready emotionally; that is, they must be able to transcend any inhibitions built up from previous experiences and not be impeded by competitiveness or fear of failure.

Experimentation and willingness to try out ideas is at the heart of the UDGS model for learning mathematics. It fits particularly well in an interactive computer context which both engages a learners’ actions and provides informative feedback. In such an environment and together with teacher intervention a learner is helped to reflect upon structures used; that is transcend the gap between an understanding of the syntactic aspects of actions and their logical bases. The computer environment is therefore regarded as a special learning situation which involves the dynamically related components of Using, Discriminating, Generalising and Synthesising. The essence of the model is: first use mathematics in situations in which its power is appreciated, although there might, at this stage, be little more than a syntactical or procedural level of understanding of what is being used; after successful use (successful in terms of achievement of desired goals) learners can then be provoked to rewrite in explicit form what was previously implicit and break down and reflect upon procedures that previously only operated in totalities. They will also then have the confidence to apply structures in new domains and make links with other procedures they have learned.

The components of the model are as follows:

USING, where a concept is used as a tool for functional purposes to achieve particular goals.

DISCRIMINATING, where the different parts of the structure of a concept used as a tool are progressively made explicit.

GENERALISING, where the range of application of the concept used as a tool is consciously extended.

SYNTHESISING, where the range of application of the concept used as a tool is consciously integrated with other contexts of application; that is, where multiple representations of the same knowledge in different symbolic codes derived from different domains are reformulated into an integral whole.
The focus of this paper is to describe the way pupils use Logo programs as tools within their projects and how the nature of their programming tools become more explicitly understood and generalised. These descriptions are used as illustration of the Using and Discriminating components of the learning model (the Generalising and Synthesising components are considered elsewhere, for example Hoyles and Sutherland [1985a and 1986b]). Hillel and Samurçay [1985] distinguish general procedures which operate on a variable (or variables) from fixed procedures which operate on particular values. They also refer to a procedure made up only of primitives as simple as compared to a composed procedure which contains several sub-procedures. In the interests of consistency these terms will be used here.

Using and discriminating simple general procedures
When children first start to use simple general procedures as tools to work with (as opposed to perceiving them merely as descriptions of their work) they tend to operate with them in a way similar to their earlier use of the primitives of the language (FD, BK, RT, LT). Essentially, the focus of the activity at this stage is the product or outcome of the procedure and the procedures have the status of "cognitive tools" for successfully reaching a goal. Inputs to the procedure are perceived as a collection of constants, that is as a set of numbers which produces a set of results. Any procedure can only be called upon to run in its totality and its components cannot be accessed and operated upon separately. This is an important early stage of children's use of Logo programs. However, simultaneously with such goal directed work, children engage in what we have termed "making sense of activity" [Hoyles, Sutherland & Evans 1985a] in which the process by which the program is built is explored in direct mode. The Logo programs themselves therefore become the focus of the learners' activity and the ways these programs are represented internally serve as "cognitive units of attention" [Karmiloff-Smith 1984].

Using and discriminating general composed procedures
When children first start to build general composed procedures from general simple procedures they do so in an additive manner: that is, each new subprocedure together with its input is merely added on to the previous collection.

Children introduce a new variable for every new object that varies within their task, rather than work out how to operate on any existing variables using the relationships intrinsic to the task. By this means the children retain control over their programs at the level of action since they, as opposed to the computer, decide the value of each input. This direct control over the input values allows learners to achieve successful outcomes after which they are able to reflect upon the way the outcomes were achieved. The additive strategy and how it is used and represented is illustrated in the extract below:

Julie and Nicola were involved in the task of building a procedure for a chessboard. They built a state transparent fixed procedure SQUARE for their empty squares and after considerable trial and error build a procedure COLOUR for their "filled in squares" as follows (see Figure 1):

```
TO SQUARE
  REPEAT 4 [FD 48 RT 90]
END
TO COLOUR
  REPEAT 12 [FD 48 RT 90 FD 2 RT 90 FD 48 LT 90 FD 2 LT 90]
END
```

The main obstacle in the process of building COLOUR was working out how many REPEATS were needed. The girls eventually worked out that they had to divide the length of the side of the square by 4, but then came up against the problem of a decimal input to REPEAT because their original square side was 50. They therefore changed 50 to 48 as above. The pair then used these two procedures in a modular way to produce the two different lines of the chessboard LINE and LINE1 as follows (see Figures 1, 2):

```
TO LINE
  REPEAT 4 [SQUARE COLOUR SQUARE RT 90 FD 48 LT 90]
END
```
A move between these two procedures was worked out in direct drive. It was clear that the pair knew the relationship of the distances involved in the move to the side of the squares as the turtle was first moved back a distance of 48, turned, and then moved forward a distance 384 (the calculation 8 × 48 being done on a calculator). This move was built into a procedure M as follows:

```
TO M
BK 48
LT 90
FD 384
RT 90
END
```

Finally Nicola and Julie put all their procedures together into a program for a chessboard called ALL (see Figure 3):

```
TO ALL
REPEAT 3 [LINE M LINE1 M]
LINE M LINE1
END
```

It was then suggested to the pair that they make a “general sized” chessboard, which led to the immediate response:

Nicola: “Oh yes, we need an input!”

They edited their SQUARE and COLOUR procedures introducing inputs as follows:

```
TO SQUARE :S
REPEAT 4 [RD :S RT 90]
TO COLOUR :C
REPEAT 12 [FD :C RT 90 FD 2 RT 90 FD :CLT 90 FD 2 LT 90]
END
```

They then started to edit their LINE procedure. They added :S and :C to the title line (and after the calls of SQUARE and COLOUR respectively). This illustrates their additive strategy. It did not occur to them to use the same input for both “types” of square length. At this point Julie has an insight:

Julie: “What about the number of REPEATS (in COLOUR)? We need another input.”

She knew that the 12 REPEATS in COLOUR had to change; she certainly knew that 12 had been used originally as it was a quarter of 48, but she still wanted to use an input with another name rather than operate on her already assigned “unknown” C. I intervened at this stage in order to find out whether the girls might be able to more consciously conceptualise the relationship between the number of REPEATS and the value of C.

Intervention: “Isn’t there a relationship between the number of REPEATS and the value of C?”

Nicola: “Oh, oh, into 360”

(This was a guess arising from experience with the relationship used in work with regular polygons.)

Julie: “Halve it”

This response triggered an idea from Nicola
Nicola: “I know, halve it twice!”

The pair then edited their COLOUR procedure to:

```lisp
TO COLOUR :C
  REPEAT :C/4 [FD :C R 90]
END
```

This was the first time that they had operated on inputs in this way. It is clear that they would not have moved away from their additive strategy without the specific intervention. It did, however, have consequences later when they came to edit their move procedure. They introduced a variable :MOV and typed:

```lisp
TO M :MOV
  BK :MOV
  LT 90
```

At this point they paused:

Nicola: “How far forward? I know it’s 8 times the square!”

They did not therefore introduce a new variable for the distance to be moved forward. Interestingly enough they changed their variable name to S in M. This seemed to suggest that the actual name of the variable was still perceived to have some significance. Their procedure M was then as follows:

```lisp
TO M :S
  BK :S
  LT 90
  FD :S *8
  LT 90
END
```

This was the first time that the pair had spontaneously operated on an input inside a procedure and not used an additive strategy. Their progression could be interpreted as following on from the previous discussion concerned with dividing :C by 4 and also the conscious way the calculation of $8 \times 48$ had been undertaken in the first instance.

Finally the girls edited ALL1 and after missing out several instances of :S and debugging these mistakes, they ended up with the following procedure:

```lisp
TO ALL1 :S :C
  REPEAT 3 [LINE :S :C M :S LINE1 :S :C M :S]
  LINE :S :C M :S LINE1 :S :C
END
```

They then triumphantly asked other children:

“What size squares do you want?”

They typed in the number given, twice after ALL1, for example ALL1 100 100; ALL1 200 200. As they used ALL1, they knew that the two inputs had to be the same. However, when I intervened as follows:

Intervention: “Couldn’t you modify your procedure so that you don’t have to type that twice?”

The girls just did not want to reflect further:

Julie: “Hey miss, can’t we just leave it for the moment?”

However, another rather more successful intervention was made later by another teacher:

Intervention: “Can’t you put your chessboard in the middle of the screen?”

Putting LINE and LINE1 together to make ALL1.

Figure 3
Nicola and Julie thought that was a good idea. They immediately typed \texttt{HT I 190 FD:S*4 RT 90} and built these commands into a procedure. They now seemed quite happy to perform a linear operation on an input.

The following session, Julie was on her own. She was asked to make a “really general” chessboard, that is with any number of squares. First of all she decided to make a $10 	imes 10$ chessboard and systematically recorded changes that had to be made to all the procedures. She changed \texttt{FD:S \times 8} to \texttt{FD:S \times 10} in \texttt{M} and changed the number of \texttt{REPEATS} in \texttt{LINE} and \texttt{LINE1} from 4 to 5. When \texttt{ALLL} was tried out there was a bug since it only had 8 rows. The number of \texttt{REPEATS} in \texttt{ALLL} was then altered from 3 to 4.

\textbf{Intervention:} “That’s great. How about another size chessboard?”

Julie: “You know all those changes I made (she had kept a careful record) I’ll just put in an input for them!”

Julie saw the input as \textit{a way of collecting together all the changes that had to be made but individually and related to each specific variable} as can be seen from what followed. She introduced a variable \texttt{CH} for the number of \texttt{REPEATS} in \texttt{LINE} and \texttt{LINE1} and a variable \texttt{M} for the number which multiplied \texttt{S} in her move procedure. When she came to edit \texttt{ALLL} she still had to change the number of \texttt{REPEATS} and continued with her additive strategy:

Julie: “It’s different so I’ll put in a \texttt{D}.”

The final procedure for \texttt{ALLL} was as follows:

\begin{verbatim}
TO ALLL :S :C :CH :D :M
   LINE :S :C :CH M :S :M LINE1 :S :C :CH
END
\end{verbatim}

where \texttt{S} and \texttt{C} were both names for the lengths of the sides of the squares, \texttt{M} was the name for the number of squares along a side of the chessboard, \texttt{CH} the number of \texttt{REPEATS} of \texttt{SQUARE} and \texttt{COLOUR} (with value of half of \texttt{M}) and \texttt{D} the number of \texttt{REPEATS} in \texttt{ALLL}, (that is the value of \texttt{CH} minus 1).

Julie was able to use these quite complicated relationships between her inputs. She confidently asked:

Julie: “What size squares do you want and how many? Your size of square has to divide by 4.”

When given the numbers 32 and 10 she took pencil and paper and worked out that the inputs to \texttt{ALLL} had to be 32, 32, 5, 4 and 10.

\textbf{Intervention:} “How did you get all those numbers?”

Julie: “Well this one is the same as that (pointing to 32). This one is half (pointing to 5) and I take 1 off for the next!”

She spent the rest of the lesson using \texttt{ALLL} and making many different sized chessboards!

In considering this extract in the light of the UDGS model, it is apparent that children, while using their simple general procedures to build up general composed procedures, tend at first to see the inputs as related to the \textit{physical representations within their design}. New inputs are therefore introduced for every new variable “object” encountered. This does not of course imply that the children are completely unaware of the relationship between the inputs, as can be observed from their use of the composed procedures. Julie and Nicola, for example, having constructed a procedure for the $8 	imes 8$ chessboard knew the two inputs had to be the same. However this relationship between particular sets of numbers is not made explicit in any formal sense. The girls’ behaviour whilst working on the chessboard problem was constrained implicitly by their underlying representation of that problem, but this representation was not at this stage accessible to either girl explicitly. It is suggested that after successfully using their chessboard procedure, Julie and Nicola can then pass into a new phase of discrimination in which explicitly defined links are established across their previously independently represented inputs. At this point a didactical intervention is needed to suggest how this relationship can be made explicit within their Logo programs. Discrimination, it is therefore suggested, occurs after many opportunities to try out, use and make sense of additively composed general procedures. Such use allows the pupils to experience the relationship between the inputs which produces a required output. They experience procedural success and then go beyond this success to try to understand why the success was achieved: that is, reflect upon what was previously implicit. In so doing learners come to focus on the values of their inputs rather than the “objects” to which they refer. It is however possible that the need for this next stage of generalisation does not really occur until the pupils’ projects require that their additive general procedures serve as a means to reach a goal rather than as goals in themselves: that is, their additive general procedures be used as sub-procedures within another procedure. In such circumstances the limitation of the additive general procedure will become immediately apparent.

\textbf{Summary}

It has been argued that children using general Logo programs, both simple and composed, first focus on the outcome of their programs and consider the computer code merely as a sequence of instructions which can only run in its totality. However, during such use, children begin to break down their programs into what they come to see as semantic components. In addition the computer environment allows pupils, who have built what have been termed general composed procedures, to use these procedures in a way in which the relationship between inputs is implicit (using the additive strategy described) and to perceive the “concrete” outcomes of this use. They are then able to develop a “theory in action” in which there is a focus on the relationship between the values of the inputs rather than what the inputs signify. This relationship might later be “captured” in the formalisation of the program itself if the overall plan appears to need this form of generalisation. This latter development involves operating on the “unknown” quantities and on explicit awareness of a consistent relationship between variables. These stages would seem to correspond to a developing understanding of variable (Note 2).
How far children continuously reflect upon the products of their work is however problematic and subject to social pressure, as the final extract illustrates:

Nicola and Julia had built a sophisticated program ALLL with five inputs for a chessboard with a variable number of squares of variable size as described earlier. During the following sessions, the girls used their ALLL procedure working out each time the “correct” relationship between the five inputs. They explained very confidently the meaning of each of the inputs as follows:

“The first two are the squares (meaning the size of the squares) and you take another number for the next one, double it for the last and take one off here.”

Then came the surprise. I asked to look at the procedure ALLL again and was taken aback to see that it only had two inputs! It was in fact the first ALLL procedure (given on P 8) which drew an 8 X 8 chessboard only. The pair had not managed to save their new, more general ALLL procedure due to a fault in the computer. They had therefore been diligently typing in five inputs to ALLL for the whole of the previous session, but the last three inputs could have had no effect on screen output — and they had not noticed it! Even when the procedure came up in the editor with just :S and :C in the title line, Nicola was neither puzzled nor concerned.

In the above extract, the procedure ALLL, although built by the girls themselves, was perceived as something which required a certain number and type of input whose connection with the object produced had been at least partially lost. What we should have done in this situation was to provoke a conflict by asking the girls to draw a 12 X 12 chessboard. Undoubtedly they would have worked out the “correct” five inputs but would still only have obtained an 8 X 8 board. They then are very likely to have investigated what had happened! Unfortunately we did not think of this until too late. We had already planned our agenda for the Logo activity and missed the opportunity. This perhaps is another lesson for researchers to learn!

Notes
1 For further details of the Logo Maths Project see Hoyles Sutherland and Evans 1985a and 1985b
2 It will be important here to make some synthesis of the conceptual demand of these stages with reference to the work of Küchemann [1980], and Booth [1984], for example. How children’s activity in the Logo environment might “transfer” to non-computational contexts is at present being investigated by R. Sutherland

References
Hoyles, C [1985] Culture and computers in the mathematics classroom University of London Institute of Education Bedford Way Paper
Hoyles, C., Sutherland, R. and Evans, J [1985a] The Logo Maths Project Interim Report to the Leverhulme Trust
Hoyles, C, Sutherland, R. and Evans, J [1985b] The LOGO Maths Project, 2nd Year Report to the Leverhulme Trust
Vergnaud, G [1982] Cognitive and development psychology and research in mathematical education: some theoretical and methodological issues. For the Learning of Mathematics Vol 3 No 2 p 31-41