THE COMPUTER IN SECONDARY SCHOOL MATHEMATICS: AN ANALYSIS AND CLASSIFICATION OF POSSIBLE MODES OF APPLICATION, WITH SUGGESTED IMPLICATIONS FOR THE MATHEMATICS CURRICULUM IN SOUTH AFRICA

A Thesis

Submitted in Fulfilment

of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in the Faculty of Education of Rhodes University

by

TERENCE ANTHONY MARSH

December 1990

"The challenge of the computer revolution [in education] rests squarely at the door of the mathematics classroom."

(Fey and Heid, 1984)

"Never in the history of mathematics education has any development opened up such a vast range of new possibilities and challenges to the educator as has the microcomputer."

(Howson and Kahane (eds.), 1986a)

"... the computer ... will affect not only the mathematics we study, but also the way in which we do, teach and learn mathematics."

(Howson, 1985)

ACKNOWLEDGEMENTS

I wish to record my sincere thanks to:

- Professor Piet Human, Department of Didactics, Stellenbosch University, for the inspiration he provided at various stages of the project;

- Emeritus Professor Arthur Noble, Rhodes University, for his encouragement and help, particularly in the final stages of the write-up and submission for examination;

- Professor Graham Oberem, Director of the Computer Based Education Unit, Rhodes University, for his helpful comments on the final draft of the thesis;

- Professor Nic Heideman, Department of Mathematics, Rhodes University, for showing some interest in the project;

- Professors Denis Riordan and Alan Penny, ex Rhodes, now in Canada and England respectively, for their support at the inception of the project;

- The Human Sciences Research Council of South Africa for financial support in the form of a bursary for part-time study;

- My wife, Cecille, our children, Hadleigh, Damian and Camilla, and my parents for their unfailing interest and support in so many ways.

ABSTRACT

There is a variety of possible ways in which computers can be used to enhance mathematics education. This thesis attempts to identify, analyse and classify these possibilities, particularly at the secondary school level. It describes and exemplifies applications ranging from drill-and-practice through games and simulations to problem solving by computer programming. Software evaluation procedures are considered in some depth. Illuminative evaluations of various items of software and there classroom use are reported. The underlying methodology is small-scale action research.

Insights gained during the process of investigating each class of software lead to the eventual formulation of a scheme for classifying mathematics education software by means of 'multi-dimensional attributes'. It is contended that this scheme will help mathematics teachers to make well informed and sound professional judgements regarding the evaluation and use of computer programs for teaching/learning purposes. Also, it is hoped that this scheme and the thesis as a whole will contribute towards the establishment of well founded standards and procedures for software development in the field of mathematics education.

Several implications of the computer for the mathematics curriculum in South Africa are suggested. A note of caution is sounded regarding possible detrimental effects of the computer and several questions requiring further research are posed.

A recommendation arising from the thesis is that in-service training courses concerning computer applications in mathematics education should be run for secondary school teachers.

CONTENTS

CHAPTER INDEX

TABLE INDEX

FIGURE INDEX

PROGRAM INDEX

The names of programs and/or systems referred to in this thesis, their authors or publishers, and the pages on which they appear are listed below:

ADVISOR	Genesereth	47
ANGLE360	SMILE, Centre for Learning Resources, London	23,35,39,40
ARITHSUM	Burrell and Marsh	145
AUTOFRAC	ITMA, College of St Mark and St John, Plymouth, England	35,37,168,169,170
BOXED	SMILE, Centre for Learning Resources, London	23,92
BUGGY	Brown and Burton	47
CIRCLE	Anonymous	191
DARTS	SMILE, Centre for Learning Resources, London	23,92,93
DEFINE	SMILE, Centre for Learning Resources, London	23,35,40
DICECOIN	ITMA, College of St Mark and St John, Plymouth, England	73,95
DIRECTED	ITMA, College of St Mark and St John, Plymouth, England	35,36,169,170
DRAWSIN	Burrell and Marsh	147
DRAWTAN	Burrell and Marsh	148
EQUATORS	NASOU Beperk, Box 5197, Cape Town	73,79,168,169

ERGO	ITMA, College of St Mark and St John, Plymouth, England	23,73,74
EUREKA	ITMA, College of St Mark and St John, Plymouth, England	23,73,95,96
FRAC	ITMA, College of St Mark and St John, Plymouth, England	23,35,38,169
GEOMETRIC SUPPOSER:	Sunburst Communications, Pleasantville, New York	
TRIANGLES		124,125,184,185
QUADRILATERALS		124,125,184,185
CIRCLES		124,125,184,185
GEOMETRY I	WAMSOFT, WACAE, Perth, Australia	122,124
GEOMSUM	Burrell and Marsh	145
GRAPHIC CALCULUS	Tall	125,126
GREEN GLOBS	Dugdale and Kibbey	66,73,80,82,83,84,87, 88,169
IDENTIFY	SMILE, Centre for Learning Resources, London	23,35,40,41
INTEGRATE	Kimball	47
LINEOVER	SMILE, Centre for Learning Resources, London	23,73,89,90,91,92
LMS	Sleeman and Smith	47
MACSYMA	Massachusetts Institute of Technology	47
MATHCAD	MathSoft Inc., Cambridge, Massachusetts	184
MATH SEQUENCES 1980	Milliken Computer Courseware	35
MINIMAX	SMILE, Centre for Learning Resources, London	23,93

MONTY	SLIMWAM, The ATM, King's Chambers, Derby, England	23,73,89
muMATH	The Soft Warehouse, Honolulu, Hawaii	49,101,118,119,121,122 175,176,178,183
NUMBER MACHINE	NASOU Beperk, Box 5197, Cape Town	73,96,168,169
PARM	Anonymous	186
PC PLANNER	Sagesoft, Regent Centre, Newcastle upon Tyne, England	101,108,109,122
PIRATES	Cambridge Microsoft, Cambridge University Press	73,75,78
PLATO System	Control Data Corporation, Minnesota, USA	8,26,27,46
PROGRAM C1	Perkins and Perkins	127
QUADRATIC	O'Shea	47
QUEST	Robin Stead Assocs., Claremont, Cape Town	46
SASEC (microPLATO)	Control Data South Africa Sandton, Johannesburg	26,27,28,169
SEQUENCES SET	Baldwin	68
SERGO System	SERGO, Hatfield, Pretoria	8,22,26,28-30,169
SKETCHER	Marsh and Halpin	2,8,47,49,50,52,169
SPECIAL PARALLELO-GRAMS	Hativa	62
SQSPIRAL	Anonymous	155
SUBGAME	ITMA, College of St Mark and St John, Plymouth, England	23,73,75,76,77,93
SUMINV	Burrell and Marsh	146
SUMINVSQS	Burrell and Marsh	146

SUPER GRAPH	Tall	125
SuperPILOT	Apple Computer Inc., Cupertino, California	46
TAXI	SMILE, Centre for Learning Resources, London	23,94
TenCORE System	Computer Teaching Corporation, Champaign, Illinois, USA	46
TK!Solver	Miller	104,181,183
TOAM System	Centre for Educational Technology, Ramat Aviv, Israel	2,8,26,33,34,169
TRIANGLE	Anonymous	155
VECTORS:	Cambridge Microsoft,	
NAVIGATION	Cambridge University	23,73,78,79
TREASURE HUNT	Press	23,73,75,78,79
RACING DRIVER		23,73,78,79
WEST	Burton and Brown	47
WUSOR	Carr and Goldstein	47

CHAPTER 1 INTRODUCTION AND OVERVIEW

CHAPTER 1

INTRODUCTION AND OVERVIEW

1.1 Background to this Study

Computers can be programmed to perform a variety of educational tasks. These range from simple tasks like comparing answers to set questions with correct stored answers, to intellectually complex tasks like demonstrating strategies for the solution of problems of a certain class, these solutions and strategies being generated by an expert program. An example of a system which carries out simple tasks such as those mentioned above is the set of *TOAM* programs - see section 3.2.3. On the other hand, *SKETCHER* - see section 4.2.1 - is an example of a system which carries out the kind of complex tasks referred to above. The effective analysis, evaluation and usage of existing educational computer programs, as well as the generation of ideas for new programs, is dependent on the availability of an analytical framework delineating the variety of educational tasks that can be performed by computers and the ways in which these tasks can be performed. Existing frameworks like Taylor's (ed.) (1980) classical categories "Tutor, Tool, Tutee" are useful though too broad to illuminate many important differences between existing educational programs in a specialized field like mathematics.

This study involves an analysis of the possible modes of application of the computer as a tool for the enhancement of mathematics education at the secondary school level. An attempt is made to classify these modes of application into a conceptual framework to enable optimal use of the computer as an educational resource. Suggested implications for the mathematics curriculum in South Africa are also presented.

1.2 The Aim of this Study

The principal aim of this study is to contribute towards the development of an analytical framework of the educational tasks that computers may perform in the field of mathematics education. This is done by analyzing a variety of existing educational programs for mathematics - with emphasis on secondary school mathematics - both conceptually and by way of field-testing, in order to come to a sharp understanding of the educational tasks performed by these programs. The information generated in this way is then used as a basis, together with existing classificatory categories, to develop a multi-dimensional analytical framework.

The need for a refined analytical framework stems from the wealth of information-processing functions that can be performed by computers, and the variety of ways in which each of these functions can be utilized to promote learning. Without an analytical framework which categorizes this multiplicity of educational functions, teachers may have difficulty in coming to a valid appreciation of the educational merits, or an awareness of their lack, in a given program, and hence be unable to make informed decisions about its possible use. In the present situation where there is still fairly limited experience of computer usage in education, but (in South Africa at least) a distinct need for effectively augmenting the teaching capacity of a limited corps of available teachers, the absence of an illuminating frame of reference for analyzing available educational programs and designing new programs, may seriously inhibit the effective use of computers in education - mathematics education in particular.

In attempting to establish a frame of reference which can be used for computer program design, selection and usage in the field of secondary school mathematics, the major thrusts of the study will be towards:

/- identifying ...

- identifying and describing the various fundamentally different modes of using computers in mathematics education;

- identifying and describing possible educational uses of the various modes;

- developing standards for mathematical software design and evaluation;

- identifying and describing possible mathematics curriculum changes implied by the availability and use of computers.

There are then two main aspects to the study: the generative - broadening and sharpening perspectives on the use of the computer in mathematics education; and the evaluative - setting standards of quality for software production and selection.

Educators with some experience of teachers' reactions when they are confronted with given educational applications of computers know the frustration of having to deal with program appraisals based on narrow and prejudiced perceptions of what educational computer programs can or should do.

1.3 Identifying and Classifying Modes of Computer Use in Mathematics Education

A substantial portion of this study will be devoted to an attempt to classify and characterize a wide range of existing mathematical software, according to modes of use, into a conceptual framework. Small-scale action research case studies will form part of the strategy adopted for this purpose.

/A scheme ...

A scheme by which modes of computer use can be identified and classified is first introduced in a primitive way in section 1.3.1 below and then worked into a more refined model, a multi-dimensional scheme, in Chapter 9.

1.3.1 A Simple Linear Classification Scheme for Computer Programs Used in Mathematics Education

The approaches to educational computer program classification advocated by Kansky (1982) and Kemmis et al. (1977), about which more will be said in Chapter 9, give rise to a one-dimensional classification scheme which has played a part in determining the chapter divisions of this thesis. The categories in this scheme are:

- drill-and-practice;
- traditional computer assisted instruction (CAI);
- 'intelligent' tutoring;
- computer guided teaching (CGT);
- games;
- simulations;
- pre-programmed computational environments;
- programming.

Using this scheme it is easy to identify, amongst others, the following modes of computer-based mathematical activity:

(1) playing a computer game designed to develop skills and concepts;

(2) working through a computer simulation, designed to develop skills and concepts;

(3) working through a tutorial program designed to develop skills and concepts;

(4) solving a problem using a supplied program (a tool program) designed specifically for that purpose;

(5) solving a problem creatively with the aid of a supplied program designed as a general purpose computational environment, for example, a spreadsheet system;

(6) writing a program to solve a problem.

Detailed illustrations of each of these modes (and others) will be given. The potential educational usefulness of each of these modes of computer usage will also be considered.

A logical extension is to go further than the one-dimensional distinction of modes, by developing a multi-dimensional system of classification. As mentioned above, an attempt at doing this is presented in Chapter 9.

1.4 Computer Inspired Mathematics Learning

In striving to sharpen perspectives on computer use in mathematics education, we shall also need to consider the computer as an agent for provoking the pupil to solve mathematical problems. This mode of computer use is well demonstrated by, for example, the simple challenge to use Logo to draw a circle with prescribed radius. Writing the program is a straightforward matter, once the mathematical problem of finding the relationship between turtle step ('arc length') and radius has been solved (using the fact that the circle has constant curvature).

In this context the computer has two obvious functions:

- motivating problem solving by pupils;

- augmenting the capacities of its users - the teacher and the pupils.

As a motivating agent the computer provides a dynamic medium for pupils to work with. It can be instrumental in generating interesting mathematical problems and it can promote in a natural, meaningful way the desirable mathematical attributes of explicitness, exactness and rigour.

Concerning the question of rigour, Lawler (1985) claims that certain Logo programs can provide a powerful environment in which young children can come to terms with the advanced notion of functions of several variables and begin to appreciate the need, at their own level, for a formal, rigorous treatment of these concepts. Noss (1986) in commenting on this claim makes the point that:

> **"Such findings (see also Lawler, 1985, for a stronger developmental claim) seem to depend on the provision of a context in which the construction of children's own formalism becomes a necessary (and natural) component of the environment itself (Papert, 1975), rather than a mere agreement to play the game of mathematical rigour according to the whim of the teacher."**

As a capacity augmenting agent the computer can release both pupil and teacher from drudgery, setting them free to concentrate on more stimulating and demanding tasks. For example, by having the computer carry out whatever tedious computations are required, the

/pupil ...

pupil is set free to concentrate on problem solving. Moreover, by having the computer evaluate the pupils' answers, the teacher is set free to teach!

More and more emphasis is being placed on pupil-centred exploration and discovery learning in mathematics education worldwide. The importance of this approach is certainly being recognized in syllabus planning initiatives in this country (DEC Draft Document on Proposed New Syllabus, 1989).

The computer can help the process of exploration and discovery in several ways. Some examples will be given in this thesis. A comprehensive list is given by Howson and Kahane (1986a), drawing on Murakami and Hata (1985, 1986) and also on Tall (1985b, 1985c).

1.5 Software Resources for Mathematics Education

Suitable software resources for the exploration of mathematical concepts will be identified and characterized.

Three large-scale traditional CAI systems will be described and partially evaluated. These are *PLATO*, *TOAM* and *SERGO*. In addition, amongst others, numerous small programs developed by the ITMA and SMILE organizations in Britain (see reference list) will be dealt with. These small programs will provide examples of a range of application categories, including drill-and-practice, games and simulations. Also to be considered in some depth are the computational environments offered by electronic spreadsheets and computer algebra systems. The 'intelligent' tutoring category will be exemplified by *SKETCHER*, a program developed by the author and a colleague for rational function curve sketching. The programming languages which will receive attention are Logo, BASIC, Pascal and Prolog.

1.6 Comments on Research Methodology

According to Cohen and Manion (1986), action research is concerned with "diagnosing a problem in a specific context and attempting to solve it in that context". It is participatory - researchers and practitioners take part collaboratively, directly or indirectly, in implementing the research. Furthermore, it is self-evaluative - "modifications are continuously evaluated within the ongoing situation, the ultimate objective being to improve practice in some way or other".

This research makes use of small-scale action research case studies, the researcher functioning as a participant observer. In these case studies a form of triangulation is used in that questionnaire responses are elicited from both pupils and a non-participant observer - the class's teacher. Another source of data is a series of action research software evaluation workshops run by the author and in which the participants are practising teachers and teachers in training.

There were altogether nineteen participants in the software evaluation workshops. Nine of these were experienced, practising teachers and the other ten were student teachers in their diploma year. The software items evaluated by these groups are listed in section 2.3 of Chapter 2. Each of these programs was evaluated by a sampling (usually five or six people) of the nineteen participants. Evaluation findings are reported in Chapters 3 and 6.

The software evaluation reports - brief in that each attempts to capture merely the essence of a program - are set in the illuminative paradigm as discussed by Parlett and Hamilton (ed. Tawney, 1976):

> **"Illuminative evaluation, ... seeks ... to describe and interpret, and takes account of the contexts in which educational innovations**

must function. ... Within a three-stage framework of observation, further inquiry and explanation, the investigation's focus is progressively reduced and concentrated on the issues that emerge."

The primary concern of illuminative evaluation is with description and interpretation rather than measurement and prediction. Through evaluation of this form the study seeks to promote educational connoisseurship as advocated by Eisner (1985) with regard to computer applications in mathematics education.

In brief, the research approach is largely descriptive and analytical in nature, the emphasis being on identifying, describing, evaluating and classifying possible modes of computer application in the processes of mathematics education, and suggesting resultant curriculum innovations.

1.7 A Synopsis of Contents of this Report

The present chapter consists largely of a scene-setting overview, referring to possible applications of the computer in mathematics education, presenting an initial classification scheme and detailing the rationale for the study.

Chapter 2 presents a discussion on the evaluation of computer software, particularly with reference to mathematics education. It is a logical precursor to subsequent work on software classification.

Chapters 3 through 8 embody a primitive, linear separation of applications of the computer for the purposes of mathematics education into the following categories (each in a separate chapter in the order given):

/Chapter 3: ...

Chapter 3:	Drill-and-Practice;
Chapter 4:	Computer Tutoring, both Traditional Computer Assisted Instruction (CAI) and 'Intelligent' Tutoring;
Chapter 5:	Computer Guided Teaching (CGT);
Chapter 6:	Games and Simulations;
Chapter 7:	Problem Solving Environments;
Chapter 8:	Programming as a Mathematical Activity.

Chapter 9 entails a refined classification scheme, a multi-dimensional model, which takes account of overlaps between categories and the fact that programs can often be used in various ways.

Chapter 10 describes possible changes to the mathematics curriculum which could (or should) occur as a result of incorporating the computer into the processes of mathematics education.

Chapter 11 contains conclusions and recommendations arising from the research.

It should be mentioned that, rather than establishing a separate chapter reflecting the results of a literature review, the approach taken has been to integrate comments arising from the literature as and where appropriate throughout the text.

CHAPTER 2 THE EVALUATION OF COMPUTER SOFTWARE FOR USE IN MATHEMATICS EDUCATION

CHAPTER 2

THE EVALUATION OF COMPUTER SOFTWARE FOR USE IN MATHEMATICS EDUCATION

It is essential that the evaluation of computer software for use in mathematics education be explored in some detail, before any specific examples of software and the educational applications which they support are considered. It is necessary to establish standards by which the efficacy of programs can be judged, and this involves identifying criteria concerning technical design, content and educational viability which are, in some useful sense, measurable.

2.1 The Evaluation of Educational Computer Software

In the January 1984 issue of the magazine 'Primary Teaching and Micros' there is a suggested list of software evaluation criteria presented as 'Eight Steps to Evaluation':

1. Package title, source, machine, memory size, special equipment required.

2. Subject area, topic, target age, target ability, class, group or individual use.

3. Brief description.

4. Definition of educational aims and objectives.

5. Appropriate use of sound, colour, graphics.

/6. Documentation ...

6. Documentation, screen instructions, ease of use, user adaptability?

7. Achievement of educational objectives, robustness, educational value.

8. Technique used: what sort of program is it?

This list, although it has limitations, as indeed would any list of evaluation criteria, is intended for use by individual teachers to help them look critically at programs in a systematic way. It provides a framework within which the software reviewer can work.

Brown (1984) suggests a more detailed list:

Educational

- Is the program the best way of teaching the topic?
- Are there more efficient, cheaper methods?
- Does it lead on to other programs in a series?
- Is it consistent with the approach of other programs being used, or will its terminology and notation cause confusion?
- Is it compatible with teaching methods in general use in the school?
- Does the program relate positively to other classroom activities, or is the use of the microcomputer a completely unrelated activity?
- Does the program keep a record of the pupil's results?
- Is there any evidence that the program has been well tried and tested in schools?
- Were class teachers involved in a feed-back process at the design stage?

Practical

- Is the package easy to use?

- How much time must the teacher spend selecting skill-level, number of questions, etc., before the pupil can begin?
- Is the supporting documentation clear and adequate?
- Is the program 'child-proof'?
- Does the program progress at the right speed for the child?

The Open University in-service training course P541, 'Micros in Schools: Educational Software' (Open University Press, 1984), aims "to provide teachers with enough information and experience to be able to confidently assess educational software using a purpose-built check-list" (Blease, 1986). This list is as follows:

1. Educational Documentation
   - 1a. Statement of aims and objectives.
   - 1b. Information about the content and background.
   - 1c. Statement of intended type of use and audience.
   - 1d. Suggestion of ways to use the program.
   - 1e. Pupils' activities or worksheets.
   - 1f. Instructions for running the program.
   - 1g. Presentation of a typical run.
   - 1h. General impressions.

2. Achievement of Stated Aims
   - 2a. Aims/objectives.
   - 2b. General impressions.

3. Appropriateness of the Micro and Program
   - 3a. For teaching this topic.

3b. For the suggested audience and type of use.

3c. General impressions.

4. Screen Presentation

   4a. Use of graphics.

   4b. Use of colour and animation.

   4c. General impressions.

5. Friendliness and Flexibility of the Program

   5a. Helpful messages to correct user errors.

   5b. Help to pupils in understanding the program.

   5c. Versatility so that the user can control what the program does.

   5d. Feedback to pupils.

   5e. Adaptation of program to pupils' performance.

   5f. Record of pupils' performance kept by the program.

   5g. Accessibility of model to pupils.

   5h. Suggestions or help for teacher to modify the program.

   5i. General impressions.

6. Technical Documentation

   6a. Information about machine requirement.

   6b. Information about the model used.

   6c. Information about the program structure.

   6d. Listing and readability of the program code.

   6e. Portability, that is, ability to transfer the program to a different computer.

   6f. General impressions.

/Many other ...

Many other such lists have been proposed, for example, those by Coburn et al. (1982), Croft and Evans (1985), Mullan (1982), and the magazine 'Personal Computing Today' (May, 1984). Each proposed set of criteria has its own strengths and weaknesses but, in general, according to Straker (1982), the reviewer might realistically expect a check-list to provide answers to the following three major questions about any given educational computer program:

1. What kind of program is it?

2. Is it in principle suitable for the general teaching scheme of which it would be part?

3. What are its merits and in what ways could it be improved?

The check-lists presented above and, in particular, Straker's questions, have all had a bearing on the ideas concerning the evaluation of software for use in mathematics education which are described in sections 2.2 and 2.3 below.

2.2 Criteria for the Evaluation of Mathematical Software

There are numerous attributes which characterize educational computer programs - specifically those for use in mathematics education. Some of these are listed, together with criteria by which they can be judged, under various headings below.

2.2.1 General Description of the Program

It is obviously necessary for a potential user of a program to know the type of computer required: make; capacity; whether colour or monochrome; whether disk or tape driven; whether other peripherals are needed or possible. Other requirements at this descriptive level are:

program file name; full title (and series title if applicable); Dewey Number if there is one; syllabus area; author; publisher; cost; source; supporting materials; recommended user levels (age groups); and any other appropriate characteristics.

With the ever growing volume of educational computer software it is becoming more and more necessary to catalogue and store this software methodically. The characteristics listed above are all useful descriptors which could be built into a cataloguing system. In fact, Slaughter (1989), under the supervision of the author and J. Cornwell, the Rhodes University Education Department librarian, has designed such a system. A brief report, including examples of its use in cataloguing programs referred to in this thesis, is given in Appendix A9.

2.2.2 Purpose of the Program

For what purpose has the program been produced? Is it, broadly, a learning program, a teaching program, a control program, or does it have some other purpose?

2.2.2.1 Learning Programs

Learning programs as described by Ellingham and Haydon (MEP document, undated) are programs which enrich work initiated by the teacher. They are intended for use by individual pupils or groups of pupils, with minimum interference by, or interaction with, the teacher. They can involve reinforcement of skills, development of processes (as in generating alternative courses of action to solve a problem, forming and testing hypotheses, etc.), information retrieval (in the case where the program includes access to information stored within itself or in a data file), or teacher simulation (where the program attempts to simulate a teacher's responses to a pupil's errors).

/Following ...

Following the classification scheme outlined in section 1.3.1, some divisions within this category would be drill-and-practice, games, simulations, and computational environments.

2.2.2.2 Teaching Programs

Teaching programs can be taken to mean programs which have been designed to be used either by a teacher with a group of pupils, that is, as an aid to the teacher's presentation, or as teacher simulations in the sense of replacing the teacher as the initiator and facilitator of learning.

Included in this category would be computer guided teaching (see Chapter 5), computer assisted instruction and 'intelligent' tutoring (see Chapter 4).

2.2.2.3 Control Programs

Control programs are those which enable the control or driving of any piece of equipment outside the normal computer configuration, for example, a robot such as a Logo turtle or the HERO I, or a LEGO Technics buggy. These, however, do not fall within the ambit of this particular study - they are mentioned merely for the sake of completion. All the programs to be dealt with here are either learning or teaching programs or amalgams of both types.

2.2.3 Users of the Program

Who might use the program: an individual pupil, a large group of pupils, a small group of pupils, the teacher, the teacher and a group of pupils, ... ? In other words, what should be the underlying lesson management strategy?

2.2.4 Objectives of the Program

Where possible the objectives of the program designer(s) should be identified and the extent to which these are met, measured in some suitable way. In addition, the meeting of other objectives generally regarded as desirable, whether specified or not, should be investigated. Examples of software objectives for mathematics education are:

- development of problem-solving skills;
- development of investigatory ability;
- encouragement of the use of appropriate computational skills;
- encouragement of the making of predictions;
- enabling of the testing of predictions;
- promotion of awareness of inter-relationships of mathematical concepts;
- effective communication in words, symbols and diagrams;
- development of a positive attitude to mathematics as an interesting and attractive subject.

2.2.5 Technical Design of the Program

Concerning the technical design of the program, there are certain criteria which are obviously important:

- instructions to the user should be clear;
- the user should have good control over the execution of the program;
- progress through the program should be logical and smooth;
- graphics, if they are used, should be effective;
- the user should have freedom to explore mathematical concepts;

- any 'help' information provided by the program should be relevant and useful.

2.2.6 Subject Content

A most important attribute of any mathematics education program is its subject content, and there are various criteria by which this can (or must) be judged:

- is it accurate?
- what are the underlying mathematical concepts?
- does it motivate the pupil?
- is it appropriate in terms of the prescribed syllabus?
- is it applicable to other subject areas?

2.2.7 Documentation for Teachers

It is reasonable to expect a program designed for use as a mathematical education resource (or for any other purpose for that matter) to be accompanied by an instruction manual which is clear, easy to follow, useful and accurate.

2.2.8 Viability of the Program as an Educational Tool

Any program designed for use as an educational tool must be viable - it must work well and it must serve to enhance teaching and or learning. Viability is essentially a summative attribute: if all the applicable criteria listed above are met with a high degree of success, then the program can be judged a viable educational tool.

/2.3 Questionnaires ...

2.3 Questionnaires for the Evaluation of Mathematical Software

A careful consideration of the ideas expressed in sections 2.1 and 2.2 above led to the design of three questionnaires for use as data collection instruments. These are headed 'Computer Software for Mathematics Education: Evaluation Form for Teachers', 'Computer Software for Mathematics Education: Evaluation Form for Pupils', and 'Teacher Evaluation of Software for Use in Mathematics Education'. Copies of these instruments can be found in Appendices A1, A2, and A3 respectively. They are orientated towards the assessment of both the technical quality of a program and its educational viability. It should be noted that the design of the first two questionnaires was heavily influenced, as was the discussion in section 2.2 above, by the design of a questionnaire previously used as part of this study for evaluating the *SERGO* system (Marsh, T.A. and Marsh, C.J.A., 1989). A questionnaire designed, as part of the MEP project in the United Kingdom, for the "Evaluation of Microcomputer Programs for Primary Schools" (Ellingham and Haydon, undated) was also found to be useful.

The evaluation forms which resulted from this part of the study were used to gather data in a series of small-scale action research case studies involving the use of selected items of software in the classroom and laboratory. This research is reported in subsequent chapters of this write-up. The abovementioned *SERGO* system evaluation is reported in the next chapter.

The questionnaire called 'Teacher Evaluation of Software for Use in Mathematics Education' was used specifically for gathering the opinions of groups of practising teachers and trainee teachers in a series of BBC-software evaluation workshops run by the author. The programs considered in these workshops were:

/*ANGLE360*...

ANGLE360, *BOXED*, *DARTS*, *DEFINE*, *IDENTIFY*, *LINEOVER*, *MINIMAX* and *TAXI* (all by SMILE);

FRAC, *SUBGAME*, *EUREKA* and *ERGO* (all by ITMA);

VECTORS: *NAVIGATION*; *TREASURE HUNT* and *RACING DRIVER* (all by CAMBRIDGE MICROSOFT);

MONTY (by SLIMWAM).

The evaluations are reported in Chapters 3 and 6.

Details concerning the SMILE, ITMA, CAMBRIDGE MICROSOFT and SLIMWAM organizations are given in the reference section of the thesis.

CHAPTER 3 DRILL-AND-PRACTICE COMPUTER PROGRAMS IN MATHEMATICS

CHAPTER 3

DRILL-AND-PRACTICE COMPUTER PROGRAMS IN MATHEMATICS

3.1 The Nature of Computer Based Drill-and-Practice

In typical drill-and-practice computer programs, there is no attempt to introduce new knowledge or new concepts. Instead, students are asked to answer questions and perform tasks which develop or sharpen the skills associated with material to which they have already been exposed. They learn by repeatedly solving, or attempting to solve, problems that are grouped in a developmental sequence, correct responses being reinforced and corrective feedback being supplied where necessary.

According to Kelman et al. (1983):

> **"Three essential assumptions are at the heart of all drill-and-practice ... programs: first, that basic skills are learned like physical skills through repeated practice; second, that more complex ideas and skills can be learned by being broken down for the student into appropriate sequences of sub-ideas and sub-skills; and third, that students will replicate behaviours that are reinforced with a pleasant experience."**

It should not be forgotten, however, that drill-and-practice is often seen as purely augmenting and consolidating conceptual teaching of basic skills. In addition, it is often used in full cognizance of the fact that it is limited to the learning of simple ideas and skills, but conceived without any Pavlov-Skinner expectations.

/It should ...

It should be noted also that there is a related teaching strategy which we might call 'drill-in-context' in which concept development and skill exercise are integrated. Several of the programs to be considered in later chapters fall into this category.

The computer-based drill-and-practice medium has a valuable educational role to play. Kelman et al. (1983) report that:

> **"Some research indicates that significant gains have been realized, above those achieved using traditional techniques, when students in basic skills improvement courses have had regular and frequent sessions with drill-and-practice programs."**

There is a large volume of mathematics education programs in the drill-and-practice category. The best of these often provide self-paced problem sets, employ creative graphics, animation and sound effectively, and are highly motivating.

3.2 Large-Scale Computer Systems for Mathematics Drill-and-Practice Used in South Africa

One of the best known large-scale computer systems using the drill-and-practice mode is *TOAM*, developed in Israel. This is a system designed mainly for the purpose of presenting and administering arithmetic drill-and-practice programs. Other well known systems which employ this mode are *PLATO* (for example, the so-called *SASEC* programs) and *SERGO*. All three of these systems are discussed below.

/3.2.1 The *microPLATO* ...

3.2.1 The *microPLATO SASEC* Programs

The *microPLATO SASEC* (South African Schools Educational Courseware) collection, running on IBM-compatible microcomputers, was developed by Control Data South Africa. The programs were designed by practising teachers to help pupils prepare for their matriculation examinations. As explained in the documentation, "These lessons will not replace good tuition. They are a supplement to aid teachers, giving students opportunities to practise topics that have already been taught." (Control Data *SASEC*, undated).

The suite of microcomputer *SASEC* programs covering the algebra syllabuses (Higher and Standard Grades) for standards 9 and 10 were evaluated for the author as part of the project reported in this thesis by a discerning colleague, J. M. Alummoottil of the Dr W. B. Rubusana College of Education, Mdantsanse, Ciskei. Some of his findings, based on the questionnaire 'Computer Software for Mathematics Education: Evaluation Form for Teachers' (see Appendix A1), are recorded immediately below.

> The programs help moderately in the development of investigatory ability and problem-solving skills, as well as in encouraging the use of appropriate computational skills and the making and testing of predictions. They are also of some help towards promoting awareness of inter-relationships of mathematical concepts.
>
> The programs communicate effectively in words, symbols and diagrams, but there is no evidence that they are effective in developing a positive attitude to mathematics as an interesting and attractive subject.
>
> Instructions to the user are clear. The user has good control over the execution of each program and progress through the program is logical and smooth.

Graphics are used with moderate effectiveness and any 'help' information provided by the programs is moderately relevant and useful.

The user does not have freedom to explore mathematical concepts at various stages. In fact, the programs offer only drill-and-practice.

Alummoottil's conclusion is that the *SASEC* software forms a viable reinforcement type learning aid, provided that suitable computers on which to run it are made available to students on a one-to-one basis, regularly for sufficiently long periods of time.

3.2.2 The *SERGO* System and an Evaluation Thereof

SERGO is an acronym for the Afrikaans name 'Sentrum vir Rekenaargesteunde Onderrig', translated as 'Centre for Computer Aided Instruction'. It designates an extensive drill-and-practice system covering the South African mathematics syllabus from the standard 1 level through to the end of the Junior Secondary phase. Work is currently in progress to extend this coverage through to the end of the Senior Secondary phase.

Under the author's supervision, two colleagues and a final-year B.Prim.Ed. student carried out an evaluation of the *SERGO* CAI system for mathematics during the four months from July to October 1988. Full details of the findings were reported by the author and one of his colleagues (Marsh, T.A. and Marsh, C.J.A., 1989). A brief synopsis of this report is given below. Before presenting this synopsis, it should be noted that an interesting critique of the *SERGO* system and CAI in general has been done by M. Appel and S. Appel (1987). Their main argument is that *SERGO* cannot educate, in the true sense of the word, because it is grounded in positivism and its psychological counterpart, behaviourism.

3.2.2.1 Phases of the Evaluation

The evaluation was done in two phases. First a small-scale pilot study investigated the use of the *SERGO* system by pupils and staff of the Victoria Girls' Primary School in Grahamstown. This experiment involved thirty-six pupils, three groups of four from each of standards 2, 3 and 4, and three teachers, each being a class teacher at one of these levels. The group size of four was determined by the number of computers available. Each group of pupils had just a single one hour session working on the *SERGO* system. The teachers attended these sessions, that is, three sessions each at the appropriate level. They also spent an afternoon examining the full set of *SERGO* lessons and the pupils' records. The opinions of pupils and teachers were collected by means of questionnaires.

As the teachers found some of the questions and the rating scale used difficult to interpret, their responses will not be recorded here. Instead we shall consider the responses given on a simplified questionnaire (see Appendix A4) by the teachers who participated in the second phase of the evaluation. A copy of the questionnaire used to gather data from the pupil participants may be found in Appendix A5.

The second phase involved a more extensive investigation carried out in three Grahamstown schools simultaneously over a period of eight weeks. The participating schools were The Diocesan School for Girls (DSG), Kingswood Junior School and St. Andrew's College. Following an explanatory talk and demonstration, a set of *SERGO* software, a key card and documentation were loaned to each of the schools and the use of the system was left up to the teachers entirely. DSG used lessons for standards 2 to 7, Kingswood lessons for standards 2 to 5 and St. Andrew's lessons for standards 6 and 7. It should be mentioned that at the time of this evaluation the software for the Senior Secondary phase was not yet available.

DSG left a computer in their library and allowed pupils to access the *SERGO* system on a voluntary basis. Kingswood set up a computer with the *SERGO* system in their computer laboratory and a few pupils used the system under the guidance of a teacher. A St. Andrew's teacher used the system in the school's computer laboratory with a few pupils on an ad hoc basis.

3.2.2.2 Response from the Pupils

Altogether thirty-three pupils spread over standards 2 to 7 participated in the second phase of the evaluation. All the pupils stated that they enjoyed the *SERGO* lessons, with many of them commenting that the lessons were fun and interesting. Twenty-seven pupils (that is, 82% of the sample) maintained that the lessons helped their understanding of mathematics. It should be noted, however, that eleven pupils (that is, 33%) did not fully understand the lessons' instructions and had to consult either the teacher or other pupils.

It was interesting to note that the majority of those who normally did well in mathematics liked both mathematics and working on computers, whereas those who did not do well in mathematics not only disliked the subject, but also were not as enthusiastic about working on computers. Overall though, twenty-six (79%) liked working on computers and twenty-seven (82%) wanted more computer based lessons.

/3.2.2.3 Response ...

3.2.2.3 Response from the Teachers

3.2.2.3.1 Attitude towards Lesson Content

Misgivings were voiced regarding the fact that all pupils regardless of ability were frequently able to score 100% and yet still did not understand fully the underlying mathematical concepts of certain exercises.

The methodology was found to be very out-dated and there were fears that pupils would come to rely on the drill-and-practice approach rather than to employ a more open-ended approach to the manipulation of numbers and symbols in problem-solving.

It was felt that the system had its merits for consolidating mathematical concepts for the average to below-average pupils, who enjoyed the system most and were thrilled with their uncustomary success. In one case, even a pupil with severe learning problems scored in the 80-90% range! However, the brighter pupils quickly tired of the system and found it to be unstimulating and lacking challenge. There was little room for creativity and exploration.

3.2.2.3.2 Attitude towards Technical Design

The teachers found the registering and moving of pupils in the system to be very time-consuming and tedious. They commented on the awkwardness of having to work with so many disks. They also found it difficult to preview the lessons.

There were also criticisms about the conflicting ways of ordering the digits of an answer and the lack of a means of typing the 'carry' number.

/3.2.2.3.3 Attitude towards ...

3.2.2.3.3 Attitude towards Record-Keeping and Documentation

The teachers found the record-keeping facility useful. Although the manual was adequate, several teachers would have liked the inclusion of a more detailed description of the contents of the various indexes.

3.2.2.3.4 Positive 'Spin-Offs' from the System

The teachers saw as beneficial that the pupils were given the opportunity to learn to follow instructions carefully and independently of a teacher, and that they were given the experience of setting goals and applying themselves to a task-orientated system.

3.2.2.4 Conclusions and Recommendations

As the system stands it is recommended for reinforcing mathematical concepts for the less able pupils; its lack of creative scope and challenge does not suit it to the more able pupils.

The system's teaching methodology appears to be out-dated and does not employ any of the currently advocated teaching approaches and theories related to learning mathematics. It does not: develop investigatory ability; encourage the making and testing of predictions; encourage awareness of inter-relationships of mathematical concepts; or apply mathematics to everyday situations.

The subject content was found to be accurate and highly applicable to the present mathematics syllabus. There were not many other subject areas identified, apart from general calculation, where the system could be used.

/There are ...

There are aspects of the technical design mentioned above and in the teachers' comments which need to be carefully examined and streamlined, and in particular a 'browsing mode' should be developed to allow teachers to preview the lesson material with the minimum of fuss.

The accompanying documentation is clear and easy to refer to, but there could be more detail included on the actual content of the indexes. The computer-based instructions to the teachers were clear, but those to the pupils were not readily understood by all.

3.2.3 The *TOAM* System

The Israeli designed *TOAM* System for computer assisted instruction (CAI) consists of a central computer and sixteen to thirty-two dedicated pupil terminals. The supervision of the system and administrative tasks such as registering pupils, printing out progress reports, etc., are carried out by a specially trained administrator. The system was developed specifically to upgrade the learning process and to fill the gap created by the paucity of trained teachers and educational facilities in Israel - a problem possibly even more acute in South Africa.

The system software presents fifteen arithmetic topics, each of which comprises exercises graded into levels of increasing difficulty. The range of levels covers all the aspects of a topic as it is studied in the different grades of the primary school. For example, in topic 1 (Number Systems), the exercises begin in the first grade and continue through to the sixth grade. The exercises are generated algorithmically by using random numbers.

The initial level at which a pupil begins to work in a topic is pre-determined by extensive testing by the system. When a problem is presented to a pupil, the computer leads him through the steps required to arrive at the solution, checking each character he types in. This process ensures that the pupil knows at exactly which step he has made his mistake. Each pupil receives a variety of exercises, according to his level, from several different topics during his work session, and his progress in each topic is independent of his progress in other topics. The system automatically reinforces learning in topics where a pupil is below average by presenting more exercises in those topics and fewer exercises in topics where his performance is satisfactory.

Detailed reports of the pupils' activities on the system are printed. These enable the teacher to handle the specific problems of each pupil accordingly.

The *TOAM* system was introduced in 1983 into a few schools in Soweto in the form of a mobile trailer classroom. Evaluation of the system was carried out by the Centre for Continuing Education of the University of the Witwatersrand. The evaluators, C. MacDonald, P. Smith and T. Metrowich (1986), found that the pupils agreed that computers are fun to work on and generally they (the pupils) felt more confident that they could do mathematics and solve the problems presented to them by the computer than by conventional means. C. MacDonald (1986) found that the Sowetan teachers who had pupils on the *TOAM* system had a generally positive attitude towards using it:

> **"Several teachers have asked for more systems, not only because they hope that their school will receive a mobile unit, but also because they want other children to receive the benefit of working on *TOAM*."**

It should be recorded that the Sowetan teachers and pupils involved in this project were experiencing the use of computers in education for the first time and their evaluation of the *TOAM* system was, in that sense, done in a vacuum. They had no standards against which to compare it.

3.3 Small-Scale Computer Programs for Mathematics Drill-and-Practice

3.3.1 Milliken Software

A sophisticated package of microcomputer arithmetic drill-and-practice programs is marketed by Milliken Computer Courseware ("Math Sequences 1980"). These programs cover addition, subtraction, multiplication, division, laws of arithmetic, integers, fractions, decimals, percents, equations, and measurement formulas. The package keeps track of each pupil's performance and provides the teacher with progress reports.

3.3.2 ITMA Software

There are two interesting small-scale drill-and-practice applications developed as part of the ITMA (Investigations on Teaching with Microcomputers as an Aid) project in England. These are programs called *DIRECTED* and *AUTOFRAC* which provide drill-and-practice in working with directed numbers and equivalent fractions respectively. Another drill-and-practice program in the same series, but apparently not of the same quality as these two, is the program *FRAC*.

Three further small-scale drill-and-practice programs comprising part of the British SMILE Next 17 series are those called *ANGLE360*, *DEFINE* and *IDENTIFY*.

All the abovementioned programs are discussed in some detail below, results of evaluations, carried out as described in section 1.6 of Chapter 1, for *FRAC* and the SMILE programs being included. It so happened that the SMILE programs were only evaluated by student teachers and not by any of the practising teachers who participated in the evaluation workshops.

3.3.2.1 The ITMA Program *DIRECTED*

DIRECTED, designed and programmed by T. Greenslade, uses the movement of a graphically represented robot up and down a number line to instigate drill-and-practice in the addition and subtraction of directed numbers. The robot follows simple rules to provide the answer to a sum:

1. addition - face to the right;
   subtraction - face to the left;
2. positive number - walk forwards;
   negative number - walk backwards.

The magnitude of the numbers in the sum is always less than 10 and the following types of sum are catered for:

- sum with unsigned numbers, such as 3 + 2 or 4 - 3;
- sum with positively signed numbers, such as ${}^{+}3 + {}^{+}2$ or ${}^{+}3 - {}^{+}2$;
- sum with negatively signed numbers, such as ${}^{-}3 + {}^{-}2$ or ${}^{-}3 - {}^{-}2$;
- sum with a mixture of positively and negatively signed numbers.

There are three modes of program operation:

- Demonstration mode - the computer asks for sums of the above types to be input and the robot 'walks' along the number line from first number through second number to answer.

/- Practice ...

- Practice - when the first number has been input, the robot moves automatically to the correct place on the line; when the second number has been input, the computer waits for an answer to the sum; if an incorrect answer is entered, the robot stays where it is and 'comments' on the error; if the correct answer is entered the robot moves to the position indicated and gives a congratulatory message.

- Test mode - a number of randomly selected sums are produced on the screen in so-called 'film' mode; the time delay may be altered and answers can be given immediately or at the end.

The program provides a simple but motivating way of practising single digit addition and subtraction sums with signed and unsigned numbers.

3.3.2.2 The ITMA program *AUTOFRAC*

AUTOFRAC, designed and programmed by J. Coupland (ed. R. Phillips, 1984), provides a continuous 'film' mode display of equivalent fractions, in the form, for example,

$$8/6 \;=\; 4/?$$

Once started, the program runs indefinitely without intervention from the user. Each new pair of equivalent fractions, chosen at random, is displayed in its incomplete form for a specified time, after which the missing number is shown. The program then moves on to the next pair. The user may specify the level of difficulty required, and the delay time for which the unsolved problem is displayed (wherein, together with the ability to generate any number of problems randomly, lies the essential difference for this purpose between the capabilities of a computer and that of a video playing machine).

The level of success achieved by the program is indicated by this comment from its author (J. Coupland, ed. R. Phillips, 1984):

> **"... it has been amazing to leave the program running in the lunch-hour and to watch children glued to the windows of the classroom, desperately trying to find the correct solutions!"**

3.3.2.3 The ITMA Program *FRAC*

FRAC, produced by the Liverpool Primary Computer Group as part of the ITMA project, provides drill-and-practice in working with fractions in different forms. It involves addition of fractions and writing mixed numbers as fractions, appropriate at the standard 4-5 level. Execution of the program is rather slow and the exercises are very routine.

3.3.2.3.1 An Evaluation of *FRAC*

This program was classified by all the teachers who evaluated it as a teaching/drill-and-practice program for which they did not see much potential use except as a corrective exercise for weak pupils.

The student teachers who evaluated it came up with a similar classification for the program, but they were more positive in finding a use for it in teaching pupils to distinguish between different types of fractions.

Both groups felt that the initial teaching of the concepts of fractions could be better accomplished by a teacher.

/The general ...

The general consensus was that it was a worthwhile tool for use by weaker pupils who needed extra practice at working with fractions. The teacher could achieve the same purpose without the computer but only at the expense of a great deal of precious time.

A drawback identified by one of the respondents was that the program goes through the complete section again if the user gives just one wrong answer to a question within that section.

3.3.3 SMILE Software

3.3.3.1 The SMILE Program *ANGLE360*

ANGLE360 gives the user drill-and-practice at estimating sizes of angles between 0° and 360°. A rotating arm, shown graphically on the screen, sweeps out angles in the plane. The user stops this arm with a single key press in order to indicate his estimate of an angle of a particular required size. The program judges the accuracy of the estimate and responds accordingly.

It is worth noting that the proposed new syllabus (DEC Draft Document, 1989) suggests the use of suitable computer games as a means of providing practice with the concept of angular measure. *ANGLE360* might prove to be a useful resource for this purpose.

3.3.3.1.1 An Evaluation of *ANGLE360*

This drill-and-practice game, the student teachers agreed, had use in familiarizing pupils with the sizes of angles and they all maintained that, if available, they would use it for that

/purpose ...

purpose. However, they thought that the computer program was not essential in this case and that a dial clock or a protractor could be used instead.

3.3.3.2 The SMILE Program *DEFINE*

DEFINE enables drill-and-practice at distinguishing the properties of a given number: whether it is a multiple of some number, a factor of another given number, a square number or a prime number. Given a list of numbers of length chosen by the user, his task is to define one of these numbers relative to the others using as few statements as possible. The numbers all lie between 0 and 100.

3.3.3.2.1 An Evaluation of *DEFINE*

This drill-and-practice program, the student teachers agreed, could be used by pupils to identify numbers within the number system particularly at the standard 6 level. All but one student said that they would use this program in that way as more traditional ways would be slow and laborious by comparison.

3.3.3.3 The SMILE Program *IDENTIFY*

IDENTIFY is similar to *DEFINE*. It requires the user to single out a number selected by the program from a list of fourteen numbers generated by the program - the numbers all lie between 0 and 50. The object is to identify this number in as few steps as possible by asking questions from the following list: is it a factor of ...? is it a multiple of ...? is it a square number? is it a prime number?

/3.3.3.3.1 An Evaluation ...

3.3.3.3.1 An Evaluation of *IDENTIFY*

This program was also described by the student teachers as being essentially a drill-and-practice program which could be used by pupils to identify and describe specific numbers in the standard 6 number system. Most of them said they would use the program since pen and paper would not be as effective and challenging for the pupils.

3.4 Concluding Remarks

The large-scale systems discussed in this chapter, that is, *microPLATO SASEC*, *SERGO* and *TOAM*, all provide reinforcement learning facilities and are useful for remediation purposes. They are mainly suited to the needs of the weaker pupil. They do not offer stimulation for the more able pupil.

Some of the small-scale, single-purpose, programs dealt with are potentially useful learning aids for pupils of varying ability.

Several of the software items considered here will be reviewed and reclassified in Chapter 9.

CHAPTER 4 COMPUTER TUTORING SYSTEMS IN MATHEMATICS

CHAPTER 4

COMPUTER TUTORING SYSTEMS IN MATHEMATICS

Tutorial teaching often follows the pattern of question and answer dialogue. The tutor guides the student's learning, drawing responses from him through a process of judicious, goal-directed questioning. Correct answers elicit reinforcement, explanations are provided where necessary, and the cycle continues until the student has mastered the material under discussion. Computer tutoring systems are often based on this model.

Tutorial programs are normally more sophisticated than drill-and-practice programs. Whereas drill-and-practice programs usually provide self-correcting drill on skills the student has already been taught, tutorial programs attempt to teach new information and concepts.

In this chapter we shall consider two types of computer tutoring system: the traditional computer assisted instruction (CAI) system and the so-called 'intelligent' tutoring system (ITS).

4.1 Traditional Computer Assisted Instruction (CAI) Systems

Many traditional computer assisted instruction (CAI) programs may be represented by the flowchart, adapted from Oberem (1983), shown below:

/Figure 4-1: ...

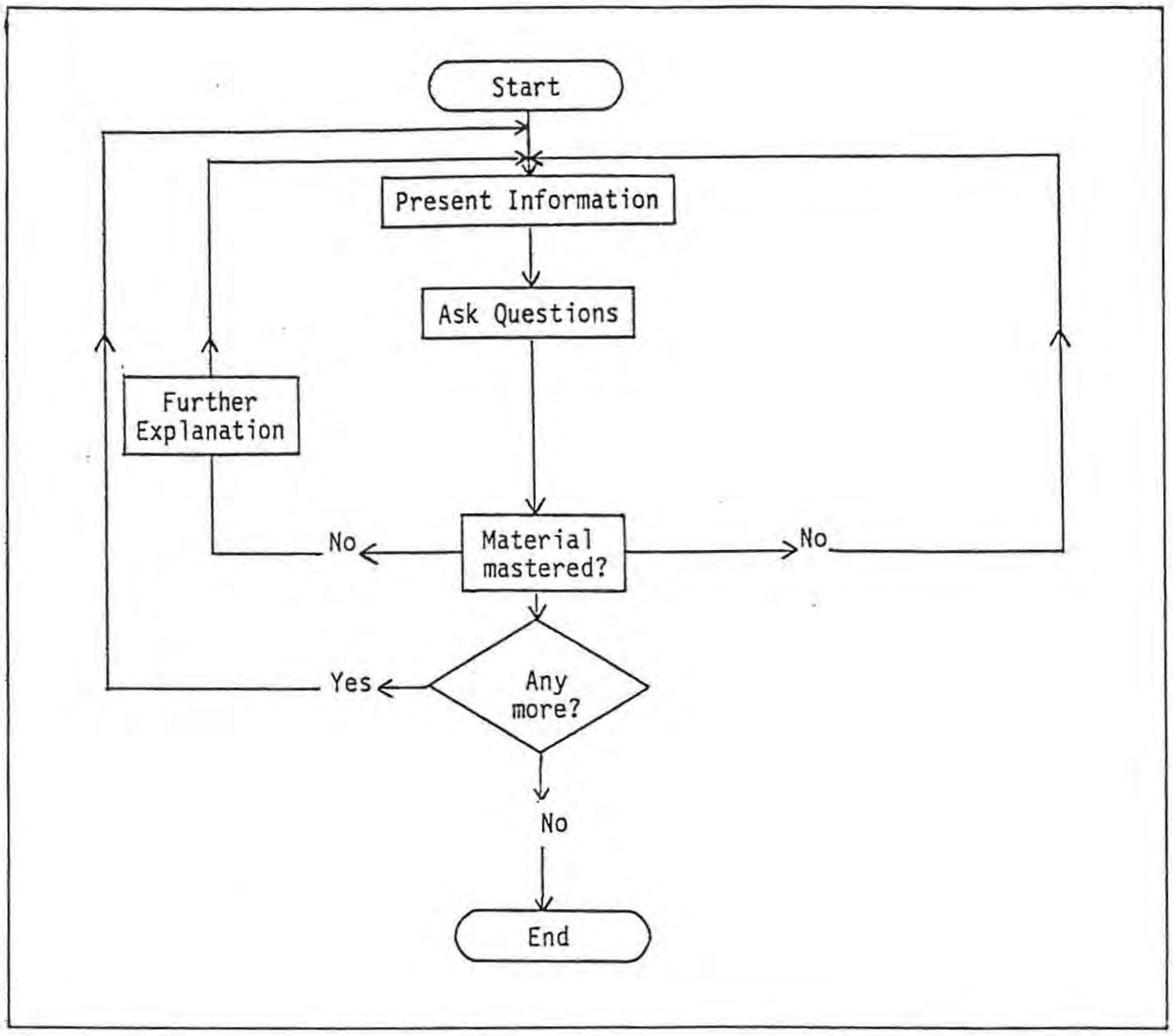


Figure 4-1: A Flowchart Representing Traditional CAI

Apart from a computer, the systems which run these programs have two major components, subject matter and tutorial driver, as shown below:

/Figure 4-2: ...

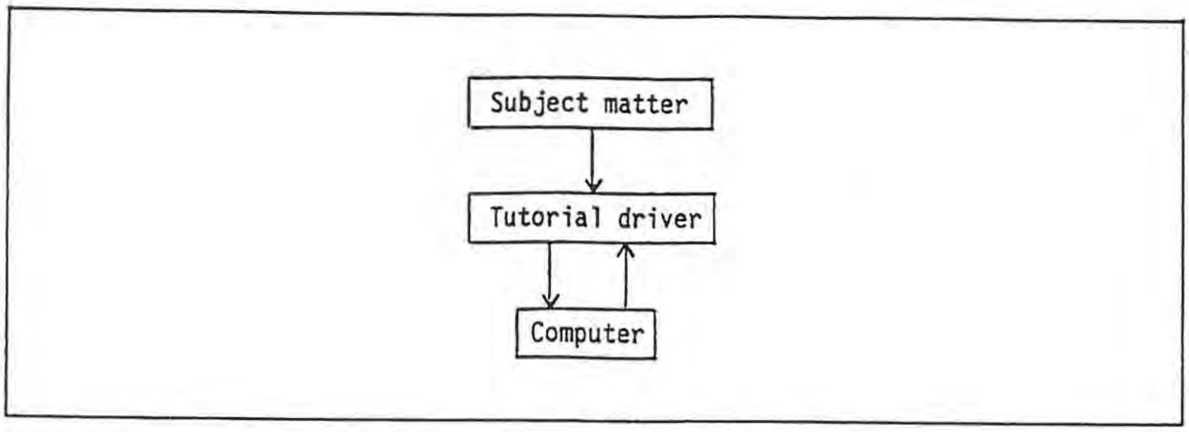


Figure 4-2: Major Components of Traditional CAI Systems

The subject matter, in the form of information, explanations and questions, is delivered by the tutorial driver. The tutorial driver also performs answer judging, branching and item selection.

The student is presented with a unit, or frame, of information, sometimes several screens full. His understanding of it is then tested by means of appropriate test questions, often of the multiple-choice type. If he has mastered the material in a given unit, he either carries on with the next unit, or 'quits' if he has completed the lesson. If he has not mastered the unit, he may review it, with the help of further explanation (if required), until mastery is achieved.

A CAI system of the above kind offers the advantage that it provides the student with a self-paced, interactive learning environment. A rather severe limitation, however, is that there is no individualization in the presentation. Each unit of information is presented in an identical way to every student, with the same assumptions being built in about the degree of mastery of the previous unit, and with no account being taken of the ability of the particular student. Other limitations are that the criteria for branching must be explicitly stated in advance and all questions and corresponding answers must be constructed and stored in advance, making such systems somewhat inflexible.

/Traditional ...

Traditional CAI systems, as described above, function according to pre-stored question-and-answer sequences. They cannot actually solve the problems they pose. This is the point of departure for so-called 'intelligent' tutoring systems with their built-in expert knowledge. Intelligent tutoring systems will be discussed in section 4.2 below.

4.1.1 Some Examples of Traditional CAI Construction and Delivery Systems

There is a wealth of tried and tested software in the form of mathematics tutorial programs, much of it originally developed on mainframe computers but now converted, or being converted, to run on microcomputers.

4.1.1.1 The *PLATO* System

The *PLATO* system with its Tutor authoring language is perhaps the most well known mainframe computer tutoring system of the traditional CAI type. CAI tutorial lessons on a wide variety of topics in mathematics, at a number of different levels, have been developed using this software. To name a few, there are CAI tutorials on sets and numbers, polynomials and factoring, equations and inequalities, rational expressions, graphs and relations, systems of equations, probability, and calculus.

4.1.1.2 Microcomputer Lesson Authoring Systems

Of course, *PLATO* and other similar projects are large-scale, highly expensive operations - not in the ambit of ordinary school education, except perhaps at the microcomputer level. There are, however, several fairly powerful and more readily accessible microcomputer tutorial construction and delivery systems, such as *TenCORE*, *SuperPILOT* and *QUEST*. The volume of mathematics tutorial material resulting from these systems is also considerable.

4.2 Intelligent Tutoring Systems (ITSs)

In section 4.1 we considered ordinary tutoring systems which cannot solve the problems they pose, but rather contain pre-stored solutions to these problems. Another limitation of such systems is that they cannot solve problems posed by the user. They are merely (sophisticated) information presenting question-and-answer administrators.

We turn our attention now to a consideration of so-called 'intelligent' tutoring systems (ITSs). These are systems endowed with all the necessary expertise required for the solving of the problems which they pose, or indeed, those posed by the user. Some of these systems can also explain their reasoning. Examples of such systems, all described by Sleeman and Brown (1982), are: Brown and Burton's *BUGGY*, a system which detects and accounts for systematic errors in pupils' arithmetic computation; Stanfield, Carr and Goldstein's *WUSOR*, a mathematical game playing coach; Burton and Brown's *WEST*, also a mathematical game playing coach; Genesereth's *ADVISOR*, an automated consultant for users of the *MACSYMA* algebraic manipulation system; Sleeman and Smith's *LMS* (Leeds Modelling System), which models a student's algebraic performance and diagnoses his errors; Kimball's *INTEGRATE*, a self-improving symbolic integration tutor; and O'Shea's *QUADRATIC*, a self-improving quadratic function tutor. Another such system, described in some detail below, is Marsh and Halpin's (1986b) *SKETCHER*, a rational function curve sketcher.

ITSs, sometimes referred to as ICAI systems, result from the marriage between so-called AI ('artificial intelligence') and CAI (computer assisted instruction). They comprise two major components: an expert system capable of solving any problem in a given limited domain; and a tutoring system capable of interacting with the user and explaining step-by-step the solution to any problem solved by the expert.

The components of an ITS are shown in the figure below (taken from Oberem (1983)):

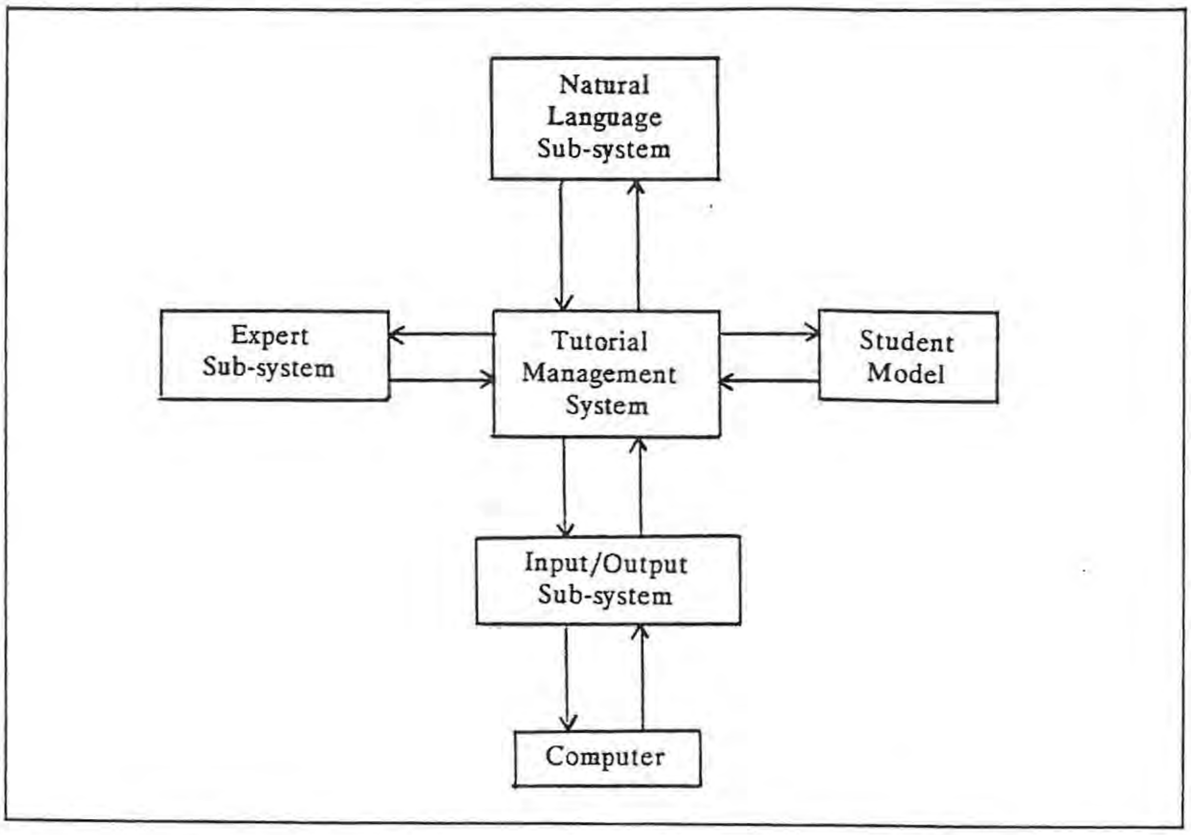


Figure 4-3: The Components of an Intelligent Tutoring System

In broad outline, Oberem (1983) describes these components as follows:

The Input/Output (I/O) Sub-system interfaces the user to the Tutorial Management System and is concerned mostly with screen format and response timing.

The Tutorial Management System applies an instructional strategy and tailors this to the needs of each individual student.

/The Expert ...

The Expert Sub-system holds domain specific knowledge as well as a problem solving strategy. It simulates the working of a human expert.

The Student Model component models the student's thought processes, assesses his responses and keeps a dynamic profile of his ability and level of proficiency.

The Natural Language Sub-system parses input from the student and converts it to a suitable internal representation. It must also be able to compose sentences as output.

In an ITS both the student and the tutor are modelled as realistically as possible.

In discussing ITSs, we shall confine our attention to *SKETCHER*, it being the system with which the author is most familiar, having been instrumental in its development.

4.2.1 An Intelligent Tutoring System for Rational Function Curve Sketching

SKETCHER, written in the LISP derivative muSIMP and running on the IBM PC (with 512K) is able to determine all the essential features necessary for sketching the curve of any rational function in a particular domain. It also produces a sketch of the curve in question. It makes use of some of the producer-supplied *muMATH* routines to compute such features as zeros, asymptotes, maxima and minima. These capabilities are embodied in the expert component of the system.

The approach used in developing *SKETCHER* was that of providing a supportive learning environment intended to facilitate learning-by-doing. The aim was to combine the problem solving experience and motivation of discovery learning with the effective guidance of tutorial interactions.

The first phase of the project was to construct a program with built-in expertise pertaining to the basic decision-making aspects of rational function curve sketching. This phase was successfully completed as illustrated below. The second phase, upon which work is still in progress, is the development and linking in of a tutoring program capable of providing step-by-step explanations to the problem solving processes executed by the expert program. The present state of development of the tutoring program is such that it can give step-by-step explanations of the way that equations in a particular class are solved. This is reported in Halpin and Marsh (1988) and discussed in section 4.2.1.2 below.

The efficacy of the expert component of *SKETCHER* is borne out by the fact that it is able to emulate a student in arriving at all the essential features required for the sketching of a rational function. It computes intercepts, maxima and minima, vertical and oblique asymptotes as well as judging behaviour with respect to these asymptotes, and determines the function's sign between x-intercepts and vertical asymptotes. Given a particular function, it goes through the following process: sets the numerator equal to zero and solves for the x-intercepts; sets x to zero and calculates the y-intercept; sets the denominator to zero and solves to find vertical asymptotes; divides the denominator into the numerator to find an oblique asymptote; differentiates the function and sets the derivative to zero in order to solve for stationary points; finds the second derivative in order to establish maxima, minima and points of inflection; and draws up a table of signs for the determination of regions of positivity and negativity.

This is all illustrated by the way in which the program handles, for example, the function defined by

/$2x^2$ ...

$$\frac{2x^2-9x-18}{x^2-x-2}$$

The input has to be given in the form indicated below:

```
Numerator:    2X^2-9X-18

Denominator:    X^2-X-2
```

The output produced is as follows:

```
X-intercepts:   {6;-3/2}

Y-intercept:    9

Vertical asymptotes:    {2;-1}

Oblique asymptote:    Y = 2

Graph cuts oblique asymptote:      at X = -2

Local maximum:    (-4;25/9)

Local minimum:    (0;9)

Sign table:     +++(-3/2)---(-1)+++(2)---(6)+++
```

Having displayed this information the program then produces a sketch of the function showing all the critical features.

/Figure 4-4: ...

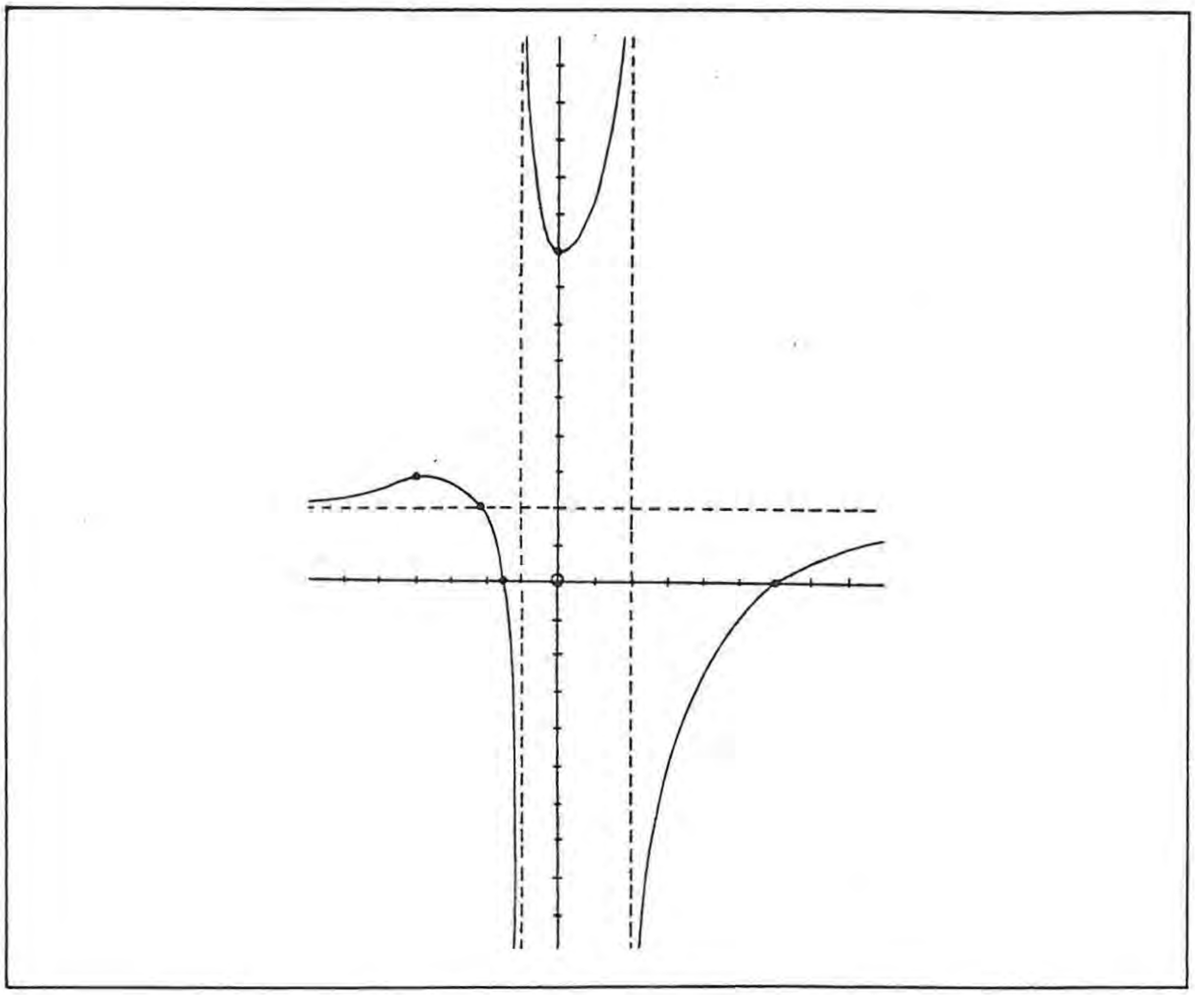


Figure 4-4: **Type of Graph Produced by *SKETCHER***

It should be stressed that this program differs markedly from conventional curve sketching programs in that it performs algebraic manipulations as opposed to numeric computations. It differentiates, for example, (2X^2-9X-18)/(X^2-X-2), using the quotient rule, to obtain (-7X^2-28X)/(X^2-X-2)^2, then sets this derivative to zero and solves to find X = 0 or X = -4. After this it evaluates the second derivative to determine whether the stationary points at X = 0 and X = -4 are maxima, minima, or points of inflection. In this way it pieces together, as a person would, the information necessary for the drawing of a given function's curve. Conventional programs simply compute numerous pairs of co-ordinates and plot the resulting points.

Thorough tests have shown that the expert program deals successfully with functions of the following forms:

'linear', 1 / 'linear', 'linear'/ 'linear';

'quadratic', 1 / 'quadratic', 'linear'/ 'quadratic',
'quadratic'/ 'linear', 'quadratic'/ 'quadratic';

'cubic', 1 / 'cubic', 'linear'/ 'cubic',
'cubic'/ 'linear', 'quadratic'/ 'cubic',
'cubic'/ 'quadratic', 'cubic'/ 'cubic'.

4.2.1.1 A Curve Sketching Workshop run for Standard 9 Pupils

4.2.1.1.1 Description of Workshop

In October of 1988 the author ran a brief workshop for standard 9 pupils, Higher and Standard Grade, at Kingswood College in Grahamstown. The pupils worked in pairs, one pair per machine, using the curve sketching program. They worked through standard textbook example problems, such as, sketch the curves of:

$$x^3-4x^2+4x; \quad -(x+1)^3+8; \quad 1-x^3; \quad x+1/x; \quad 1/x^2$$

(the latter two problems were intended for the Higher Grade pupils only).

/4.2.1.1.2 Reactions ...

4.2.1.1.2 Reactions of Pupils

The reactions of the pupils were recorded by means of a brief questionnaire (see Appendix A6). There were twenty-two participants and each but one submitted a completed questionnaire to the class teacher the following day. All but two respondents (Standard Grade pupils) found the program easy to use. Two others (Higher Grade pupils) were irritated by the requirement that each input to the program had to be terminated by means of a semi-colon. The central question was, "Do you think it (the curve sketching program) could help you to learn curve sketching?" Here is a summary of the responses:

	YES	**NO**
HG	5	7
SG	6	3

Table 4-1: Responses to the Question, "Do you think the curve sketching program could help you to learn curve sketching?"

The reasons given for the responses can be summarized as follows:

/Table 4-2: ...

'YES' responses	HG	SG
Useful answer checking device	3	1
Facility for practising	2	1
Aids understanding		2
Adds interest		1
Blank		1
'NO' responses		
Does everything for you	7	
Doesn't explain		3

Table 4-2: Reasons for the Responses in Table 4-1

The 'NO' responses are interesting. Obviously the pupils would have preferred a tutoring program; they were put off by the fact that the program functioned rather as a curve sketching expert. It is heartening to note, however, that seven pupils saw merit in the facility provided by the program for answer checking and practice on an unlimited range of examples - these were two of the design criteria. Three of the seven Higher Grade pupils who protested against the computer doing everything for them and one of the Standard Grade pupils who wanted explanations, also recorded, as a general comment, that the program was a handy checking aid.

In answer to the question, "Would you like to make more use of the computer as a mathematical problem solving aid?", the responses were:

/Table 4-3: ...

	YES	NO
HG	8	4
SG	6	3

Table 4-3: Responses to the Question, "Would you like to make more use of the computer as a mathematical problem solving aid?"

4.2.1.2 An Equation Solving Program

As mentioned above, the tutoring component of the system is as yet not fully developed. A pilot version restricted to equation solving, one of the faculties of the expert component, is, however, working successfully. It makes use of the techniques of 'Attraction', 'Collection' and 'Isolation', embodied in re-write rules as propounded by Bundy (1983), for the purposes of step-by-step equation solving. It is also able to produce step-by-step explanations. The program, the way it functions and the theory on which it is based are described in some detail in Halpin and Marsh (1988). Some attention will be devoted here to a discussion of the possible educational benefits of working with this equation solving tutorial program.

The program under discussion is able to solve, with explanations, all linear equations, quadratic equations of the form

$$ax^2 + b = c$$

and transcendental equations of the form

$$p + q\ln^n(f(x)) = r,$$

where p, q and r are real, n = 1 or 2 and f(x) is any linear function or any quadratic of the form

$$ax^2 + b.$$

This solving is achieved through the application of re-write rules such as

$$U + V = W \leftrightarrow U = W - V,$$

$$U^2 = V \leftrightarrow U = \pm\sqrt{V},$$

and $$\log_U V = W \leftrightarrow V = U^W.$$

It is in the process of identifying all the necessary re-write rules and establishing procedures for controlling their use, that one first realizes the educational value of this form of programming activity - both the construction and the use of such a program. The reasoning involved takes place at what Bundy and Welham (1984) call the meta-level:

> **"... inference is conducted at two levels simultaneously: the meta-level and the object-level. The object-level encodes knowledge about the facts of the domain (in this case rules of algebra), while the meta-level encodes control or strategic knowledge (in this case methods of algebraic manipulation)."**

Meta-level rules guide the use of object level rules, and the cognitive framework which is thus characterized enables the formation of explicit problem solving strategies. It is contended that exposure to the kind of meta-theory which arises in this context would provide an enriching

/opportunity ...

opportunity for teachers, and hence their pupils, to develop an effective overall perspective on, and sensitivity to, problem solving processes.

The way the equation solver works is best illustrated by means of an example. Asked to solve

$$1 + 2 \ln [(x^2 - 2)/3] = 4$$

the program would go through the following steps:

Step 1: Apply the 'inverse-plus' rule to obtain

$$2 \ln [(x^2 - 2)/3] = 4 - 1 = 3$$

Step 2: Apply the 'inverse-times' rule to obtain

$$\ln [(x^2 - 2)/3] = 3/2$$

Step 3: Apply the 'inverse-log' rule to obtain

$$(x^2 - 2)/3 = e^{3/2}$$

Step 4: Apply the 'inverse-divide' rule to obtain

$$x^2 - 2 = 3e^{3/2}$$

Step 5: Apply the 'inverse-minus' rule to obtain

/x^2 ...

$$x^2 = 3e^{3/2} + 2$$

Step 6: Apply the 'inverse-square' rule to obtain

$$x = \pm\sqrt{[3e^{3/2} + 2]}$$

By isolating the unknown in this way the program emulates a human equation solver. For this reason and because it is able to explain its heuristics on request, the author believes that it could be a useful teaching tool. The teacher could put it to good use in several possible ways: as a means of introducing and explaining equation solving methodology; as a means of providing drill-and-practice in equation solving; as a handy 'ready reckoner' against which a pupil can check his own work; and as a medium within which a pupil can explore equation solving on his own and learn by discovery.

4.2.1.3 The Tutee Mode of Program Construction

Returning to the notion of intelligent tutoring, a point that bears stressing is that in order to get the computer to function effectively in the tutor mode, one has to program it well in the tutee mode as propounded by Taylor (ed.) (1980). In fact, the computer has to be 'taught' to solve problems in such a way that its reasoning is human-like and that it can explain this reasoning when asked to do so. It is in the process of 'teaching' the computer in this way that a teacher's own problem solving strategies are made absolutely explicit and, as a consequence, his teaching methodology can become greatly improved. Techniques which hitherto were known and applied in a largely instinctive way are forced to the surface and can be revealed to pupils in an illuminating way.

CHAPTER 5 COMPUTER GUIDED TEACHING (CGT) IN MATHEMATICS

CHAPTER 5

COMPUTER GUIDED TEACHING (CGT) IN MATHEMATICS

5.1 The Nature of Computer Guided Teaching

A relatively new and largely undeveloped application of the computer in education is that of so-called computer guided teaching (CGT). It refers to the situation in which the computer has been programmed to guide the teacher through all the steps involved in the presentation of a particular topic. The program divides the topic into subtopics, connects them smoothly, summarizes them at regular intervals, and provides questions to ask the students, or examples and problems for the teacher to discuss with them. In addition, it provides the teacher with the facility for presenting, in the appropriate places, features which are usually supplied as separate programs. Such features are concept demonstrations, simulations, experimentations, fast computations, real-life applications and educational games.

Extensive time, effort and expertise have to be invested in planning such software-controlled lessons - a great deal more than a single teacher can put into the planning of his own lessons. Under the pressure of real classroom situations, teachers are usually unable to exploit the full range of potentially effective teaching strategies known to them. CGT can help to alleviate this problem.

The use of CGT programs can save teacher time and effort before, during and after the presentation of a lesson. Before presentation there is no need for the teacher to prepare demonstration aids or to read several sources for enrichment purposes. During presentation the teacher does not have to write much on the chalkboard, or explain verbally, concepts which are clearly and dynamically presented by graphical pictures and animations. After lesson presentation, pupils who have missed the lesson, or who need to review the material, can work with the software instead of making demands on the teacher. Once 'canned', a CGT lesson can be used over and over again.

Another very real benefit offered by CGT software is that it can assist the teacher of mathematics who is inexperienced or in any way ill-prepared.

5.2 A Sample Computer Guided Teaching Program

Hativa (1984) describes one of a sequence of CGT programs which have been designed to aid mathematics teachers in the introduction of topics in Euclidean Geometry. The program, *SPECIAL PARALLELOGRAMS*, incorporates a whole range of effective teaching strategies as discussed below.

SPECIAL PARALLELOGRAMS is structured into three components: introduction, body and conclusion. It starts with a review, then provides a rationale as to why the topic should be studied. From here it branches to each of the three special parallelograms: the square, the rectangle and the rhombus. It ends with a summary of the material taught. An overview of the lesson stages is provided by the menu.

The program guides the teacher and pupils interactively into a discussion of the properties of parallelograms. Once these properties, such as opposite sides are congruent, consecutive angles are supplementary, etc., have been established, a further five properties, such as consecutive sides are congruent, etc., are presented and checked for validity through animated illustrations. Eventually the fact is established that the general parallelogram does not have any of these five additional properties, but that there are special parallelograms which have some or all of these properties. Intuitive understanding of the interrelations between special parallelograms and the general case is precipitated by the drawing of the family of parallelograms shown in the figure below. The program draws this picture on the screen.

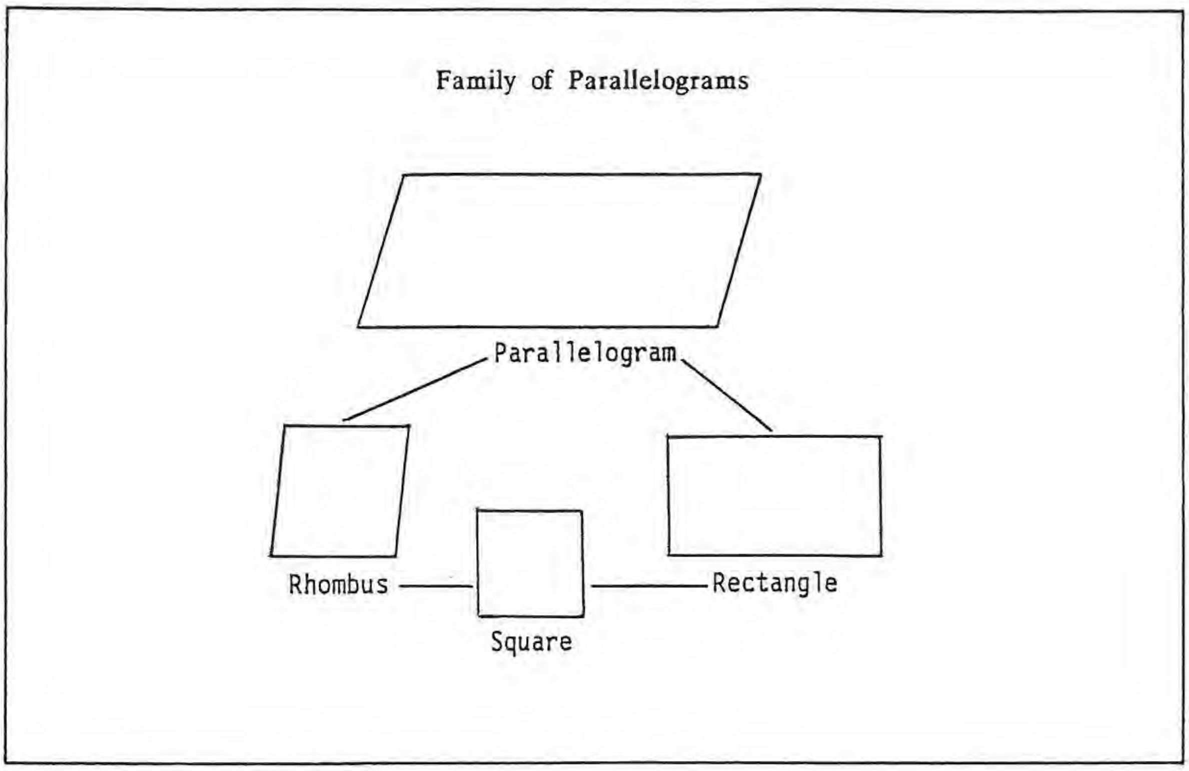


Figure 5-1: **Picture on Screen: Interrelations between Special Parallelograms and the General Case**

Two equivalent definitions are provided for the rectangle and two for the rhombus, while four are provided for the square. One of the definitions for each of these special parallelograms is a minimal definition and pupils are asked to explain how the other definitions can be deduced from this one. In this way the program leads the teacher into discussing the important notion of equivalent definitions - a basic concept which is often overlooked in the teaching of mathematics.

Next, the properties of the special parallelogram under consideration which distinguish it from the general parallelogram, are emphasized by means of diagrams such as that in the figure below. Finally, there is an animation which provides step-by-step hints for the proof of a theorem concerning a particular property, such as that the diagonals of a rectangle are congruent.

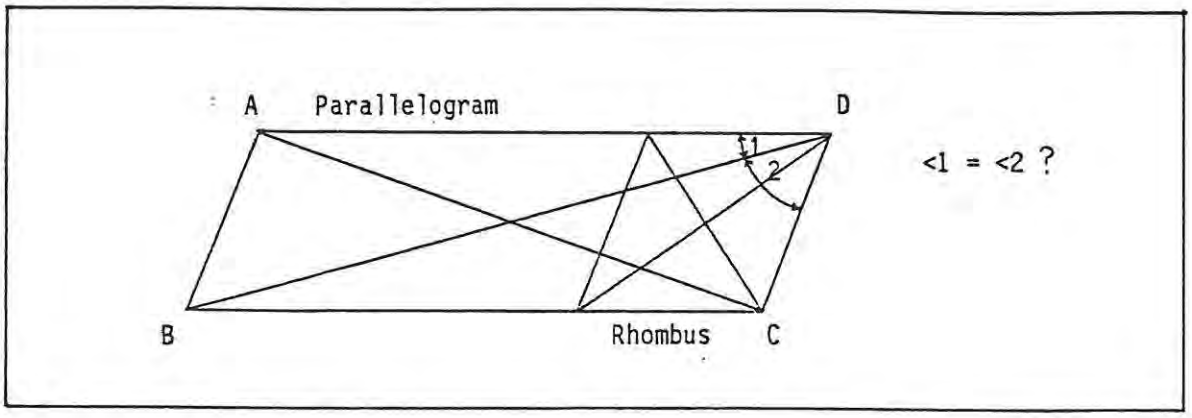


Figure 5-2: **Picture on Screen: Animation providing Step-by-Step Hints for Theorem Proving**

5.3 Features of Computer Guided Teaching Programs

CGT programs like the one described above make extensive use of animations, colour, aesthetic graphics, and sound to develop pupils' imagery and intuition. They encourage the teacher-user to apply beneficial teaching strategies, such as asking questions at various cognitive levels, structuring and smoothly sequencing the presentation, and establishing strong links with previously acquired knowledge.

An attractive feature of CGT software is that it creates an environment for three-way interaction, between teacher, pupils and computer. It dispenses with the traditional CAI stereotype: a single pupil interacting with a single terminal. In this traditional situation the teacher is often no more than an aide to the computer instead of vice-versa. The CGT approach allows the teacher to use the computer as *his* aide in providing quality in-class teaching.

5.4 The Computer as a Demonstration Aid

A related mode for computer usage in the mathematics classroom is that of the computer as a demonstration aid. Hativa (1984) lists three examples of this mode:

- Graphical illustrations of concepts:

  The teacher illustrates the derivative concept, for instance, by using software that plots, point by point, the numerical values of slopes of tangents to the original function and then connects them with a continuous curve.

- Numerical illustration of concepts:

  The teacher illustrates a limiting process, for instance, by using software that computes quickly and prints the results of each step of the computation.

- Experimentation:

  When teaching the graphing of functions, for instance, the teacher can easily demonstrate what happens to the graph of f(x) when a constant is added to the function, as in, for example, f(x)+5, f(x)-2, or the argument, as in f(x+5), f(x-2). Similarly, the effect on the graph of multiplying the function or the argument by a constant can also be explored, as in the cases 5f(x), (½)f(x), f(5x), and f((½)x).

Two of the above three examples of the computer used in demonstration mode will be given added substance elsewhere in this study. Numerical illustration of concepts will take the form of an investigation concerning convergence of series using an electronic spreadsheet (see section 7.2.2.6 of Chapter 7). Experimentation will be evident in the use of Logo programs to explore the effects of the changes in amplitude and period on the graphs of trigonometric functions (see section 8.3.2 of Chapter 8).

5.4.1 Case Studies of the Computer in Demonstration Mode

5.4.1.1 Using a Graph Drawing Program

van Hille (1986), under the author's supervision, carried out a case study in which he tried out the use of the computer as a demonstration aid. He used a program, designed and written by Collett (1985), as an aid in reviewing the graphing of straight lines, circles, hyperbolas and parabolas. The program, which is rather similar to *GREEN GLOBS* (see section 6.2.6 of Chapter 6), has two main components:

- a graph drawing facility, which allows the user to display a graph by typing in the values of the coefficients in the defining equation given in standard form (an aspect of computer-supported teaching);

- a graph game in which three points selected by the program are shown on the screen, and the aim is to choose a type of graph and then input the relevant coefficients so that a minimum number of lines and curves pass through the given points.

The findings were largely positive. This statement can be supported by quoting comments made by some of the pupils:

- "Computers ... stimulate interest and concentration."

- "The graphs were explained much better and were easier to understand."
- "This method should be used more often."
- "This lesson helped me to understand more about graphs ..."
- "It was interesting and different. It would be nice if we could have more of these lessons."
- "This should be done on a regular basis. It was enjoyable."

Some of the researcher's evaluative comments (van Hille, 1986) are also worth recording:

- "The pupils seemed to enjoy the lesson and were very alert as a result of the fact that a different approach was being used."
- "The correct balance between normal teaching (this includes interaction between pupil(s) and teacher and written work by pupils) and computer-teacher-pupil interaction is very important."
- "Some pupils seemed to be mesmerized by the action on the computer screen rather than trying to understand why [that is, interpret the underlying mathematics]."

This last statement sounds a rather sobering note of warning: allowing the medium to obscure the message must be guarded against.

As with CGT, a great advantage offered by the computer in demonstration mode is that pupils can use it on their own after hours. In the research project discussed above it was found (van Hille, 1986) that:

> **"... class time [was] too short for pupils to take full advantage of the program. ... However, on the [pupils'] own suggestion they were given an opportunity to work on the program by themselves. They tended to do this in small groups and would spend about an hour systematically working through the program. Their comment [invariably] was that they found it too rushed in class, but that**

they had now gained real understanding. Watching a group working on their own and noting their confidence and competence was most pleasing."

5.4.1.2 Using a Program Demonstrating Pascal's Triangle and Related Sequences

The author used the computer in demonstration mode to teach a lesson on Pascal's triangle and related sequences. The program used was part of a collection called *SEQUENCES SET* by D. Baldwin, published by Chalksoft Ltd. (undated), and the lesson was taught to standard 7A (the top set) at St. Andrew's College in March 1989. The class consisted of eight boys (St. Andrew's College) and five girls (Diocesan School for Girls, the sister school to St. Andrew's College). The experiment was conducted along action research lines, the researcher being a participant observer, and responses being collected via questionnaires from the class's regular teacher (a non-participant observer) and the pupils themselves - a form of triangulation.

The first part of the program was in the form of a tutorial on various sequences, including the Fibonacci sequence, prime numbers, square numbers and triangular numbers. The second part of the program consisted of a tutorial on Pascal's Triangle.

Following the stimulus provided by the program the author had no difficulty in getting the pupils to abstract the sequences of natural numbers, triangular numbers, tetrahedral numbers and pyramidal numbers from Pascal's triangle as shown in the diagram below.

/Figure 5-3: ...

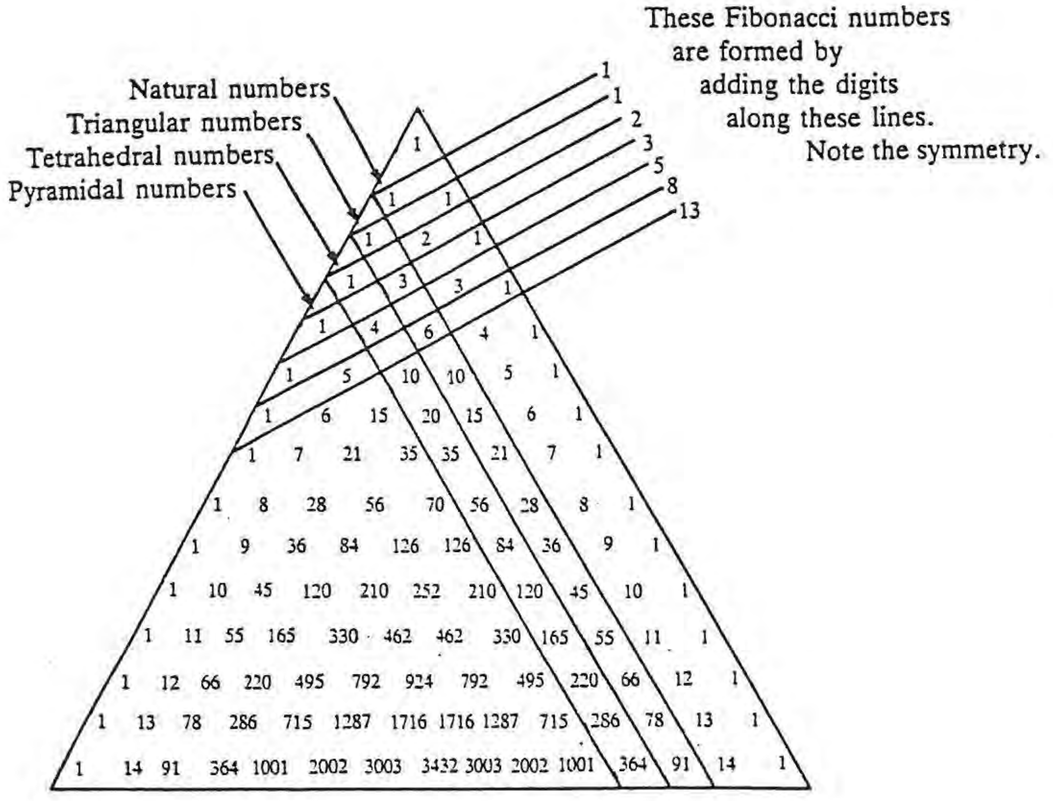

These Fibonacci numbers are formed by adding the digits along these lines. Note the symmetry.
Natural numbers
Triangular numbers
Tetrahedral numbers
Pyramidal numbers
1
1
2
3
5
8
13

Figure 5-3: **Pascal's Triangle and Related Sequences**

This part of the lesson was concluded with an explanation by the author of probabilities of various boy-girl combinations in families of different sizes, based on the coefficients shown in the different rows of Pascal's triangle.

Finally, after an explanation of Fibonacci numbers via a step-by-step development of the well-known breeding rabbits lineage, it was most illuminating for all concerned, teacher and researcher included, to see how the Fibonacci sequence could be generated from the information in Pascal's triangle (see Figure 5-3 above).

/The class ...

The class responded in a particularly positive way to this lesson. All the pupils recorded that they had enjoyed the lesson and would like to do more computer-aided mathematics. Only one pupil stated that the work with the computer had not helped him to understand the content of the lesson better than he might have otherwise. The others all felt that the program had enhanced their understanding. One drawback of the program, commented on by several of the pupils, was that it had been rather slow and drawn out in places.

The non-participant observer (the class's regular teacher) also responded very positively, saying that the lesson had been a great success. He had a keen interest in the use of the computer as an educational aid in the mathematics classroom, and, on this occasion as well as others when similar collaboration occurred, he asked that the software be loaned to him so that he could pursue the topic and approach of the lesson further with the class.

CHAPTER 6 COMPUTER GAMES AND SIMULATIONS IN MATHEMATICS

CHAPTER 6

COMPUTER GAMES AND SIMULATIONS IN MATHEMATICS

6.1 Introduction

In this chapter we shall consider two effective and popular modes of computer-stimulated mathematics education: games and simulations. A variety of programs which exemplify these classifications has been produced. We shall look at a small subset of these, several generated by the British Investigations on Teaching with Microcomputers as an Aid (ITMA) project. The games chosen are *ERGO*, *PIRATES*, *SUBGAME* (all by ITMA), *VECTORS* (by Cambridge Micro Software), *EQUATORS* (by NASOU), *GREEN GLOBS* (by CONDUIT), *MONTY* (by SLIMWAM) and *LINEOVER* (by SMILE). The simulations are *DICECOIN* and *EUREKA* (both by ITMA), and *NUMBER MACHINE* (by NASOU).

The essential difference between game and simulation programs (as explained in Chapter 9) is that: game programs encourage strategic thinking and provide opportunities for the practising of skills and developing of concepts in recreational ways; whereas simulation programs provide good approximations to real-world situations, allowing the user to experiment and solve problems in these simulated environments safely and efficiently. There are, of course, simulation games in which the characteristics of games and simulations are combined (see, for example, section 6.2.4.1 below).

The abovementioned example programs will be taken in the order in which they are listed and each one will be analyzed to find what it does and the educational value it possesses. The assessment of educational value will be based on the author's own perceptions as well as, in most cases, on the results of evaluations done by a sample of student teachers and practising teachers, as explained in section 1.6 of Chapter 1.

6.2 Some Mathematical Computer Games

6.2.1 The ITMA Game *ERGO*

ERGO, designed and programmed by R. Phillips, involves the filling in of numbers in a 5x5 grid to establish a pattern. The user selects one of two levels of difficulty; the program selects a pattern from among 1 000 possibilities and displays just two of the numbers on the screen.

The user's task is to discover the pattern and so fill in the missing numbers one at a time, controlling cursor movement from the keyboard. A given response is either inserted if correct, or judged to be too large or too small so that another attempt can be made. Points are scored for getting numbers right first time.

This game, which is in the nature of an alluring problem, aids the development of a pupil's understanding of sequences, especially arithmetic sequences, and gives him practice at working with such sequences.

6.2.1.1 An Evaluation of *ERGO*

The majority of teachers in the sample agreed that this game could be used to give pupils practice at pattern perception and working with sequences. The student teachers came to the same conclusion and recognized the computer's advantage over traditional means for generating examples for individual pupils.

6.2.2 The ITMA Game *PIRATES*

PIRATES, designed by R. Fraser et al. and programmed by C. Wells and M. Allnut, is a bearing-directed 'treasure hunt' on a co-ordinate grid. The 'treasure' is located at a point on the grid randomly selected by the program. The user tries to find this hiding place by making guesses expressed in co-ordinates. The program judges each guess and supplies a clue in the form of the compass bearing from the point guessed to the point at which the 'treasure' is situated. For each new search the initial amount and type of 'treasure' is determined by the program. During a search the amount of 'treasure' is diminished (stolen by the pirates!) with each unsuccessful guess. After ten tries the 'treasure chest' is empty.

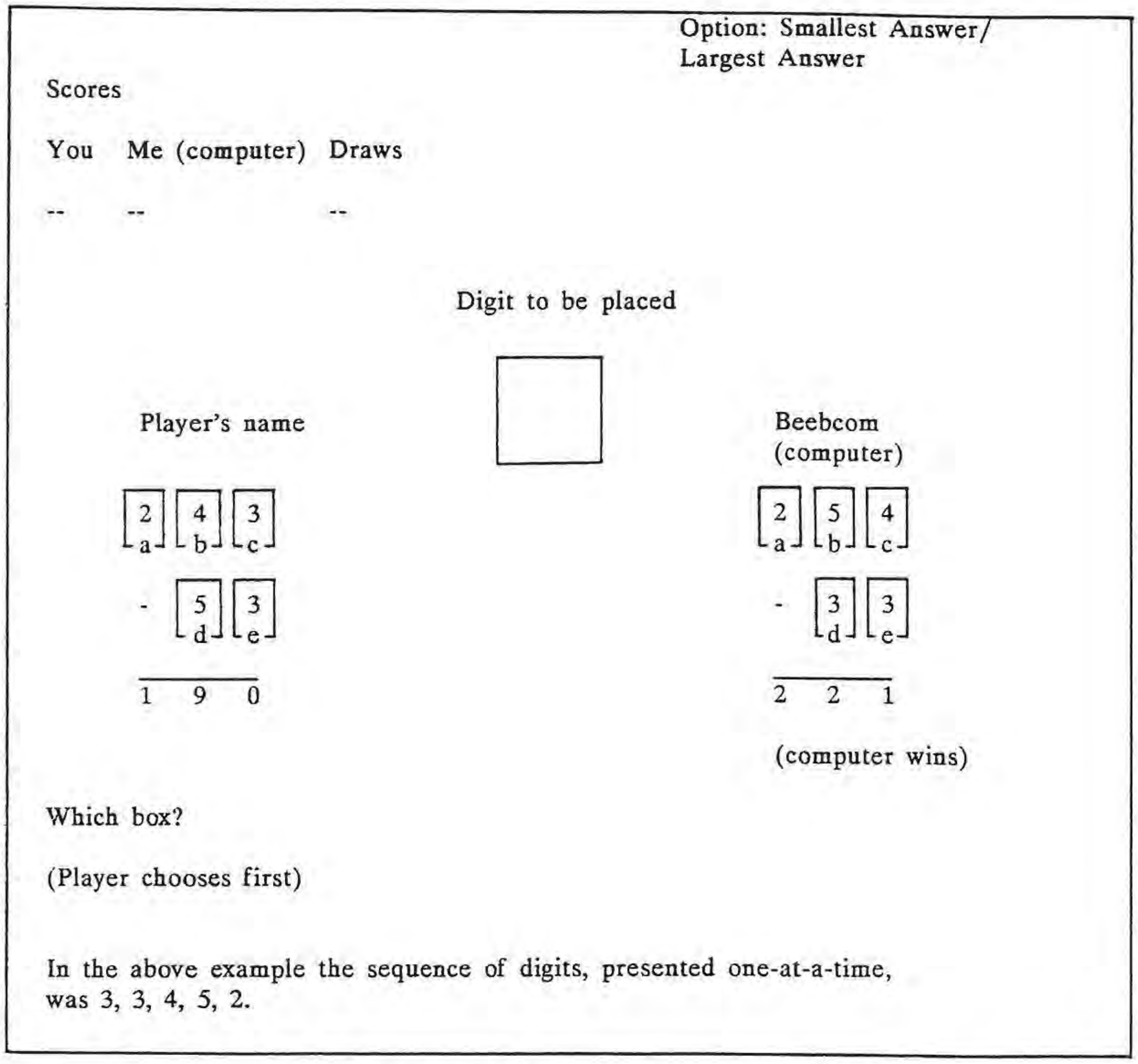
Using this program pupils can gain valuable practice in: the use of two-dimensional co-ordinate representations of position; compass work; the plotting of data; the construction of hypotheses; and the formulation of best strategies for an investigation.

A similar game called *TREASURE HUNT* is described and evaluated in section 6.2.4 below.

6.2.3 The ITMA Game *SUBGAME*

SUBGAME, designed and programmed by B. Ives, involves a competition between the user and the computer in which each attempts to allocate digits - randomly generated by the computer - to a subtraction 'sum', so as to maximize (or minimize) the result. When all digits have been allocated, the computer calculates its own result, the user types his result, and the winner is the player with the largest (or smallest) answer. The computer displays a record of rounds won, lost and drawn. A typical screen display is shown below:

/Figure 6-1: ...



Option: Smallest Answer/
Largest Answer

Scores

You Me (computer) Draws

-- -- --

Digit to be placed

Player's name

$$\begin{array}{rccc} & \boxed{2}_a & \boxed{4}_b & \boxed{3}_c \\ - & & \boxed{5}_d & \boxed{3}_e \\ \hline & 1 & 9 & 0 \end{array}$$

Beebcom
(computer)

$$\begin{array}{rccc} & \boxed{2}_a & \boxed{5}_b & \boxed{4}_c \\ - & & \boxed{3}_d & \boxed{3}_e \\ \hline & 2 & 2 & 1 \end{array}$$

(computer wins)

Which box?

(Player chooses first)

In the above example the sequence of digits, presented one-at-a-time, was 3, 3, 4, 5, 2.

Figure 6-1: ***SUBGAME*** **Screen Display**

This game can be played by one pupil at a time or by a whole class at once, each pupil working individually and recording his own attempts on templates supplied by the teacher.

Although this game is most appropriate for pupils at the Senior Primary level, a fruitful Junior Secondary level exercise arising from it is the investigation of the strategy used by the computer. Pupils could either discover this strategy or have it given to them (as shown below). They could then be encouraged to suggest possible improvements and to test their ideas by implementing them in their own strategies against the computer.

Digit	Order of search for an empty place (to create largest possible answer)				
1					
2	d	e	c	b	a
3					
4	e	d	c	b	a
5					
6	c	b	a	e	d
7					
8	a	b	c	e	d
9					

Table 6-1: ***SUBGAME*** **Strategy**

SUBGAME can provide a useful and stimulating environment in the classroom, promoting class discussion, individual pupil work and development of a gaming strategy. It affords pupils, even those who are weak at Junior Secondary level, an appealing way of practising subtraction skills.

6.2.3.1 An Evaluation of *SUBGAME*

This game was seen to be a stimulating way to practise subtraction skills in the primary school by most of the teachers in the sample. Most of the student teachers, although agreeing with this opinion, did not think that they would use it in the classroom. Such a game, most respondents concluded, was best played on a computer but the underlying mathematical concepts could be as easily or better taught without this particular program. Interestingly, none of the evaluators commented on the value of considering the underlying gaming strategy.

/6.2.4 The Cambridge ...

6.2.4 The Cambridge Micro Software Game *VECTORS*

VECTORS, designed by D. Kite and programmed by C. Forecast, is a program prefaced by a CAI section on vector concepts and containing three games, each for the purpose of teaching and reinforcing the understanding of a vector as a movement. Through using the program the pupil should master the use of vector notation (for a two-dimensional movement) and gain valuable spatial experience through using vectors to achieve success in the games. The first two games, *NAVIGATION* and *TREASURE HUNT*, are based on a visible grid similar to that in *PIRATES* (discussed earlier), but references are given in vector form, $\binom{x}{y}$, as opposed to co-ordinate form, (x;y). The third game, *RACING DRIVER*, provides a scale but no grid.

In *NAVIGATION* the user must 'navigate' using vectors from a port to a harbour avoiding the islands. The object is to make the total length of the journey as short as possible. The use of decimal co-ordinates in the vectors they choose allows pupils to pass closer and closer to the islands and so reduce the journey length even more.

In *TREASURE HUNT*, which is similar to *PIRATES* (see section 6.2.2), the user searches for the hidden 'treasure' using vectors and trying to avoid barriers. A thermometer indicates how close he is to the treasure at each stage. The map and the position of the treasure change from game to game.

In *RACING DRIVER* the user 'drives' a car round a track trying not to crash into the inner and outer perimeter barriers.

/6.2.4.1 An Evaluation ...

6.2.4.1 An Evaluation of *VECTORS*

All the teachers and student teachers in the sample identified the CAI section as a teaching program and most thought that it could be useful as an entertaining introduction to vectors. Although the computer was not necessary in introducing the vector concept, this program had the advantage of effective visual impact.

NAVIGATION and *TREASURE HUNT* were seen by both teachers and student teachers as being simulation games which provide the pupils with drill-and-practice at vector description of movement. The student teachers in particular saw the value of these simulations as being providers of real-world contexts for vector problems. Although the computer was not necessary in teaching this aspect of vectors, both groups acknowledged that the gaming approach would have great appeal for the pupils.

The teachers agreed that *RACING DRIVER* was a simulation game which also provided pupils with drill-and-practice in the basic concepts of vectors, but in this instance the program could be used to help pupils estimate distances and familiarize them with the cartesian plane. Thus the program could be a useful aid in the teaching of co-ordinate geometry. The student teachers maintained that they would use it as a fun way of practising vector skills. Once again both groups agreed that the gaming aspect of the program had an advantage over traditional means of practising such skills.

6.2.5 The NASOU Software Game *EQUATORS*

EQUATORS, designed by P. Human, programmed by C. Joubert and published by NASOU, is a "space invaders" game of the "arcade type" which reinforces equation solving skills. It supports the development of computational and algebraic skills, as well as the understanding of variables, equations, algebraic expressions and the function concept. A large number of variations is available to meet the needs of pupils at a variety of levels. While progressing through exercises of increasing speed and difficulty, the player is compelled and motivated to search for better methods of solving equations.

The player repeatedly has to type in a number for which the 'defender' (an algebraic expression) is equal to an 'invader', either in the form of a number or another algebraic expression. The defender exterminates the invader directly above it the moment the correct solution is found. The defender, which remains the same throughout an exercise, may be moved to the right or left to confront different invaders. The invaders gradually move downwards and if one of them reaches ground level (where the defender is), the player loses the game. Scoring is based on the number of invaders shot down, as well as time elapsed. When the game is frozen to enable calculations to be carried out, time is 'bought' at the cost of valuable points.

6.2.6 The CONDUIT Game *GREEN GLOBS*

This last example of a mathematical game program was the basis for a case study carried out by the author at St. Andrew's College in Grahamstown. A full report, based on the paper by Marsh and Schafer (1989), follows.

6.2.6.1 Description of Program

GREEN GLOBS is a program for the Apple II microcomputer which deals with the graphing of equations. The authors of the program are S. Dugdale and D. Kibbey (1983) of the University of Illinois. The mathematical concepts covered by the program are standard components of high school algebra or analytical geometry. A variety of functions is incorporated, including linear, quadratic, absolute value, square root, logarithmic and exponential. Other standard graphs such as circles can also be drawn.

The computer displays co-ordinate axes with thirteen "green globs" (disc-shaped neighbourhoods of points) scattered randomly. Pupils then enter equations for graphs they think will 'hit', that is, pass through, these globs. When a glob is hit, it 'explodes' and disappears. When a shot misses the expected target(s), the display of the graph gives the pupil useful diagnostic information - perhaps it is too wide, too steep, upside down, etc.

The scoring algorithm encourages pupils to hit as many globs as possible with each shot. For each graph, the first glob hit is worth one point, the second is worth two points, the third four, and so on. Thus a five-glob shot scores 1+2+4+8+16, a total of 31 points. This scoring algorithm puts a higher premium on maximizing the number of globs hit with each individual shot: it is worth a lot to pick up one more glob on a shot.

Figure 6-2, from Dugdale and Kibbey (1983), shows an example of the screen with an initial display of thirteen globs, followed by displays of a pupil's first three shots (frames (a) to (d) consecutively).

/Figure 6-2: ...

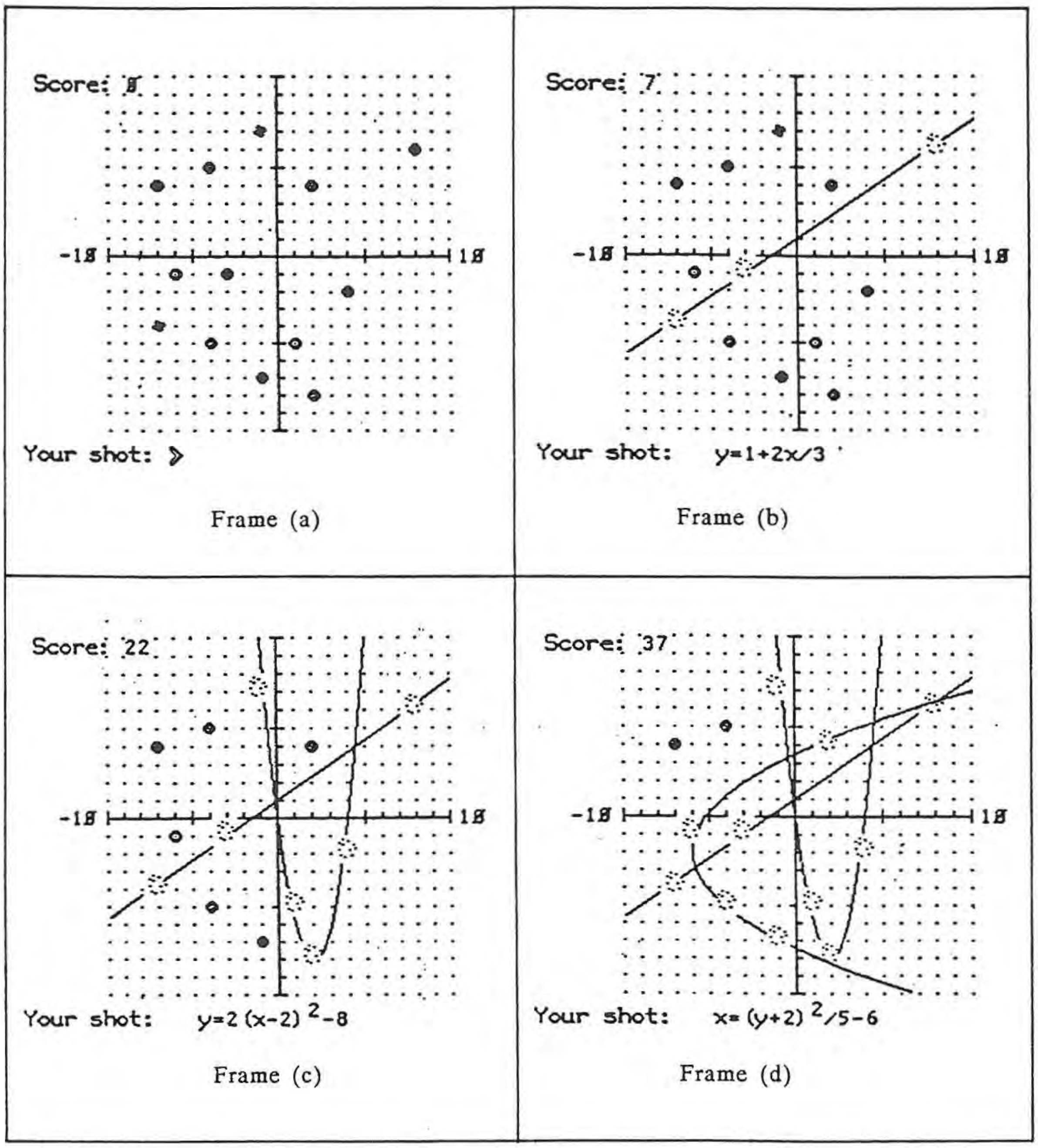


Figure 6-2: ***GREEN GLOBS*** **Screen Display: Straight Lines and Parabolas**

Pupils can enter functions beginning with 'y = ' or 'x = '. For example $y=x^2-3x+2$, $x=(y+2)(y-1)^3$, and $y=3+x^5+(5/x)-(x-1)^3/(x^2-x)$ are all acceptable. The program also recognizes conic sections, square roots, absolute values, logarithmics and exponentials. "Trig. wipeout"

/functions ...

functions like $y = 10\sin 5x$ are excluded for the obvious reason! In fact, because of possibilities like this, no trigonometric functions are allowed.

The top ten scores of all pupils who participate in the game are kept in a "hall of fame". The games which led to these scores are also stored so that they can be replayed by pupils who want to see what shots the top players have used, and so learn strategies which may prove useful in future games.

The overall design goal of the program developers (Dugdale, 1982) was:

> **"to provide students with a rich, mathematically accurate environment and the motivation to manipulate that environment to learn about graphs of equations."**

6.2.6.2 What Pupils Learn

GREEN GLOBS is appropriate for pupils of varying abilities. While it is clearly possible to hit all the globs with several linear functions (even constant functions), the more mathematical knowledge and skill the pupil applies, the more globs he is likely to hit per shot. Pupils are thus encouraged to increase their graphing skills. They are also encouraged to construct interesting graphs beyond those normally found in the syllabus. For example, a pupil might discover that the graph of $y=(x+2)(x-1)(x-3)$ crosses the x-axis at the points -2, 1 and 3. Using the co-ordinates of a target glob not on the x-axis, he might then find an appropriate adjustment to the graph to make it hit the desired globs. In the case study reported below, it was techniques just like this which were found to emerge.

/In Figure ...

In Figure 6-3, from Dugdale (1982), frame (a) shows the above function crossing the x-axis at -2, 1 and 3 and hitting four globs. In frame (b) the function has been stretched vertically by a factor of 20 and it now hits six globs.

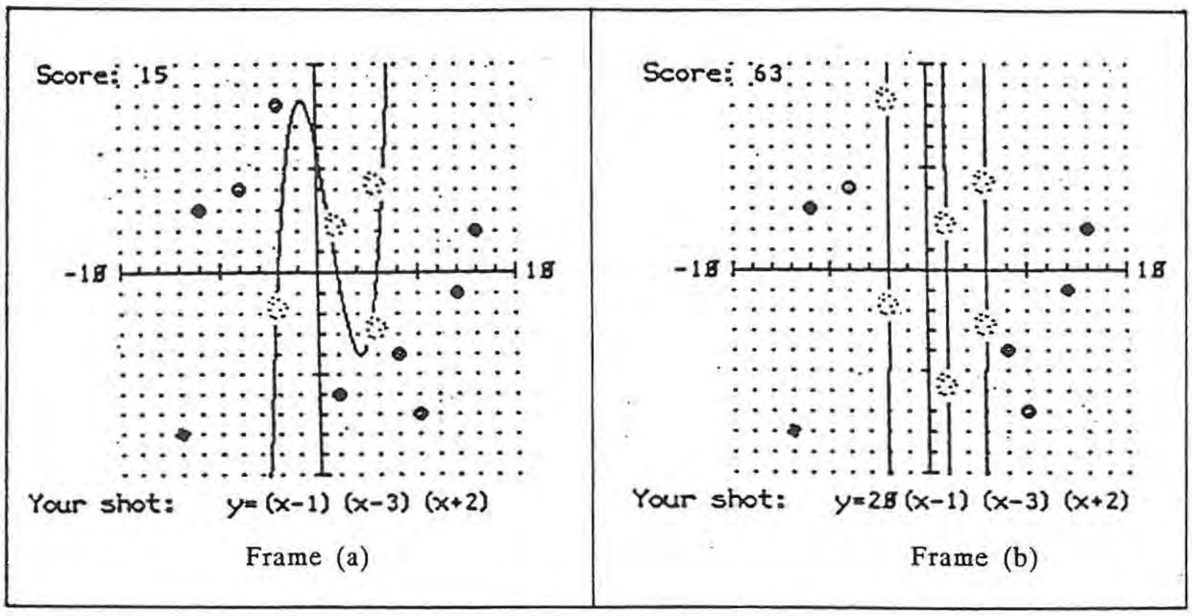


Figure 6-3: ***GREEN GLOBS*** **Screen Display: Oscillating Functions**

6.2.6.3 Case Study

The researcher ran a small-scale action research case study, involving the use of *GREEN GLOBS* by the standard 9A mathematics class (the top set) of the combined St. Andrew's College/Diocesan School for Girls in Grahamstown.

A single lesson period (thirty-five minutes) was devoted to an explanation and demonstration of the program to the class by the researcher. The class reacted well and participated fully, with the boys as a team taking turns to 'shoot' (that is, propose equations), playing against the girls as a team. In this initial exposure to the program the 'shots' permitted were limited to linear and quadratic functions and circles. The success of the lesson is evidenced by

/responses ...

responses made by the pupils to a questionnaire administered immediately afterwards. Some of the data gathered is summarized below.

There were seventeen pupils in the class, and to the question, "Would you like to do more computer-aided maths?", sixteen responded positively. To the question, "Did the work with the computer help you to understand your maths. better?", fifteen responded positively. The following are some of the free responses to the question, "What did you like most about the session?":

- "It gave us all a chance to work together as a class. It also helped me to understand parts of the parabola that I had not understood before. It was a fun way of doing maths. and showed me that there is a really enjoyable side to maths. It gave us a much needed break from the normal regular boring routine of everyday maths. but meant that we were still learning while we were having fun."

- "It was a welcome change from the usual bookwork and made drawing graphs a lot more interesting as there was a point in drawing them. I think that this kind of game is invaluable as it brings maths. alive - it is no longer just strings of numbers. I think I gained a greater understanding of circles, parabolas, etc., by having to draw them going through certain points, and definitely view them in a new light."

- "It was a change to normal maths. routine. It was an encouraging and inspiring exercise. I found [that for the class to be working on the computer as a group] was a new and exciting experience."

- "It is more fun to see the work being done for you instead of doing it yourself. The work goes faster and you can draw more graphs and therefore understand better."

- "It was interesting. It was quick - it doesn't take a long time to draw the graphs like it does on paper. The game makes it more interesting. It's different to normal maths. graphwork which can be boring."

- "The way that by drawing graphs one can score points by destroying the green globs. It puts fun back into maths. There is a strong element of discovery in the game."

- "It made me think and I could see [the equation I had chosen] without actually having to do the plotting and drawing myself. I enjoyed watching the globs being blown up and the score increasing. It was a nice change and a new approach to maths."

The other respondents said essentially the same things but in less fluent ways.

In answer to the question, "What did you not like about the session?", there were six null responses, and three respondents complained that the time had been too short - they had wanted to do more work with the program! Another three respondents would have preferred each to have had their own computer to play the game on individually. The slow execution of the program was criticized by four respondents, and two others said that, because of the distance they had been sitting away from the computer, they had had difficulty in seeing the screen clearly.

A form of investigator triangulation was used in the study, in that, in addition to the questionnaire administered to the pupils, and supplementary to the participant observer role played by the researcher, the class's teacher functioned as a non-participant observer. His findings are summarized below:

/- "The quality ...

- "The quality of the program is good. It enables the development of investigatory ability, encouraging the making and testing of predictions, and promoting awareness of inter-relationships among mathematical concepts. Progress through the program is logical and smooth but execution, because of the comparative limitations of the Apple II, is rather slow. The program is mathematically sound and educationally viable. The pupils were certainly highly motivated by it, enjoyed using it, and benefitted by their exposure to it."

What really pleased the researcher and the teacher, however, was that several pupils requested that more lessons be devoted to the use of *GREEN GLOBS* and also that they be allowed to use the computer to play the game outside normal classroom hours. Arrangements were made to facilitate these requests and the results were most gratifying.

In one of the follow-up lessons the drawing of curves was restricted to quadratic functions only. It was astounding to see how efficiently the pupils established the equations of the necessary parabolas to eliminate the green globs. A further restriction, introduced in a subsequent lesson, was that at least two of the parabolas used had to be functions of y, that is, of the form 'x = ...'. This turned out to be an instructive exercise.

In another lesson, the curves were restricted to circles and the pupils soon realized that in order to eliminate the green globs efficiently, circles with centres other than the origin had to be chosen. Once again they discovered in a fun way an aspect often only hinted at in our textbooks, that is, circles of the form:

$$(x-a)^2 + (y-b)^2 = r^2.$$

/The program ...

The program was also used in lessons with a standard 10 class and it was remarkable to see the high level of motivation which it generated. It was found that when the class was given totally free reign, that is, with no restrictions imposed as to choice of equation, the pupils were very keen to develop defining equations which would wipe out as many green globs as possible in one go. In an atmosphere of constructive discussion and experimentation they soon discovered that 'many humped' functions would serve their purpose. They realized that a polynomial of degree 2 has one 'hump', a polynomial of degree 3 has two 'humps', etc. It was most rewarding to observe how functions of the form $y=(x+2)(x+1)(x-1)(x-2)(x-3)$ crystallized out of pupils' discussions very quickly.

6.2.6.4 Conclusion

One of the valuable spin-offs from the use of *GREEN GLOBS* was that large sections of the mathematics syllabus could be covered in an enriching fashion without reference to a textbook. *GREEN GLOBS* is a tool which can be used for both introducing and revising graphing concepts and techniques. Assuming the availability of only one computer, the game can be played in the classroom in different ways. The class can be split into two or more competing groups, or the whole class can participate in the devising of strategies as a team effort. The latter was found to be the most successful and popular approach. Another variation is the splitting of the class into several groups, with each group being restricted to the use of just one equation type, a different type for each different group - for example, three groups with one restricted to the use of parabolas, another to circles and the third to absolute value functions.

GREEN GLOBS offers an opportunity for the discovery of concepts involving functions and graphs, both inside and outside the syllabus, in a recreational way.

6.2.7 The SLIMWAM Game *MONTY*

MONTY is a game in which the user inserts numbers to recreate part of a number pattern previously shown on the screen in grid form. The grids, each 10x10, are generated by the program to suit the level at which the user wishes to work. Once the user has had an opportunity to study a given grid, it is erased and he tries to recall a required part of it as demarcated by 'Monty the python'.

6.2.7.1 An Evaluation of *MONTY*

This program is a simulation game. The sample of teachers were positive that it would be of much use in promoting awareness of numerical patterns and the ability to form general rules from given data. The student teachers agreed with this opinion and both groups said that they would use it in preference to traditional means since the program's constant changing of grids would be very difficult to emulate effectively.

6.2.8 The SMILE Game *LINEOVER*

The *LINEOVER* program generates straight line graphs on the screen. It is in the format of a game in which the user plays against the computer. Turns are taken to specify straight lines, one-at-a-time, via their defining equations in the form $y = mx+c$. All lines, the computer's and the user's, are plotted on the same graph. The object is to select each new line to intersect as many of the existing lines as possible. Points are scored accordingly.

/6.2.8.1 An Evaluation ...

6.2.8.1 An Evaluation of *LINEOVER*

This drill-and-practice game was considered by the respondents to be of great value in plotting and interpreting straight line graphs. They appreciated the program's efficiency in drawing the graphs and the effectiveness of the resulting visual display. They agreed that in this instance the computer with its speed and accuracy had a distinct advantage over the chalkboard or the OHP.

6.2.8.2 Workshops with Standard 7 Pupils using *LINEOVER*

Two of the teachers who evaluated *LINEOVER* were so impressed by it that they asked the researcher to run workshops for their standard 7 classes. These were small classes of girls belonging to the Diocesan School for Girls (DSG). They were streamed according to ability, 7K being the top set and 7T being the bottom set.

Separate workshops were run, one for each of the two classes, and the feedback data gathered from the pupils via a questionnaire (that appearing in Appendix A2) was interesting.

Of the ten girls in 7K, eight said that they liked working on computers and would like to do more computer-aided mathematics. The other two felt that they could have achieved the same purpose more easily with pencil and paper. One of these, however, responded positively to the question, "Did the work with the computer help you to understand your maths. better?" There were seven other positive responses to this question and one pupil said that she had "got a bit confused".

/In response ...

In response to the question, "What did you like most about the session?", nine of the girls said that they had enjoyed the challenge of the game. One of them made an illuminating comment:

- "While you were playing games you were also learning and the whole thing wasn't so serious where you had to concentrate, but you did because it was fun."

Only one pupil responded adversely to the question, "What did you not like about the session?". She had not grasped the concepts of gradient and intercept prior to using *LINEOVER* and as a result did not benefit from the drill-and-practice with these concepts which the program was designed to offer.

The second workshop was run for thirteen pupils in the bottom set, 7T. Of these pupils, eleven said that they liked working with computers and ten said that they would like to do more computer-aided mathematics. There were seven girls who felt that the work with *LINEOVER* had helped them to understand straight line graphs better than they had before.

One of the responses, given jointly by two pupils, to the question, "What did you like most about the session?", is worth recording:

- "We liked the fact that the computer did most of the work! We could just experiment without having to work out all the sums and draw the lines, therefore we got a lot further and learnt a lot more."

There was only one adverse response to the question, "What did you not like about the session?", and this was from a pupil who claimed that, "Maths. bores me to death."

/Overall ...

Overall, the response from both groups of pupils was positive. A follow-up communication received from the 7T teacher, however, brought a new and rather sobering perspective to the *LINEOVER* evaluation. She wrote as follows:

- "Some two weeks later I gave the std. 7s who worked through *LINEOVER* an old exam. paper to do. One of the questions required two graphs (str. line) to be drawn. **Not one** pupil drew the graph correctly! Quite a few did not even make an honest attempt - said they didn't know what was expected of them - interesting!"

6.2.9 The SMILE Game *BOXED*

BOXED is a game involving order relationships among numbers. The user plays against the computer. Each has a series of boxes, shown on the screen, into which numbers, generated one-at-a-time by the program, must be placed in sequence according to order of magnitude.

6.2.9.1 An Evaluation of *BOXED*

The student teachers classified this program as a drill-and-practice game and most agreed that they would use it to provide practice for pupils in ordering numbers. One student however was not sure of the program's objectives and therefore could not see its potential use.

6.2.10 The SMILE Game *DARTS*

The program called *DARTS* simulates the throwing of darts at a dart-board shown on the screen. The player selects his throw at each stage according to the number of points required to reduce a running total from 501 down to 0. Possible selections are: Bull's Eye = 50; Outer Bull = 25; Single, Double or Treble of any number from 1 to 20; Miss = 0. The program controls the accuracy of the throws. The player must do his own arithmetic (checked by the computer) to assess his position after each set of three throws.

6.2.10.1 An Evaluation of *DARTS*

All the student teacher evaluators saw merit in this game for providing pupils with drill-and-practice in addition and subtraction in an entertaining way. The computer's simulation of the dart board could not be safely nor efficiently emulated by other means.

6.2.11 The SMILE Game *MINIMAX*

In *MINIMAX*, a game similar in concept to *SUBGAME* (see section 6.2.3), digits generated by the program must be placed in turn by the user and the computer, playing against each other, in such a way that a sum, difference, or product (depending on the user's choice) of one 3- and one 2-digit number is either maximized or minimized.

6.2.11.1 An Evaluation of *MINIMAX*

This program gives pupils drill-and-practice in arithmetic manipulations. It uses a gaming approach to supply problems and optimal step-by-step response. The evaluators maintained that they would find the program useful since it saved the teacher a good deal of time which could be better spent giving individual attention to pupils.

/6.2.12 The SMILE Game ...

6.2.12 The SMILE Game *TAXI*

TAXI is such that two players, or teams of players, play against one another in trying to maximize the profit made by one of two opposition companies. In response to each call for a taxi each company quotes a fare, depending on the starting point and required destination of the trip shown on a street map on the screen. The location of the taxi at the time of the call must also be taken into account. The customer always selects the taxi for which the lower fare is quoted. In the event of equal quotes the program awards the transaction to one of the companies in an unbiased way. For each trip, profit is calculated according to the difference between the fare and the cost of running the taxi.

6.2.12.1 An Evaluation of *TAXI*

This game was described by the respondents as affording pupils drill-and-practice in estimation and simple arithmetic. They did not see much use for it and thought that such concepts could be as easily taught by using traditional teaching methods.

The researcher happens to know, however, that *TAXI* is being put to good use by a teacher working at standard 3 level at Kingswood Junior School in Grahamstown. Her opinion is that it encourages pupils to think of more than one thing at a time and to think ahead, increasing their awareness of the possible consequences of their decisions, when trying to maximize their profits. She comments as follows:

- "I have taught Junior Accountancy for many years but I haven't seen pupils grasp the concepts of profit and loss as quickly as they did using this program."

/When evaluating ...

When evaluating a piece of software, we should perhaps look beyond the immediate requirements of the syllabus in mathematics and consider possible cross-curricular benefits that might accrue through the use of this software.

6.3 Some Mathematical Computer Simulations

6.3.1 The ITMA Simulation *DICECOIN*

DICECOIN, designed and programmed by A. Wigley, simulates the throwing of dice and tossing of coins, and uses graphs and tables to illustrate the patterns underlying random events. Using this program the teacher and/or pupil(s) can perform any number of interesting experiments and analyze the results produced. The program can act as a powerful demonstration aid in classroom ('laboratory') work with the elementary concepts of probability and statistics.

6.3.2 The ITMA Simulation *EUREKA*

EUREKA, designed and programmed by R. Phillips, supports the teaching of elementary graph interpretation. Under the user's control, it illustrates what happens to the water level in a bath when the tap is turned on or off, together with the plug being put in or pulled out and a man getting in or out. This is all shown in pictorial form in the upper part of the screen, while in the lower part a graph is plotted of water level against time.

The program can be used to teach a number of aspects of graph interpreting and sketching. The idea of gradient can be discussed by considering the different gradients which result from filling the bath, from emptying it, or from emptying it with the tap left on. Step functions are illustrated by the man entering and leaving the bath.

It is interesting to note that *EUREKA* is suggested as the medium for an investigation to be carried out by 11 year-olds in the proposals by the National Curriculum Council (1988) for a new national curriculum in the United Kingdom - "Mathematics for ages 5 to 16".

6.3.2.1 An Evaluation of *EUREKA*

Most respondents classified this program as a simulation game. The teachers in the sample saw much potential use for the program as a means of introducing graphs in a concrete way. They agreed that the visual advantages of the computer were put to good use here and the pupils would learn the underlying mathematical concepts in a recreational fashion. The student teachers were equally enthusiastic about this program and both groups came to the conclusion that this was an instance where the computer's aid was very valuable to the teacher.

6.3.3 The NASOU Software Simulation *NUMBER MACHINE*

NUMBER MACHINE, designed by P. Human, programmed by C. Joubert and published by NASOU, furnishes a simulation of mathematical functions. It not only promotes computational skills, but also provides experience of sets, variables, formulae, mappings, number properties and functions. The exercises it offers extend in scope from basic addition to the determining of function rules of the form $x \rightarrow ax + b$ corresponding to given data sets. It supplies a dynamic visual display in the form of a "number machine" (a function diagram) which embodies a one-to-one correspondence between two sets of numbers. In each case the learner must determine four elements omitted from one of the two sets. In the words of its developers, "*NUMBER MACHINE* affords the opportunity to gain computational experience of important mathematical problem types, without prior knowledge of the appropriate mathematical terminology and notation styles (apart from the self-explanatory function diagrams)" (NASOU, undated).

6.4 Concluding Remarks

There are various characteristics of computer games and simulations, such as those discussed in this chapter, which are of educational value. These include provision for concept development, hypothesis construction and testing, strategic thinking, skill practising, experimentation, and problem solving. The computer offers several advantages over traditional means of providing these opportunities. Inter-alia, it can generate examples quickly and efficiently and has effective visual impact.

Some of the programs discussed in this chapter will be reviewed and reclassified in Chapter 9.

CHAPTER 7 COMPUTATIONAL ENVIRONMENTS FOR MATHEMATICAL PROBLEM SOLVING

CHAPTER 7

COMPUTATIONAL ENVIRONMENTS FOR MATHEMATICAL PROBLEM SOLVING

7.1 Introduction

In the introduction to this write-up several uses of the computer for the purposes of mathematics education were referred to. Amongst these were solving a problem using a supplied program tailored for the purpose, and solving a problem creatively with the aid of a supplied program. The difference between these two related activities lies in the degree of intellectual effort required of the user.

When using a custom-built program as a problem solving aid the intellectual challenge is minimal. The user simply feeds in the necessary parameters and the solution is presented to him. As an example, consider the use of a specially designed linear programming package in solving linear programming problems. All the user need do is input the constraint coefficients and a suitable representation of the cost function. The program does all the necessary processing automatically and produces the optimal solution.

If, on the other hand, the user has at his disposal not a specially tailored program but a supportive computational environment, the intellectual challenge can become considerably greater. An example of this is the solving of linear programming problems using an algebraic manipulation package. The package might offer built-in routines for, say, equation solving, or determinant processing, but the user must exercise mathematical perception and expertise in using these facilities to solve a given problem.

/It is ...

It is this latter situation, that of creative problem solving within a suitable computational environment, that will be explored in this chapter. The *PC PLANNER* spreadsheet system and the *muMATH-83* algebraic manipulation package will serve as examples of pre-programmed problem solving environments for discussion purposes. Such systems require no knowledge of programming on the part of either teacher or pupil. Attention will also be paid, however, to programming languages such as Logo, Prolog and muSIMP which lend themselves to the creation of problem solving environments, otherwise known as microworlds. In these cases both teacher and pupil have to do some programming.

7.2 The Electronic Spreadsheet

7.2.1 The "Daredevil Problem"

Concerning the use of electronic spreadsheets, let us take as our initial example a problem and its solution as described by Kelman (1983), modified and demonstrated by the author (Marsh, 1985a). Kelman used the VisiCalc spreadsheet, the author used SuperCalc, but, in retrospect, a better choice would have been the more sophisticated *PC PLANNER* with its handy graphics facilities. The problem is as follows:

> A motorcycle "daredevil" is designing a stunt in which he must drive a motorcycle up a ramp and jump over 10 motor cars (a distance of 60 meters), landing safely on the other side. The problem is that he has never done this stunt before. He doesn't know how fast to go or how steep to make the jump ramp. He knows from previous experience that the motorcycle can be driven off an incline up to 65 degrees at speeds no greater than 100 km/h. However, he wishes to do the jump with maximum safety, which means at as little incline as possible, at as low a speed as possible.

/What is ...

What is the best combination of take-off speed and ramp angle to ensure a safe jump?

There are two advantages in using a spreadsheet for solving problems of this nature. Firstly, the problems themselves can be realistic ones without any need for the usual artificial oversimplification. Secondly, any seemingly complex formulae, which might otherwise have the detrimental effect of putting pupils off, can be built into the spreadsheet by the teacher and kept hidden from the pupils until the right psychological/educational moment.

The formulae required for solving the "daredevil problem" are the equations of motion:

$$V_x = V_0\cos p;$$
$$h = (V_0\sin p)^2/2g;$$
$$t_f = 2\sqrt{[2h/g]} = 2V_0\sin p/g;$$
$$R = V_x t_f = V_0^2\sin 2p/g.$$

In these equations, V_x is the horizontal component of the speed of the motorcycle, V_0 is the take-off speed, h is the maximum height in meters reached by the motorcycle as it flies through its parabolic arc, t_f is the time of the flight in seconds from take-off to ground, R is the range in meters, that is, the distance measured horizontally through which the motorcycle travels, p is the ramp angle in degrees, and g is 9.8 m/s^2, the acceleration due to gravity.

With the above formulae built into the spreadsheet as relationships between columns of cells, pupils can investigate the problem in a dynamic way. They can observe the effects on the distance covered by the motorcycle of changes they make to the ramp angle and the take-off speed, eventually arriving at an optimal solution.

/Several ...

Several educational issues arise out of problem solving in a context such as that described above. Firstly, by examining samples of output the pupil may become aware of underlying patterns and relationships. In the example under discussion there is a distinct symmetry in the physics of the problem as shown by the following table:

Ramp angle in degrees	**Speed in km/h**	**Distance in meters**
25	100	60.31
30	94	60.25
35	91	61.27
40	88	60.05
45	88	60.97
50	88	60.05
55	91	61.27
60	94	60.25
65	100	60.31

Table 7-1: Spreadsheet showing Symmetry in the Physics of the "Daredevil Problem"

This table also illustrates the point that in real-world problem solving there are seldom unique solutions.

In addition, in the traditional sense the pupils are not solving the problem, the spreadsheet system is. However, the pupils are acting as problem solvers in a far more profound sense. By using the computer in this way, they can make scores of complex calculations quickly and accurately. Freed from the tedium and guaranteed of the accuracy of the computations, pupils can explore the various problem parameters, and in so doing discover patterns that may lead them to understand fundamental concepts.

The spreadsheet processing of the "daredevil problem" provides an ideal medium within which pupils can come to terms with the function concept. The problem involves two variables: the speed of the motorcycle and the angle of the ramp. As each set of numbers is entered, the spreadsheet values change in paired fashion. There is no need for pupils to grapple with some abstract definition of the function concept. They can think of it simply as a procedure that changes output in some direct, observable way as input varies. The spreadsheet format for the "daredevil problem" emphasizes the function relationship between variables and de-emphasizes the process of calculating the values.

Before dealing with other spreadsheet applications, it is perhaps worth mentioning that *TK!Solver*, described in detail by Miller (1984), is an automatic equation solver which deals very effectively with problems of the above type. Unlike the spreadsheet, it emphasizes the solution of equations written in their usual form, rather than the manipulation of a matrix of data cells. The user simply inputs his set of equations and instructs the program to solve for any required variable. The program sorts through the equations to determine which are appropriate, and in what order they must be used, to provide the variable value needed to solve the next equation in sequence, until the desired solution is reached. The program also converts the dimensions of a variable to correspond to those of other variables in the problem.

7.2.2 Other Spreadsheet Applications

D.E. Arganbright (1984) describes several mathematical spreadsheet applications. These include the solving of 'word problems' in algebra, trigonometry and calculus, the generation of the Fibonacci sequence and of factorial numbers, and the location of roots of polynomials.

D.F. Burrell (1986), under the author's supervision, found several other suitable senior school level spreadsheet applications and some of these will be briefly reported here: in particular the computation of compound interest; the computation of optimal selling prices; the plotting

/of straight line ...

of straight line, parabolic and hyperbolic graphs; trigonometric computations; and computations concerning series convergence.

7.2.2.1 Compound Interest

Taking compound interest first, consider the case where someone invests an amount of money, say R100, at a particular interest rate, say 6%, for a certain period of time, say 15 years. In order to establish how the value of the investment increases over the period, we set up a table such as that shown below:

	Capital Amount:	100.00		
	Interest Rate:	6.00	percent	
Year	**Capital**	**Interest**	**Cuml. Int.**	**Total**
0	100.00	0.00	0.00	100.00
1	100.00	6.00	6.00	106.00
2	100.00	6.36	12.36	112.36
3	100.00	6.74	19.10	119.10
4	100.00	7.15	26.25	126.25
5	100.00	7.57	33.82	133.82
6	100.00	8.03	41.85	141.85
7	100.00	8.51	50.36	150.36
8	100.00	9.02	59.38	159.38
9	100.00	9.56	68.95	168.95
10	100.00	10.14	79.08	179.08
11	100.00	10.75	89.83	189.83
12	100.00	11.39	101.22	201.22
13	100.00	12.07	113.29	213.29
14	100.00	12.80	126.09	226.09
15	100.00	13.57	139.66	239.66

Table 7-2: Spreadsheet for Compound Interest Problem

Note that in constructing this table the user enters only the capital amount, the interest rate, the column of numbers 0-15 to indicate the years, and the formula by which the amount at the end of the first year is computed. The amounts at the ends of successive years are automatically computed by the spreadsheet program by a process of replication.

Having computed all the necessary values it is a simple matter to generate a graph such as that shown below.

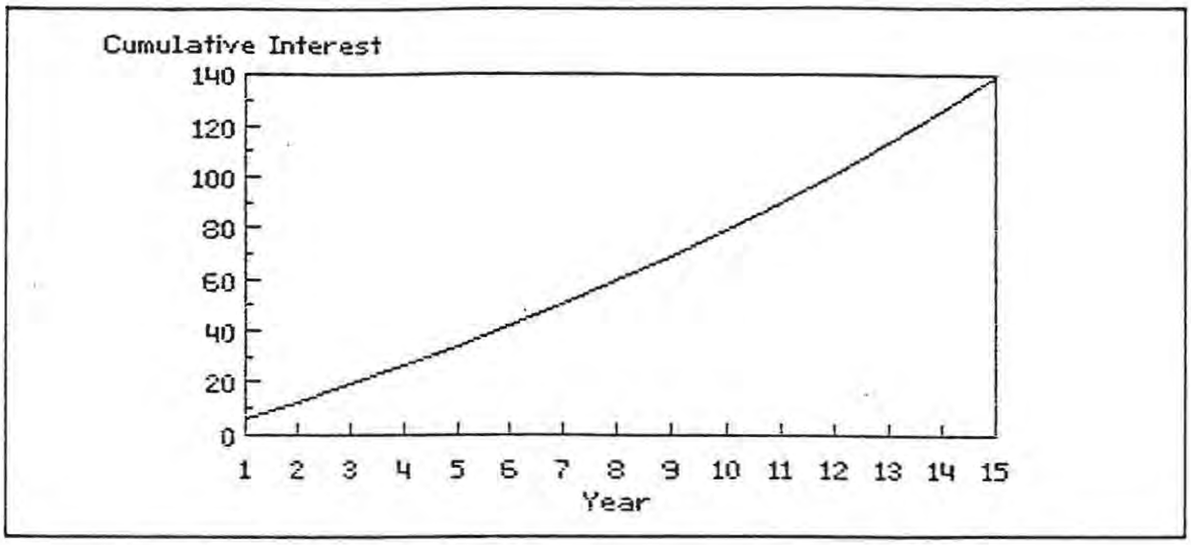


Figure 7-1: **Spreadsheet Graph for a Compound Interest Problem**

Spreadsheets are ideal tools for experimenting with data and answering the 'what if ...?' question. The user need only change one value and the values of all the other cells are recomputed automatically. In the above example, the effect of, say, a change in interest rate can be determined with the greatest of ease, the new table and new graph being produced within seconds.

Some 'real-life' compound interest problems which could be fruitfully investigated using a spreadsheet are described, but solved by traditional methods, by Noble (1990).

/7.2.2.2 Optimization ...

7.2.2.2 Optimization Problems

Optimization problems can usually be solved quite easily with the aid of a spreadsheet program. The "daredevil problem" gave us one rather sophisticated (or specialized) example. A more general and somewhat simpler type is represented by the following example:

> An entrepreneur produces and sells some particular kind of goods, say, 'whatsits'. It costs him 5c to produce one 'whatsit' and he produces 50 of them each day. A market survey has revealed the following trend in 'whatsit' consumption:

Selling Price (in cents)	**Units Sold**
7	47
8	39
9	37
10	35
11	29
12	25
13	22
14	19
15	16
16	14

Find the optimal selling price.

Table 7-3: Data for Optimization Problem: Selling Price vs Units Sold

The optimal selling price is obviously the price at which the greatest profit is made. This can be found very easily by setting up a spreadsheet such as that shown below:

/Table 7-4: ...

	Cost per unit:	5 cents			
Units produced	Production cost	Selling price	Units sold	Total sales	Net profit
50	250	7	47	329	79
		8	39	312	62
		9	37	333	83
		10	35	350	100
		11	29	319	69
		12	25	300	50
		13	22	286	36
		14	19	266	16
		15	16	240	-10
		16	14	224	-26

Table 7-4: Spreadsheet for Optimization Problem: Finding Optimal Selling Price

As before, variables can be changed and the resulting effects observed almost instantly.

We can also use *PC PLANNER* to show our results graphically, and herein lies a sizeable part of the value in using this system for this kind of problem solving. The following bar chart, which is readily produced by *PC PLANNER* shows production cost (constant in this case), sales and net profit for each different selling price in the example above.

/Figure 7-2: ...

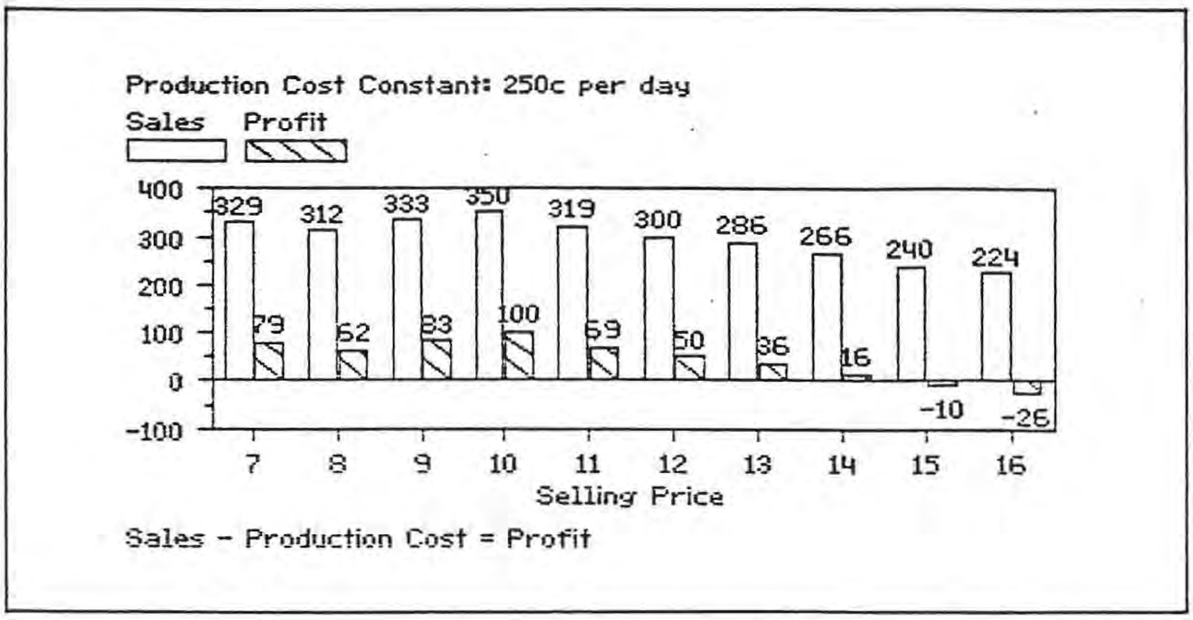


Figure 7-2: **Spreadsheet Bar Chart showing Production Cost, Sales and Net Profit for an Optimization Problem**

The spreadsheet thus provides us with a dynamic and visually powerful problem solving environment.

7.2.2.3 Graph Plotting Using a Spreadsheet

While considering the graphical capabilities of *PC PLANNER* let us look briefly at the plotting of straight lines, parabolas and hyperbolas using this system. In the case of the straight line a spreadsheet such as that shown below can be used for exploring the effects of changes in the values of m and c, the gradient and y-intercept respectively of the function y = mx+c.

/**Table 7-5: ...**

y = mx+c	m =	2
	c =	3
x	y	
-4	-5	
-3	-3	
-2	-1	
-1	1	
0	3	
1	5	
2	7	
3	9	
4	11	
5	13	

Table 7-5: Spreadsheet for Exploring the Straight Line Function

The graphs produced have the following form:

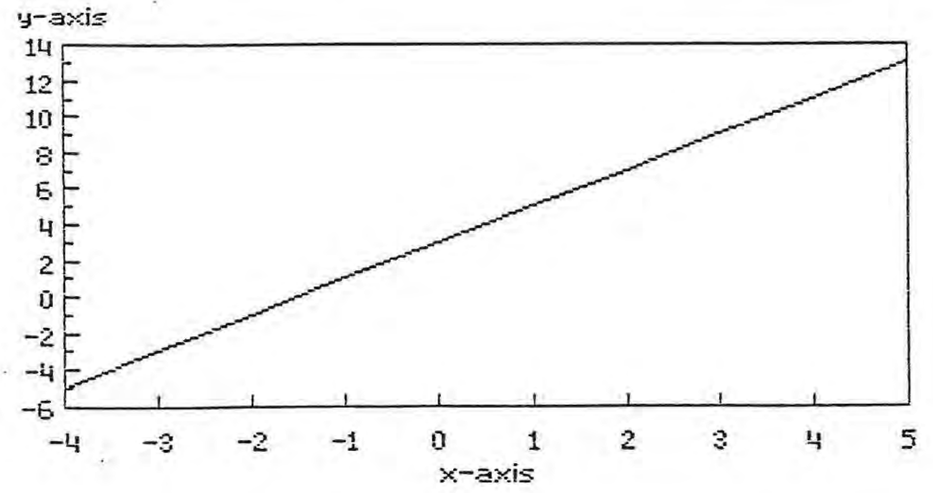


Figure 7-3: Spreadsheet Graph of the Straight Line Function
y = 2x + 3

/The parabolic ...

The parabolic function can be explored in a similar way. Changes in the values of a, b and c in $y = ax^2 + bx + c$ affect the graph in clearly perceptible ways. A sample spreadsheet with graph is given below.

The spreadsheet:

y = ax*x+bx+c	a =	2
	b =	-3
	c =	5
x	y	
-10	235	
-9	194	
-8	157	
-7	124	
-6	95	
-5	70	
-4	49	
-3	32	
-2	19	
-1	10	
0	5	
1	4	
2	7	
3	14	
4	25	
5	40	
6	59	
7	82	
8	109	
9	140	
10	175	

Table 7-6: Spreadsheet for Exploring the Parabolic Function

/The graph: ...

The graph:

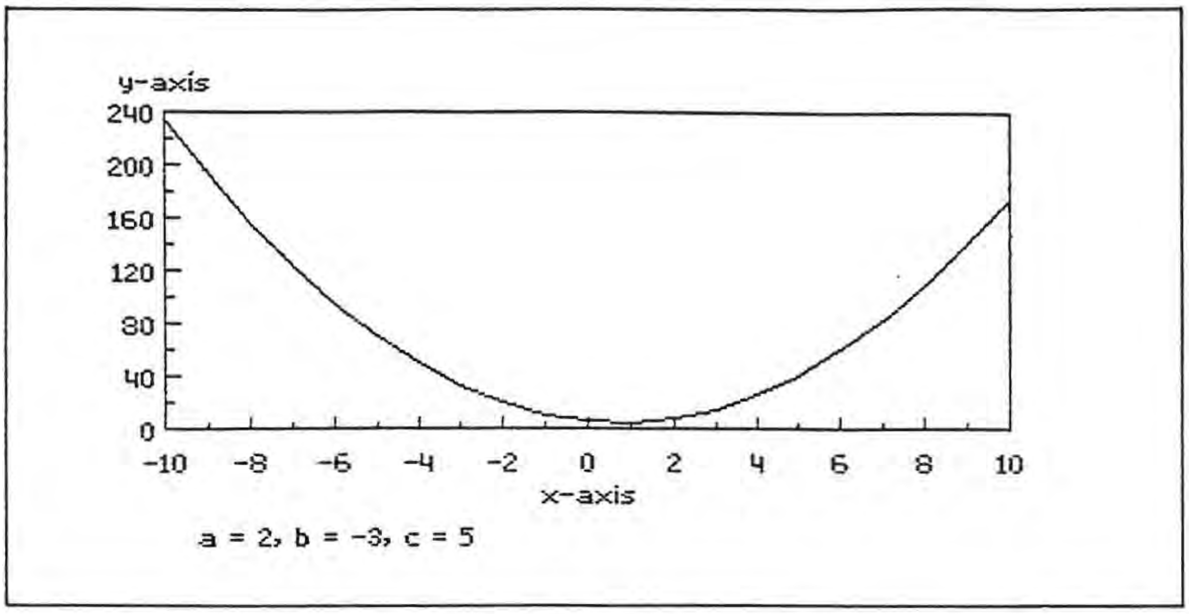


Figure 7-4: Spreadsheet Graph of the Parabolic Function $y = 2x^2 - 3x + 5$

It should be noted that hyperbolic functions also lend themselves readily to this type of investigation.

7.2.2.4 The Method of Successive Approximation

The spreadsheet provides us with a handy environment within which to solve quadratic (or other) equations by the method of successive approximation. Take, for example,

$$x^2 - 5x + 3 = 0.$$

We start by establishing a convenient domain, say, (-5,5), and a suitable step size, say, 1. It is a simple matter to use the spreadsheet to display a table of domain values with related function values as shown below.

/Table 7-7: ...

Domain value	Function value
-5	53
-4	39
-3	27
-2	17
-1	9
0	3
1	-1
2	-3
3	-3
4	-1
5	3

Table 7-7: Spreadsheet for Solving a Quadratic by Successive Approximation: Initial Estimate

Analyzing this table it becomes evident that there must be zeros in the intervals (0,1) and (4,5). A beneficial side-effect is that the table also provides indications as to where the minimum of the quadratic lies and what the solutions are to the inequalities:

$$x^2 - 5x + 3 < 0,$$
$$x^2 - 5x + 3 > 0.$$

To get more accurate estimates of the zeros we simply tabulate the domains (0,1) and (4,5) with a refined step size, say, 0.2, and instruct the system to compute the corresponding function values. A sample spreadsheet follows:

/Table 7-8: ...

Domain value	Function value
0.0	3.00
0.2	2.04
0.4	1.16
0.6	0.36
0.8	-0.36
1.0	-1.00
4.0	-1.00
4.2	-0.36
4.4	0.36
4.6	1.16
4.8	2.04
5.0	3.00

Table 7-8: Spreadsheet for Solving a Quadratic by Successive Approximation: Refined Estimate

It is clear that 0.7 is a reasonable approximation to the zero which lies in the interval (0,1). Similarly, we find that 4.3 is a reasonable approximation to the other zero. Obviously, further refinement will help to locate these zeros more precisely if required.

From the educational point of view, numerical methods such as that described above, whilst not mathematically elegant, have several virtues. In the words of Fey et al. (1984):

> **"First, the 'guess and test' strategy seems incredibly easy to remember - it is so natural. Long after students have forgotten the quadratic formula or developed bugs in their factoring skills, they will likely know enough to try successive approximation. Second, the method of successive approximations applies without a major change to the solution of nearly every type of equation, from higher-degree polynomials to rational, exponential, or trigonometric conditions. Third, the numerical approach stresses the functional point of view that is at the heart of most later applications of algebra - in calculus, for instance."**

/7.2.2.5 Trigonometric ...

7.2.2.5 Trigonometric Spreadsheet Applications

Problem solving involving the use of mathematical formulae, such as is often the case with trigonometric applications, can be profitably done with the aid of a spreadsheet. An example which entails the use of the triangle area formula

$$\frac{ab\sin C}{2}$$

is given below:

Calculate the area of a triangle, given the lengths of two of the sides and the size of their included angle.

Side a	Side b	C(deg)	C(rad)	sin C	Area
7.8	5.4	122.0	2.129	0.848	17.860
40.0	50.0	30.0	0.523	0.500	499.770
80.0	90.0	150.0	2.617	0.501	1804.136
10.0	8.0	45.0	0.785	0.707	28.273
10.0	16.0	60.0	1.047	0.866	69.261
50.0	72.0	73.0	1.273	0.956	1721.008
136.0	163.0	122.0	2.128	0.849	9406.100
76.0	95.0	105.0	1.832	0.966	3487.859

Table 7-9: Spreadsheet for Calculating Areas of Triangles using Trig. Formula

By using the computer in this way, the pupil is freed from the sometimes obfuscating mechanics of the computation process, and given the opportunity to explore the relationships between (in this case) sides, angles and area of a triangle.

/7.2.2.6 Series ...

7.2.2.6 Series Investigation Using a Spreadsheet

One more fruitful application of the spreadsheet is in the investigation of series convergence. Take the geometric series

$$1 + 1/2 + 1/4 + 1/8 + \ldots$$

for example. The convergence of the sequence of partial sums of this series is clearly demonstrated in the following spreadsheet:

a =	1
r =	0.5

n	S_n
1	1.00000000000000
2	1.50000000000000
3	1.75000000000000
4	1.87500000000000
5	1.93750000000000
6	1.96875000000000
7	1.98437500000000
8	1.99218750000000
9	1.99609375000000
10	1.99804687500000
11	1.99902343750000
12	1.99951171875000
13	1.99975585937500
14	1.99987792968750
15	1.99993896484370
16	1.99996948242180
17	1.99998474121090
18	1.99999237060540
19	1.99999618530270
20	1.99999809265130
21	1.99999904632570
22	1.99999952316280
23	1.99999976158140
24	1.99999988079070
25	1.99999994039530
26	1.99999997019770
27	1.99999998509880
28	1.99999999254940

29	1.99999999627470
30	1.99999999813730
31	1.99999999906870
32	1.99999999953430
33	1.99999999976710
34	1.99999999988360
35	1.99999999994180

Table 7-10: Spreadsheet for Investigating Series Convergence

This is a handy non-programming alternative to the kind of programming solution to problems like this which we shall consider in Chapter 8.

It should be noted that I.E. Phillips (1989) has produced a lesson plan along with a set of worksheets, implementing a guided discovery approach, using the spreadsheet, to the establishment of the conditions under which a geometric series converges.

The spreadsheet can also be used to illustrate special sequences such as

$$\lim_{n \to \infty} (1 + 1/n)^n$$

or

$$\lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

and to compare, for example, f and f′ to show up regions of increase and decrease, illuminating the concept of derivative.

/7.3 A Computer Algebra ...

7.3 A Computer Algebra System

The electronic spreadsheet is one example of a rich computational environment for mathematical problem solving. Let us look now at another such environment, but of a rather different type: the *muMATH-83* interactive computer algebra package. The astonishing capabilities of this system have been pointed out by its developers, D. Stoutemyer and A. Rich (the *muMATH*/muSIMP-83 Reference Manual, 1983). Strong backing (for the 1980 release) has also been provided by Steen (1981), Wilf (1982), Pavelle, Rothstein and Fitch (1981) and (for the 1980 and 1983 releases) by the author (Marsh, 1985b). *muMATH* is virtually the realization of C. Engelman's (1968) vision of the way the computer could bring about the mechanization of mathematics, providing the user with:

> **"a mechanical assistant possessing an encyclopedic knowledge of mathematical formulae and results, capable of rapid and flawless computation, familiar with a broad spectrum of computational algorithms."**

In *muMATH* we have a powerful algebraic manipulation system running on at least two popular microcomputers. It is a system which transforms algebraic expressions according to the rules of algebra, without making any use of numerical methods. As demonstrated previously (Marsh, 1985b), the *muMATH* system handles conventional school algebra and first year university calculus with ease.

7.3.1 Examples of *muMATH* Performance

Perhaps a few simple examples of *muMATH* performance are in order at this point. These will be presented in tabular form without regard for the system's syntax:

/Table 7-11: ...

Problem	***muMATH* solution**
Expand and simplify: $3x(x^2-y)+3y(x+y)$	$3x^3+3y^2$
Solve: $\frac{x+5}{10x-7} = \frac{2x+3}{5x-7}$	$x = -\frac{14}{19}$
Simplify: $\tan x \cos x + \frac{1}{\operatorname{cosec} x}$	$2\sin x$
Find $\frac{dy}{dx}$: $y = \tan x - \cot x + \arctan x$	$\sec^2 x + \operatorname{cosec}^2 x + \frac{1}{1+x^2}$
Compute the limit: $\lim_{h\to 0} \frac{\sqrt{[4-h]}-2}{h}$	$-\frac{1}{4}$

Table 7-11: Examples of *muMATH* Performance

7.3.2 Computer Algebra in the Classroom

Does such a system, which simply produces solutions without giving any account of its reasoning processes, have a role to play in ordinary classroom mathematics? The author believes it has, and a vital role at that. Rather than replacing conventional problem solving activities it could augment them in a powerful way. Pupils could use the *muMATH* environment as a 'laboratory' resource, experimenting with mathematical concepts, verifying conjectures, investigating *real* problems, and making their own discoveries. They could concentrate on fundamental ideas instead of spending a disproportionate amount of time on repetitive mechanical transformations.

Fey and Heid (1984) have pointed out that:

> **"If an algorithm or symbol manipulation is taught to secondary school or calculus students, in all likelihood there is a microcomputer program that will perform the procedure."**

Continuing this line of reasoning they remark that we shall soon be in the situation where:

> **"Easy-to-use, low-cost programs will perform most of the skills that today's students spend hours practising."**

Clearly there are pressing implications for curriculum renovation. Some of these will be considered in Chapter 10.

7.3.2.1 A Computer Algebra Workshop run for Standard 9 Pupils

7.3.2.1.1 Description of Workshop

St. Andrew's College, Grahamstown, holds an annual "Learning Conference" for the benefit of its standard 9 pupils and those of its sister school, The Diocesan School for Girls (DSG). The 1989 conference was held in April and as part of the proceedings the author ran a computer algebra workshop for Higher Grade pupils. There were thirty-two participants working in eight groups of four, each group using one machine. Following a brief introduction the pupils spent an hour working as teams to solve the problems posed on a worksheet.

These problems were designed so as to utilize the power of *muMATH* for freeing pupils from manipulative tasks and supporting them in the investigation of concepts. A copy of the worksheet can be found in Appendix A7.

7.3.2.1.2 Reaction of Pupils

The workshop was evaluated by means of a questionnaire handed to the pupils afterwards. A copy of this questionnaire will also be found in Appendix A7. Twenty of the participants responded and submitted their completed questionnaires two days later. Overall the feedback was very positive. Only two respondents said that they had not enjoyed the session. One of these gave her reasons as having difficulty with working as a member of a group and also finding the *muMATH* notation rather awkward: ^(1/2) for $\sqrt{}$, etc. The other gave no reasons.

Here are some sample responses to the question, "What did you like most about the session?":

- "[It provided] a new outlook on maths."
- "Trying to predict the answers to the questions."
- "It was a different approach to maths. which gave me a new insight."
- "It was different - more exciting ..."
- "... when you don't understand a problem there is an easy method whereby you can solve it."
- "... It was a quick and efficient way of doing maths."
- "The ease [with] which the computer is able to deal with problems [such as those given]. If one [could think of] the appropriate [method] [the computation] was elementary."
- "It was an enjoyable experience compared to normal classroom maths."

/Many ...

Many of the pupils did not answer the question, "What did you like least about the session?", but here are some responses:

- "[Working on the computer] takes too long. If you work in groups you can't really understand what [is being done] ..."
- "I do not really like working with computers."
- "There were too many people in a group ... not every-one could see the screen."
- "The computer language is a little confusing and takes some getting used to ..."
- "The typing ... was tedious."
- "*muMATH* [tells] you the answer without [showing] where the answer is derived from."
- "We did not have to do much thinking for ourselves."

7.4 A Small-Scale Problem Solving Environment for Transformation Geometry

PC PLANNER and *muMATH* are two relatively large-scale and general purpose computational environments. There are, of course, many much smaller, single purpose environments designed to support the exploration of mathematical concepts. One of these is the WAMSOFT package *GEOMETRY I* for exploring transformation geometry using the BBC-B microcomputer. This package is by no means in the same category as *PC PLANNER* and *muMATH*, but, nonetheless, it is a useful small-scale computational environment in its own right.

WAMSOFT's *GEOMETRY I* can be used to establish the effects of single and combined transformations in the plane for simple shapes of the user's choosing. The package consists of two programs: one for rigid transformations and the other for linear transformations.

/The first ...

The first of the above options is designed for use as an aid in an introductory course on transformation geometry. The program demonstrates the isometries of translation, reflection and rotation for various shapes of the user's choice. A key feature is that the motion of the transformation is clearly shown.

The second program demonstrates the transformations of translation, reflection, rotation, shear and dilation for shapes of the user's choice. The user has the option of transforming either the original shape or its image, thus enabling the demonstration of composition of any number and any combination of the above transformations. Once again the motion of the transformation is clearly shown.

The functioning of this software facility is best explained by means of an illustration. In the diagram below (as appears on the computer screen), the flag shown as (1) is transformed into the image shown as (2) by a shear with respect to the shear line y = -1 and with shear ratio 1,5. This image is then transformed into the final image shown as (3) by a rotation through -270° about the point (1;-1).

/**Figure 7-5: ...**

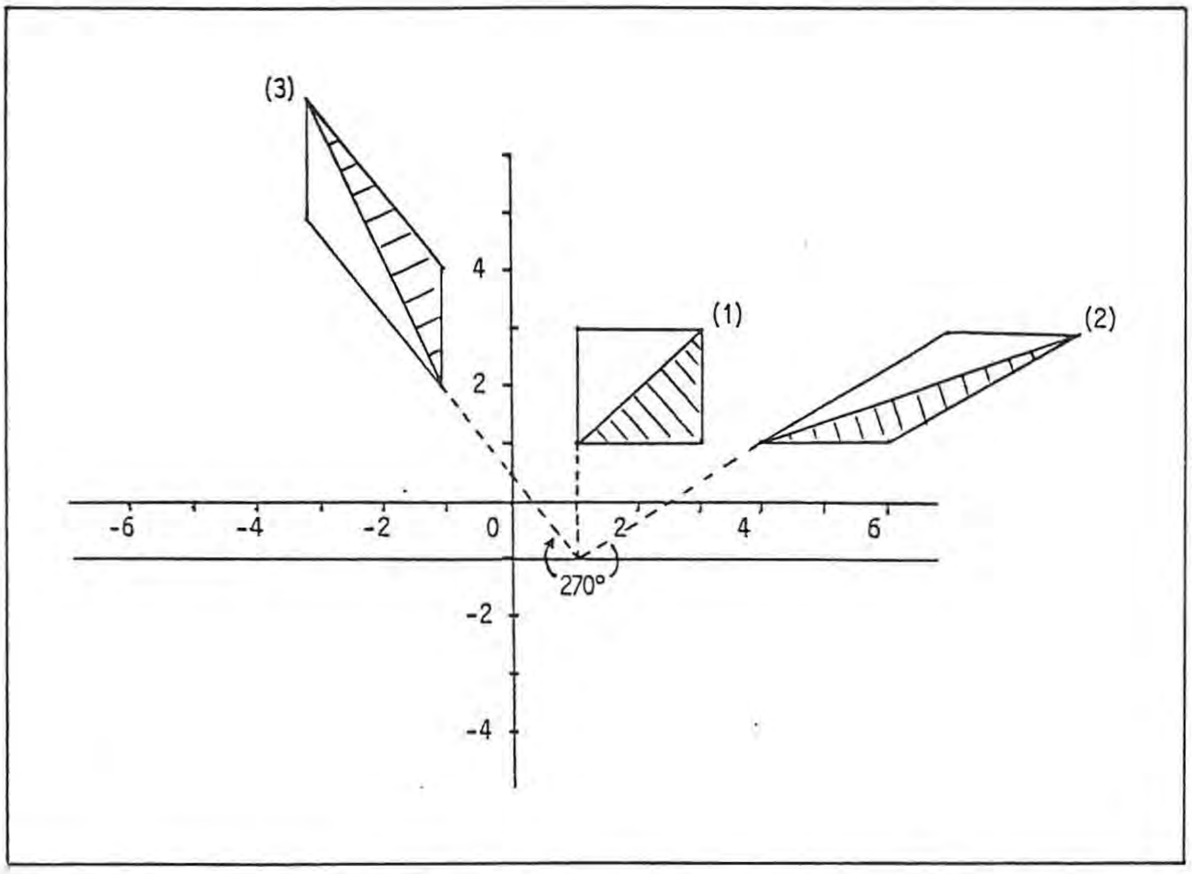


Figure 7-5: WAMSOFT Transformation Geometry Screen Display

Transformation geometry such as that demonstrated above could form a logical and dynamic extension to our conservative and somewhat static school geometry syllabus. Moreover, software such as the WAMSOFT *GEOMETRY I* programs could provide a useful context within which to implement this extension.

7.5 A Variety of Other Problem Solving Environments

7.5.1 The *GEOMETRIC SUPPOSER*

The *GEOMETRIC SUPPOSER* is a suite of three computer programs -*TRIANGLES*, *QUADRILATERALS* and *CIRCLES* - available on Apple and PC machines, which enables pupils to carry out with ease constructions that are possible using straightedge and compass. These constructions include, for example, the construction of triangles, quadrilaterals and circles and the drawing of segments, medians, altitudes, extensions, parallels, perpendiculars, angle bisectors and tangents. The program allows pupils, in addition, to carry out a variety of measurements and make computations involving lengths, angles and areas, including arithmetic combinations of these measures. Pupils are encouraged to make conjectures in geometry by using the programs to explore the relationships that might or might not exist among geometrical objects.

Although the *GEOMETRIC SUPPOSER* was designed originally for use in high school geometry classes, pupils at the middle school level in the USA are using it to explore ideas that are not part of the standard curriculum. High school pupils are using the programs to approach the traditional content of geometry courses in novel ways. The programs are also in use in both pre- and in-service teacher training. They are even being used in drafting and design courses.

7.5.2 Tall's Graphing and Calculus Programs

D. Tall (1985a) has written a comprehensive collection of programs for graphing and calculus applications. This collection, implemented initially on the BBC microcomputer but now also available on PC compatibles, consists of two packages: *SUPERGRAPH* and *GRAPHIC CALCULUS*.

SUPERGRAPH is a graph-drawing package which allows the user to draw graphs using Cartesian co-ordinates, or more advanced graphs using parametric and polar co-ordinates and other facilities. Functions may be entered in ordinary mathematical notation. The package includes a facility for producing perspective drawings of surfaces in three dimensions, and another which allows for the drawing of tangents and normals, with a 'zoom' facility to re-draw any part of the display to a new scale. It can be used by a range of people from pupils first learning about graphs to university students working at an advanced level.

GRAPHIC CALCULUS is divided into three parts, each with three programs and a book of ideas and suggestions for use.

> *GRAPHIC CALCULUS I* introduces the gradient of a graph and the theory of differentiation.
>
> *GRAPHIC CALCULUS II* introduces integration and shows how it complements differentiation.
>
> *GRAPHIC CALCULUS III* is an introduction to the theory of differential equations that draws solutions of first and second order and simultaneous differential equations. It also shows how these equations can be solved by numerical and formal methods.

All three parts of this package may be used with a minimum of instruction and are designed as teaching tutorials which provide a greater insight into the mathematical processes of the calculus. C. Smith (1990) reports positive findings in his tutorials at the Technikon Natal, using the IBM version of *GRAPHIC CALCULUS* to teach engineering students at the T2 level.

/7.5.3 Bell and Hyman's ...

7.5.3 Bell and Hyman's "Advanced Mathematical Software for Exploring Functions and Graphs"

These programs were designed for BBC computers by M. and P. Perkins (1987). They were originally developed as investigative tools for A-level mathematics pupils, but can be used by pupils of other mathematics courses. The programs are accompanied by a book whose purpose is to provide a series of forty work units which show the teacher how the programs can be used in the sixth form mathematics class. The units are designed to encourage class discussion and to suggest ideas for further investigation by individuals or small groups. The programs are used as 'black box' tools.

An example of a part of a work unit is described below:

Graphs of trigonometric functions

Use the first part of *PROGRAM C1* (composite functions) throughout this unit, leaving the built-in values for x and y unchanged.

1. Run the program for the function sinx. Now answer the following questions about the function sin2x:
   - (a) What is the range of the function sin2x? Does it have the same range as the function 2sinx?
   - (b) Bearing in mind that the value of 2x increases twice as fast as the value of x, describe how the graph of sin2x is related to the graph of sinx?

/Add ...

Add the graph of sin2x to the display by selecting the f(ax) option and entering a=2. Compare the graphs of sin2x and 2sinx by selecting the af(x) option with a=2.

2. ..., etc.

7.5.4 Sicks's Programs for Investigating Secondary Mathematics

This package, designed for the Apple II computer by J.L Sicks (1985), enables computer-enhanced investigation of many topics from the secondary mathematics curriculum, as well as several non-traditional areas of mathematics. Although the focus is on mathematics, not programming, there is interplay between the two since the computer is used to lead to discovery in mathematics, and mathematics is used to explain some of the problems such as round-off error that are encountered when using the computer for calculation.

Each chapter of the accompanying book begins with a simple BASIC program which is read off the disk. The program is used for the mathematical task being investigated and the pupils analyze it to see how it works and to get the result. The pupils are expected to modify the program to make it faster or more accurate, or to perform additional tasks. To do this the pupils need to know rudimentary BASIC.

7.5.5 Wain and Flower's Worksheets for "Mathematics Homework on a Micro"

G.T. Wain and S.M. Flower of Leeds University (1989), on behalf of the British Mathematical Association, have published a collection of worksheets in book form under the title, "Mathematics Homework on a Micro". This book is an outcome of a research and development project set up to explore the extent to which home-owned computers could be used as aids to the teaching of mathematics. The book comprises a set of homework tasks in the form of worksheets with, in each case, a short computer program written in BASIC playing the leading role in the activity. The pupil types in the program and then uses it interactively to carry out the mathematical tasks of the worksheet.

The worksheets are grouped in broad subject areas and are graded into labelled levels of difficulty. They are aimed at pupils aged 11-16 years and the tasks involve drill-and-practice, tutorials, consolidation and revision, and investigatory work. Although the worksheets are primarily intended for use at home, they could be easily and effectively used in the classroom.

An example of a worksheet on powers follows:

This program calculates the powers of any number, giving up to 70 figures accurately.

```
10  MODE3:DIM N(80)
20  INPUT "NUMBER",P
30  N(1)=1:L=1:Q=0
40  REPEAT:C=0:K=1:Q=Q+1
50  REPEAT:T=P*N(K)+C:C=T DIV 10
60  N(K)=T MOD 10:K=K+1:UNTIL K=L+1
70  IF C=0 THEN 100
80  REPEAT:N(K)=C MOD 10:C=C DIV 10
90  K=K+1:UNTIL C=0:L=K-1
100  PRINT;P"^"Q;TAB(79-L);
110  FOR K=L TO 1 STEP -1
120  PRINT;N(K);:NEXT:PRINT
130  UNTIL L>70
```

Test the program by keying in 10 when the word NUMBER? appears on the screen. You should then see a list of the powers of 10, like this:

/10^0 ...

```
10^0          1
10^1         10
10^2        100
  .           .
  .           .
  .           .
```

Press the <ESCAPE> key to stop the list.

TASKS

(1) Can you find any numbers whose powers always end in the same figure?

(2) Can you find any patterns in the first digit of each power?

Running the program with such input as 5, 9, 11, etc., produces interesting results. The authors' comment about this worksheet is that it is set at a difficult level, and that its activities involve sequences of powers being produced by the program which lead on to an exploration of number patterns.

The BASIC program for this and most of the other worksheets makes use of many of the facilities of the programming language and it would not be easy for the teacher or the pupils to modify the program unless they were reasonably proficient in BASIC. The programs are therefore best suited for use primarily as 'black box' type teaching tools.

7.5.6 Some More Small 'Tool Programs' for Mathematical Problem Solving

Several small programs falling into the mathematical tool genre and intended for use in the secondary school classroom and laboratory have been developed by Burrell (1986), in a project supervised by the author and based on work done by the author (Marsh, 1985a, 1986).

/The programs ...

The programs in this collection can be used as tools for the investigation of a variety of mathematical concepts and problem solving situations. They have all been written in either Logo, BASIC or Pascal. Most of them are short and fairly simple, but flexible and powerful. They lend themselves to adaptation and modification by the user, and it is contended that the algorithms on which they are based open up new ways to help pupils to understand the underlying aspects of mathematics.

The applications covered are: finding numerical factors; generating numbers with special properties, for example, primes; summing series; performing set and logic operations; solving triangles; drawing trigonometric graphs; animating geometrical concepts; and providing novel problem solving simulations, for example, for investigating the "Towers of Hanoi" problem.

Some of these programs are described in Chapter 8 as examples of programming exercises which could be undertaken by teachers and pupils.

7.6 Concluding Remarks

There is a variety of pre-programmed computational environments which offer facilities for investigative problem solving. They allow complex calculations and transformations to be carried out quickly, accurately and efficiently. They provide opportunities for experimenting with data, exploring relationships, verifying conjectures, creating visual images, and gaining greater insight into mathematical processes.

CHAPTER 8 COMPUTER PROGRAMMING AS A MATHEMATICAL ACTIVITY

CHAPTER 8

COMPUTER PROGRAMMING AS A MATHEMATICAL ACTIVITY

8.1 Introduction

Perhaps the most advanced, but in many ways the most rewarding, mode of computer usage in mathematics education, is that of writing a program to solve a problem. This can, of course, be done at many different levels: from writing a simple graphics program for drawing a square (Primary School children using Logo) to writing a program which performs symbolic differentiation on any given differentiable function (3rd year University Computer Science students using LISP). The essence of such problem solving is algorithm design and this surely is at the heart of much, if not most, mathematical problem solving. Computer programming is, in fact, an intensely mathematical activity.

D. F. Burrell (1986), in a study supervised by the author, identified a variety of suitable areas for computer application within and related to the secondary school mathematics syllabus in the RSA. As part of this study, over the course of several months, Burrell wrote and tested a substantial number of programs which could be used to enhance mathematical education in these areas. The languages used were Logo, BASIC and Pascal. The programs were all fairly short and reasonably simple. The intention was that they could be used in the mathematics classroom, either by the teacher for demonstration purposes or as exercises for the pupils themselves. Of course, the latter presupposes some knowledge of programming on the part of the teacher and pupils and this is the issue to which attention will be paid in this chapter. Some of Burrell's programs will serve as resource material to illustrate some of the points to be made.

8.2 Programming in the Mathematics Curriculum

Before considering actual programming examples (to be done in section 8.3), a plea for the inclusion of some programming in the mathematics curriculum should perhaps be motivated.

Notwithstanding the increasing production of suitable software and the resultant emphasis on facility with using programs rather than writing them, the author believes that, as programming is a strongly mathematical activity, there is a place for a measure of it in the school curriculum.

8.2.1 Recommendations Concerning Programming in Mathematics Education

Two strong recommendations concerning programming in mathematics education have been put forward: one in France (C. Bana et al., 1981) and the other in the USA (K. Jones and C. Lamb, 1985).

8.2.1.1 A French Experiment

In an experiment conducted on fourth form pupils (13 year olds) in French schools, Bana et al. (1981) used the methods of computer science in the teaching of mathematics. They found that:

> **"[emphasizing algorithm construction seemed to be] a major help in the acquisition, illustration and mastery of many of the basic principles of mathematics."**

/8.2.1.2 An NCTM ...

8.2.1.2 An NCTM Agenda for Action

Jones and Lamb (1985) responded to a publication by the National Council of Teachers of Mathematics (NCTM) which established the suggested modifications of the curriculum for the 1980's in the USA. The publication is, "An Agenda for Action: Recommendations on School Mathematics for the 1980's" (NCTM, 1980). Its foremost charge is that:

> **"Problem solving must be the focus of school mathematics in the 1980's."**

To meet this requirement Jones and Lamb (1985) argue the case for computer programming as a means of developing problem solving skills. In so doing they echo the view of J.S. Brown (1979) that:

> **"...the task of constructing and debugging programs [provides] an ideal domain for students to develop their problem-solving skills."**

Jones and Lamb (1985) believe that:

> **"incorporation of programming strategies in mathematics instruction is an essential component in the development of reasoning skills."**

They argue that programming involves the use of deductive reasoning processes. The programmer learns general rules and discovers specific applications for those rules. There is an obvious analogy between this and the way problem solving in geometry is done. It should be noted, however, that programming also entails a great deal of inductive reasoning, the solutions to specific problems leading to generalizations of a global nature.

/In a follow-up ...

In a follow-up to "An Agenda for Action", the NCTM issued a statement in September 1987 advocating that changes be made to mathematics curricula, instructional methods and teacher education to reflect the impact of computer technology. This statement was published in the NCTM News Bulletin (November, 1987). It stressed that:

> **"Teachers should use computers as tools to assist students with the exploration and discovery of concepts, with the transition from concrete experiences to abstract mathematical ideas, with the practice of skills and with the process of problem solving."**

Furthermore, it was felt that teachers should be able to identify topics which can be enhanced by application of the computer and that they ought to be capable of writing the necessary programs.

8.2.2 A Report by the UK Schools Inspectorate

More support for the inclusion of programming in the mathematics curriculum is to be found in a report by the UK Schools Inspectorate (DES, 1985). In this report it is acknowledged that if the computer is to be used as "a powerful means of doing mathematics", then "if programming is not taught elsewhere, it should be included in mathematics lessons" (p. 35).

8.2.3 The Work of Ross and Howe

P. Ross and J. Howe (1981) have written a definitive paper on the teaching of mathematics through programming. They draw a strong parallel between program development and mathematical thinking, and maintain that:

> **"[in programming, the] cyclical activity of testing and development provides an experience on which to build the mental concept of the**

internal testing of a hypothesis, which is a cornerstone of mathematical ability."

8.2.4 Feurzeig's Claims

Perhaps the first computer educationists to press the case for teaching children programming in mathematics classes were Feurzeig and his research team (Feurzeig et al., 1969). They made four claims:

(1) that programming provides some justification for, and illustration of, formal mathematical rigour;

(2) that programming encourages children to study mathematics through exploratory activity;

(3) that programming gives insight into certain mathematical concepts;

(4) that programming provides a context for problem solving, and a language with which a pupil may describe his own problem solving.

Initially some of the work with a bearing on these claims was done using BASIC, for example, Dwyer's so-called Project SOLO at the University of Pittsburgh (Dwyer, 1975). The vast bulk of work in this area, however, has been based on Logo. A recent, related development has been the investigation into the suitability of Prolog as a computer language for children. Among the leading researchers in this field are B. Kowalski and R. Ennals (see section 8.2.5 below).

/Ross and Howe ...

Ross and Howe (1981) have evaluated each of Feurzeig's claims in the light of the wealth of Logo research carried out during the 1970's decade. They report evidence - in some cases, strong evidence - in support of each of the four claims. They conclude, however, that:

> **"In terms of statistical results the research [during the 1970's] into 'mathematics through programming' [was] more encouraging than discouraging, but only mildly so."**

At the same time though, they point to the large body of non-statistical evidence which has accumulated in favour of programming as a mathematical activity. Their recommendation is that:

> **"A lot more work [must] be done in order to move the [programming] approach into normal classrooms, and to see whether 'mathematics through programming' can be blended into the existing curricula."**

8.2.5 The Case for Prolog

Kowalski and Ennals (Yazdani, 1984) each concentrate on the logical operations involved in problem solving and argue in favour of Prolog as a medium within which children can exercise the necessary logical reasoning. They report positive research findings regarding the use of Prolog as a vehicle for mathematics education.

Halpin (1986) comments on the question of logic as a discipline for school children as follows:

> **"The study of logic has led to profound insights into the nature of reasoning and the certainty or otherwise of our knowledge. ... Recent developments in Artificial Intelligence have focused attention on a subset of logic which is accessible to children from about the age of 10 years, the study of which encourages and develops logical thinking and problem solving ability and moreover provides an excellent introduction to computer programming."**

He cites Prolog as the best known of the new logic-based languages and goes on to provide examples of the use of Prolog in mathematical problem solving. One of these examples involves the estimating of the area under a curve, showing how Prolog can be used to implement integration techniques. The relevant program is not difficult to construct.

Howson and Kahane (eds.) (1986b) argue the need for the introduction of mathematical logic into the secondary school curriculum. Their reasons are two-fold:

"- in the computer age large numbers of people, particularly in scientific and technical professions, will need to handle statements of a logical or mathematical nature in a very precise fashion;

- now that the teaching of proof in geometry is rapidly disappearing in many countries, we need some place in the curriculum where students learn the art of proof."

Prolog would seem to be an ideal vehicle for the realization of this proposed innovation.

8.2.6 Teacher Competence in Programming

What of teacher competence in the area of problem solving by programming? Since as long ago as 1970 notable educationists, committees and conferences have been arguing the case for programming to be included as a component in mathematics teacher training courses. Examples of the committees and conferences are: (1) The World Conferences on Computers in Education held in Amsterdam, 1970; Marseilles, 1975; Lausanne, 1981; and Norfolk, Virginia, 1985; (2) The Committee on Guidelines of Commission on Pre-Service Teacher Education of the NCTM; and (3) The Conference Board of the Mathematical Science Committee on Computer Education. Among the educationists making this plea are Papert (1973), Minsky (1970), Luehrmann (1972), Dwyer (1974), and Kurtz (1970).

Papert (1972) and Minsky (1970) highlight the general educational value of the 'powerful ideas' inherent in programming concepts such as algorithms, procedures and subprocedures, iterative loops, recursion, bugs and debugging. Luehrmann (1972) classifies these concepts as 'new intellectual structures' upon which to build educational philosophy. Minsky (1970) goes so far as to claim that:

> **"Eventually, programming itself will become more important than mathematics in early education."**

In the light of the preceding discussion it becomes increasingly clear that programming has a strong claim to be considered as a possible part of mathematics teacher education.

P. C. Elliott (1976) points out that the act of programming can lead to the revelation and acquisition of useful teaching techniques. In her words:

> **"Several heuristics involved in the actual programming process suggest strategies for teaching that heretofore have not been highlighted in elementary teacher training. For example, emphasis on breaking difficult tasks into manageable subprocedures is a valuable heuristic for students to use. The concepts of bugs .... and debugging provide valuable lessons in problem solving."**

Milner (1974) provides a comprehensive list of possible gains to be made in the cognitive domain as a result of programming:

(1) Concepts are reinforced and clarified.

(2) The organization of information is facilitated.

(3) Independent thinking is fostered.

(4) An arena for viewing problems in alternative ways is provided.

(5) Logical thinking is promoted.

(6) The articulation of the individual's own ideas is enabled.

(7) The ability to debug is developed.

(8) Generalization skills are expanded.

(9) Problem comprehension is enhanced.

(10) Hypothesis generation is aided.

In addition to the numerous possible cognitive benefits brought about by the introduction of programming into mathematics teacher education, there is a very important gain to be scored in the affective domain. This is illuminated by Elliott (1976):

> **"Prospective teachers are cast in the role of learners with a new programming language to master and old mathematics concepts to reconsider. The awkwardness of being novice programmers is akin to the 'disequilibrium' Piaget [as reported by Ginsburg et al., 1969] says a child feels when confronted with new learning situations. During their periods of disequilibrium, pre-service teachers have an opportunity to develop empathies and sensitivities to children going through 'accommodation' and 'assimilation' periods."**

Elliott refers here specifically to pre-service teachers, but, having had the experience of presenting several programming courses to groups of in-service teachers, the author can testify that her remarks seem to apply equally well to in-service teachers.

8.2.7 Polya's Problem Solving Strategy

As a further indication of the value of programming as an activity in the mathematics curriculum, it is interesting to note that the four steps of heuristic processes described by G. Polya in his classical work, "How to solve it" (Polya, 1973), coincide neatly with the method of problem solving by computer. These steps are:

(1) understand the problem, the unknown, the data, the conditions, etc., and be able to put the problem into suitable notation;

(2) devise a plan (construct an algorithm) for solving the problem;

(3) carry out the plan (code the algorithm into a program and implement it);

(4) examine the results to see if they are reasonable and just.

This fundamental connection between mathematical problem solving and computing has been further cemented and highlighted by R. R. Dromey. Inspired by Polya, he has written an illuminating book, in the spirit of Polya's work, but translated into the computing science context: "How to solve it by computer" (Dromey, 1982).

8.2.8 The Deductive Method of Programming

Bana et al. (1981) depict the process of programming in the following way:

$$\underbrace{\text{Problem} \rightarrow \text{Statement} \rightarrow \text{Algorithm}}_{\text{analysis}}\ \underbrace{\rightarrow \text{Program}}_{\text{programming}}\ \underbrace{\rightarrow \text{Result}}_{\text{running}}$$

They reiterate Polya's problem solving strategy in the form:

(1) express the problem in such a way as to arrive at a statement of terms;

(2) describe the calculations needed to solve the statement;

(3) translate the algorithm thus expressed into the programming language that has been chosen;

(4) run the program so as to obtain the result.

Based on this analysis Bana et al. (1981) discuss the so-called deductive method of programming, introduced by Pair (1979), as a model for a teaching approach to mathematical problem solving. The essence of this approach is to find by deduction what questions should be asked and in what order, as a means of guiding the problem solver towards a solution in a direct, efficient way. The teacher introduces a new concept or problem type by starting with the presentation of several concrete examples which illustrate it. At the next stage in the learning process the pupil must attempt to describe the new problem precisely and so come to a clear understanding of it. After this he must try to find a solution in the form of an algorithm. This often necessitates the study of other prerequisite mathematical concepts.

Thus, as Bana et al. (1981) explain:

> **"we have two complementary lines of study: a 'theoretical' study, using a mainly axiomatic approach and involving the demonstration of a certain number of theorems; [and] an 'experimental' study, which can make use of certain theoretical results and lead to the algorithmic solution of certain problems."**

The theoretical study leads to theoretical results, but facilitates the construction of a program. The experimental study of the program then enables confirmation and reinforcement of these theoretical results. Finally, the solution strategy obtained can be used for solving many other problems of the same type.

8.2.9 Fitting Programming into the Mathematics Curriculum

Before attempting to illustrate the teaching of problem solving by programming, a brief comment should perhaps be made on the perennial question of how to fit a new activity - in this case, programming - into an already full mathematics curriculum. The author's view is that programming should be regarded as a method for problem solving within the existing curriculum. It should be seen as a powerful new tool for enhancing mathematical exploration, rather than as a new topic.

8.3 Sample Programs for Use in Mathematical Exploration

Support for the model propounded by Bana et al. (1981), for teaching problem solving, can be provided via an elaboration of some of the work done by Burrell (1986) in conjunction with the author. Programs for investigating sequences and series and trigonometric graphs will be presented. In addition, programs drawn from other sources for computing roots of polynomials and solving optimization problems will be considered. A salient feature of all these programs is that they are easy to write and implement. The Logo programs in this collection were largely inspired by the work of Riordan (1987).

8.3.1 Sequences and Series

The recursive nature of sequences and series makes these topics ideal for computer application using a language such as Logo with its powerful built-in recursion facility. Consider, for example, the general arithmetic series:

$$S_n = T_1 + T_2 + \dots + T_{n-1} + T_n,$$

where $$S_1 = T_1 = a \text{ and } T_n = a+(n-1)d,$$

with a the first term and d the common difference.

Now S_n can be rewritten as

$$\begin{aligned} S_n &= S_{n-1} + T_n \\ &= S_{n-1} + a+(n-1)d. \end{aligned}$$

This recursive formula leads immediately to a straightforward Logo procedure for summing the general arithmetic series:

```
TO ARITHSUM :A :D :N
    IF :N = 1 [OUTPUT :A]
    OUTPUT :A + (:N-1) * :D + ARITHSUM :A :D :N-1
END
```

Obviously, the general geometric series, with first term a and common ratio r, can be treated in the same way. The corresponding Logo procedure is:

```
TO GEOMSUM :A :R :N
    IF :N = 1 [OUTPUT :A]
    OUTPUT :A + :R * GEOMSUM :A :R :N-1
END
```

The interplay between mathematical reasoning and programming in the process of problem solving is well demonstrated in these examples. Having written the above Logo procedures, we can now investigate the behaviour of any number of arithmetic or geometric series easily and efficiently.

/An enriching ...

An enriching side-effect (or, more aptly, extension) of this approach is that we can now begin to look at other series such as $\Sigma\ 1/n$ and $\Sigma\ 1/n^2$. It is a simple matter to emulate the procedures constructed above in writing:

```
TO SUMINV :N
    IF :N = 1 [OUTPUT 1]
    OUTPUT 1/:N + SUMINV :N-1
END
```

and

```
TO SUMINVSQS :N
    IF :N = 1 [OUTPUT 1]
    OUTPUT 1/(:N * :N) + SUMINVSQS :N-1
END
```

Thus we have established handy tools for investigating, for example, the question of convergence of $\Sigma\ 1/n$ vs $\Sigma\ 1/n^2$ (cf. spreadsheet investigations illustrated in section 7.2.2.6 of Chapter 7).

The crucial feature of programs such as those above is that, in spite of being very simple to construct, they are mathematically very powerful. They can enhance and reinforce the learning of mathematical concepts in a stimulating, dynamic way. The author is inclined to regard such programs as 'laboratory instruments' via which pupils can experiment with mathematics and make their own discoveries.

/8.3.2 Trigonometric ...

8.3.2 Trigonometric Graphs

As another example in support of the model by Bana et al. (1981), let us consider the writing of Logo programs for drawing the graphs of trigonometric functions. For the case of the sine function between 0° and 360° the following procedure is suitable:

```
TO DRAWSIN
   CLEARSCREEN FULLSCREEN
   DRAWSINAXES
   PENUP SETPOS [-100 0] PENDOWN
   MAKE "N 0
   REPEAT 200 [SETX -100 + :N SETY 100 * SIN (1.8 * :N) MAKE "N :N+1]
END
```

The *DRAWSINAXES* procedure draws a suitable pair of axes for the graph. The values 100 and 1.8 in the SETY command within the *DRAWSIN* procedure are simply convenient scaling factors.

In constructing the *DRAWSIN* procedure the pupil would have to think about the underlying mathematics very carefully. The incremental plotting embodied in the use of the SETX, SETY and SIN commands would enable him to appreciate the nature of the functional dependency in y = sin x.

Now let us take the case of the tangent function between -90° and 270°. The following program serves the purpose:

/TO DRAWTAN ...

```
TO DRAWTAN
   CLEARSCREEN FULLSCREEN
   DRAWTANAXES
   PENUP SETPOS [-100 -110] PENDOWN
   MAKE "N 2
   REPEAT 99 [SETX -100 + :N SETY 7 * TAN (1.8 * :N - 90) MAKE "N :N + 1]
   PENUP SETPOS [0 -110] PENDOWN
   MAKE "N 101
   REPEAT 99 [SETX -100 + :N SETY 7 * TAN (1.8 * :N - 90) MAKE "N :N + 1]
END


TO TAN :X
   OUTPUT (SIN :X)/(COS :X)
END
```

The values 7 and 1.8 in the SETY command within the *DRAWTAN* procedure are scaling factors and the *DRAWTANAXES* procedure draws the necessary axes and asymptotes. The use of MAKE "N 101 once :N has reached 100 enables the turtle to 'leap across' the discontinuity at 90°.

Once again, the process of constructing the above procedures helps to make the underlying mathematics - in this case, the behaviour of the tangent function - that much more tactile.

Having reached this stage it is a simple matter to write a Logo program that draws the graph of the cosine function, or any other trigonometric function. Also, it is not difficult to modify the programs so that variable amplitudes, frequencies and horizontal displacements are catered for. The graphs of composite functions such as

$$\sin(x+30), \quad -\tan(2x), \quad 2\sin(x/2) \quad \text{and} \quad \sin 3x + (½)\cos x$$

can then be investigated in an instructive, visually impressive way.

It should be noted that graph drawing programs which use recursion rather than the REPEAT command, thus making the programming more elegant, are easy to write.

8.3.3 Roots of a Polynomial

Fey et al. (1984) provide a simple but noteworthy example of the use of programming for mathematical concept exploration. They illustrate the use of numerical methods for finding the roots of a polynomial. Given the quadratic

$$x^2 - 5x + 3$$

(which is discussed in the context of spreadsheet application in section 7.2.2.4 of Chapter 7), they set up the following straightforward BASIC program:

```
10  INPUT A, B, C
20  FOR X = A TO B STEP C
30  PRINT X, (X - 5) * X + 3
40  NEXT X
```

On running this program initially with A = -5, B = 5 and C = 1, the result produced is:

```
-5      53
-4      39
-3      27
-2      17
-1       9
 0       3
 1      -1
 2      -3
 3      -3
 4      -1
 5       3
```

The left-hand column contains successive values of x while the right-hand column shows the corresponding values of

$$x^2 - 5x + 3.$$

It is immediately obvious that the sign of

$$x^2 -5x + 3$$

changes from '+' to '-' between $x = 0$ and $x = 1$, and from '-' back to '+' between $x = 4$ and $x = 5$. So we have roots in the intervals (0,1) and (4,5). Now, to establish the values of these roots to any required degree of accuracy, all we need do (as with the spreadsheet example) is rerun the program with successively refined values of A, B and C.

Attractive features of an approach such as that above are, as mentioned previously, its utter simplicity and the fact that it can be used in exploring the properties of a whole range of functions.

8.3.4 Optimization Problems

A point which ought to be made here is that programming opens up the possibility for real-world mathematical problem solving. As an example let us consider the following optimization problem adapted from the 1984 Yearbook of the NCTM (ed. V.P. Hansen, 1984):

> A container manufacturing company has been contracted to design and manufacture cylindrical cans for motor oil. The volume of each can is to be 236 ml. In order to

minimize production costs, the company wishes to design a can that requires the smallest amount of metal possible. What should the dimensions of the can be?

The following easy to write BASIC program is adequate as a means for solving this problem:

```
5   LET PI = 3.14159
10  FOR R = 2 TO 5 STEP 0.5
20  LET H = 236 / (PI * R * R)
30  LET A = 2 * PI * R * (H + R)
40  PRINT R, H, A
50  NEXT R
```

In statement 10, R stands for the radius which, for practical purposes, should not exceed 5. In statement 20, H stands for the height and the formula used to derive it is that for volume: $236 = \pi r^2 h$. In statement 30, A stands for the total surface area of the can, derived from $2\pi rh + 2\pi r^2$.

Running the program produces the following table of results in which, from left to right, the columns contain values for radius, height and surface area:

```
2        18.7803     261.133
2.5      12.0194     228.07
3        8.3468      213.882
3.5      6.13234     211.826
4        4.69507     218.531
4.5      3.70969     232.123
5        3.00485     251.48
```

On analyzing this output one sees immediately that the optimal length for the radius is between 3 and 4 cm. To approximate this length to the nearest 0.1 cm it is only necessary to replace statement 10 with

/10 FOR R ...

```
10 FOR R = 3 TO 4 STEP 0.1
```

and re-run the program, obtaining optimal dimensions of 3.3 for radius with 6.9 for height.

Perhaps it is worth noting that in each of the two previous examples, the solution process entails designing a program for carrying out calculations which would otherwise be tedious, and then analyzing the results of these calculations. The computer does not solve the problem but rather it aids the pupil in the process of finding a solution.

8.3.5 132 Short Programs for the Mathematics Classroom

J. Higgo et al. (1985), on behalf of the British Mathematical Association, have published a collection of 132 short programs for use in the mathematics classroom. These programs are written largely in BASIC, most of them in two versions - BBC BASIC and 'general' BASIC. They are very simple and therefore easy to implement and to understand. They free the pupil to study and experiment with mathematical concepts by doing any tedious computations for him. They have further value in that the writing of a program to solve a problem requires the pupil to analyze the problem carefully and should result in his achieving a good grasp of it. The authors intend that the teacher and pupils develop the confidence to write programs of their own or to modify the existing programs.

The example below illustrates a typical program (the first in the book) and highlights the kind of perceptive commentary which accompanies each program and which is designed to provoke further investigations.

/Program 1: ...

Program 1: Triangle numbers

This program prints triangle numbers, that is, the sequence 1, 1+2, 1+2+3, ... At the question mark prompt (line 10) enter the number of triangle numbers required.

```
10  INPUT N
20  LET T = 0
30  FOR C = 1 TO N
40  LET T = T + C
50  PRINT C, T
60  NEXT C
```

To obtain just the Nth triangle number delete line 50 and add

```
70 PRINT N, T
```

The program can be used to find triangle numbers which are also square numbers by replacing line 50 with

```
50 IF INT (SQR(T)) * INT(SQR(T)) = T THEN PRINT C, T
```

INT(SQR(T)) is the whole part of the square root of T. (Note: there are only three square triangle numbers less than 2000.)

8.4 The Introduction of New Concepts Through Programming

Programming can be used, as in the examples above, for the purpose of fashioning powerful tools with which to investigate mathematical concepts - concepts which, perhaps, have been introduced in a conventional way. Another aspect to the use of programming in mathematics education is the medium it provides for the direct introduction and experiencing of new concepts.

R. Noss (1986) reported the results of what he described as an exploratory study which investigated the extent to which Logo provided a small group of 10-year old children with a "conceptual framework", as described by Feurzeig, Papert et al. (1969), to facilitate the learning of algebra, especially the understanding of the concept of variable. His findings are tentative:

> **"children may - under the appropriate conditions - make use of the algebra they have used in a Logo environment, in order to construct algebraic meaning in a non-computational context."**

He goes on, however, to make two rather challenging suggestions for further research action in this direction. Firstly, he suggests the possibility of designing an algebra syllabus based on Logo, that is, a syllabus in which algebraic concepts are introduced through Logo programming. Secondly, and as a consequence of the first suggestion, he suggests the possibility of constructing algebra microworlds for the exploration of algebraic concepts. These microworlds would be complementary to those for geometric exploration described by Abelson and Di Sessa (1980).

Finally, as indicated in section 1.4 of Chapter 1, the learning of a variety of mathematical techniques, concepts and notations can be enhanced by the writing of simple computer programs. Some examples are listed below:

- algorithm structuring and interpreting;
- technical processes like recursion;

- acquisition of new concepts, for example, the concept of variable through recursion as in the Logo procedure *SQSPIRAL* defined by:

```
TO SQSPIRAL :X
   IF :X > 60 [STOP]
   REPEAT 2 [FD :X RT 90]
   SQSPIRAL :X + 5
END
```

  or the concept of exterior angle as in the following Logo procedure for drawing a triangle:

```
TO TRIANGLE
   REPEAT 3 [FD 50 RT 120]
END
```

- notations such as X = X + 1, :X, and word names for variables and functions, for example, LENGTH, AREA, etc.

8.5 Concluding Remarks

An attempt has been made in this chapter to set out a case for the incorporation of a measure of computer programming in some suitable form as an aspect of the mathematics curriculum. The researcher believes it to be of vital importance that mathematics teachers should acknowledge and exploit the fact that computer technology is changing the ways we do and use mathematics. We should react by making appropriate changes to the contents of the mathematics curriculum and the way mathematics is taught. Some ideas for this are presented in Chapter 10.

CHAPTER 9 A REFINED CLASSIFICATION SCHEME FOR COMPUTER PROGRAMS USED IN MATHEMATICS EDUCATION

CHAPTER 9

A REFINED CLASSIFICATION SCHEME FOR COMPUTER PROGRAMS USED IN MATHEMATICS EDUCATION

Reference was made in Chapter 1 to the educational software classification schemes proposed by Taylor (ed.) (1980), Kemmis et al. (1977) and Kansky (1982). Elements of these schemes are apparent in the primitive linear separation of mathematical education computer application categories demarcated by Chapters 3 through 8. Having worked through the process of identifying these categories, then describing and evaluating items of software which exemplify them, it would be appropriate now to try to develop a more refined classification scheme. This is done below following a more detailed exposition of the ideas of Kemmis et al., Kansky and others. Taylor's scheme is assumed to be well known.

9.1 Some Established Classification Schemes for Educational Computer Programs

Kansky (1982) puts forward a three-category framework against which instructional applications of computers can be studied or planned. These categories are "Computer as Teacher", "Computer as Partner" and "Computer as Aide-de-Camp". Each contains several modes or classes of program use as follows:

- Computer as Teacher
    - Tutorial mode;
    - Reinforcement mode (drill-and-practice);
    - Testing operations mode (individualized assessment of students' progress);

    - Instructional management mode (overall management of curriculum delivery on a personalized basis);

- Computer as Partner
    - Simulation mode;
    - Instructional gaming mode;
    - Problem solving mode;

- Computer as Aide-de-Camp
    - Demonstration mode;
    - Informational mode (generation of information, for example, a list of prime numbers, assorted powers or roots of a given number, etc.);
    - Instructional support mode (storage of question banks, or generation of teaching materials, for example, sets of notes, worksheets, overhead projector transparencies, etc.);
    - Programming utility mode (lesson authoring, etc.).

This classification scheme is essentially an elaboration of the familiar "Tutor, Tool, Tutee" paradigm proposed by Taylor (ed.) (1980).

Maddison (1982), as reported by Blease (1986), classifies computer-assisted learning (CAL) into six "learning styles" - by:

- subject;
- mode of presentation (characterized by whether the teacher, the pupil or the computer is in control of the learning process);

- internal technique (including 'models and simulation', and 'information retrieval');
- educational paradigm (whether the major purpose of a program is 'instructional', 'revelatory', 'conjectural' or 'emancipatory');
- psychological theory (whether a program is based on behaviourist theories or on cognitive theories);
- clarity of structure (whether a program is a 'black box' giving no indication of its internal workings, or a relatively transparent 'glass box').

For the purposes of this research project, the most useful of these approaches to classification was found to be that of 'educational paradigm'. This approach, to be considered in detail in section 9.4 below, and developed into a more refined scheme in section 9.5, was originally proposed by Kemmis et al. (1977).

9.2 Relationships Between Mathematical Software Classification Categories

Considering Kansky's, Taylor's and Kemmis's schemes, in conjunction with the software analyses and evaluations reported in Chapters 3 through 8, the following observations can be made:

- any tutor program is either a traditional CAI type program or it is an intelligent tutor, and it may incorporate elements of a game and/or a simulation and/or a drill-and-practice program;

- any tool program is a computational environment/aid and it may incorporate elements of a simulation;

/- any game ...

- any game program may incorporate elements of a simulation and/or a drill-and-practice program;

- any simulation may incorporate elements of a game and/or a drill-and-practice program and/or a tool program;

- any drill-and-practice program may incorporate elements of a tutoring program and/or a game and/or a simulation.

9.3 Differences Between Modes of Computer Use in Mathematics Education

The essential differences between the eight modes of computer use dealt with in this study are listed below:

- intelligent tutors model the user; they explain content and concepts, pose questions and judge user's answers, provide answers to users' questions and give explanations of these answers;

- traditional CAI programs explain content and concepts, pose questions and judge users' answers;

- drill-and-practice programs pose questions and judge users' answers;

- tool programs aid in finding answers to users' questions;

- tutee programs learn how to answer users' questions;

- game programs encourage strategic thinking and provide opportunities for the practising of skills and developing of concepts in recreational ways;

- simulation programs provide good approximations to real-world situations, allowing the user to experiment and solve problems in these simulated environments safely and efficiently;

- CGT programs guide teachers through presentation of topics, posing questions and demonstrating concepts.

9.4 Mathematical Software Classification by Educational Paradigm

In an attempt to arrive at a more refined classification scheme, let us consider in some detail the 'educational paradigm' categories of Kemmis et al. (1977), extended by Maddison (1982), mentioned above. There are four categories in this model: the instructional paradigm; the revelatory paradigm; the conjectural paradigm; and the emancipatory paradigm. These categories are defined as follows:

Instructional Paradigm

The subject material is broken down into many small learning tasks so that the learner can concentrate on each one in turn. Regular reinforcement accompanies the successful completion of each task. If problems are encountered the learner spends more time at this stage or is routed around a 'remedial loop'. The emphasis in this type of program is to divide up the subject material into small enough 'bites' so that the learner has minimum difficulty in achieving success with each response he has to give. (This is the basis on which the old so-called programmed learning was based.)

This paradigm covers CAI tutorial systems and drill-and-practice applications.

Revelatory Paradigm

The pupil is encouraged to explore a model whose subject matter and underlying theory remain hidden. The model is progressively 'revealed' as the pupil goes through a process of discovery learning.

This paradigm covers intelligent tutoring systems and simulations, each of which depend on a fixed underlying model. It can also be considered to include instructional games.

Conjectural Paradigm

The central idea is that the pupil can be the agent of his own learning. He is given the facility to manipulate and test ideas and hypotheses whilst supported by the computer. Control is firmly in the hands of the pupil.

This paradigm covers problem-solving programming environments, that is, tutee programs, and is typified by Logo.

/Emancipatory ...

Emancipatory Paradigm

The emphasis is on reducing the workload so that teaching and learning can take place free of the incidental, often time-consuming, processing of data.

This paradigm covers pre-programmed computational environments, that is, tool programs such as spreadsheets and computer algebra systems. Its definition could even be broadened so as to incorporate CGT applications.

It would be simplistic to insist that these categories are mutually exclusive. There are obvious areas of overlap and this is taken into account in section 9.5 below.

The classification scheme suggested by the above analysis is summed up in the following table:

Global Educational Paradigm			
Instructional	Revelatory	Conjectural	Emancipatory
Drill-and-practice	Games and Simulations	Programming as a mathematical activity	Computational environments for mathematical problem solving
CAI Tutoring systems	Intelligent tutoring systems		

Table 9-1: Mathematical Software Classification by Educational Paradigm

/9.5 A Multi-dimensional ...

9.5 A Multi-dimensional Classification Scheme for Computer Programs Used in Mathematics Education

It is obviously not possible to classify educational applications of computers into mutually exclusive categories. An effective strategy, however, would be to develop an attribute profile for any given program to enable an appreciation of its quality and usefulness. The classification scheme shown in Table 9-1 above involves the attribute 'global educational paradigm'. Another differentiating characteristic is that of 'locus of control' over the learning situation. The degree of user control might be thought to increase from 'low' in the case of 'instructional software' across a spectrum to 'high' in the case of 'emancipatory software'. There are, however, counter-examples refuting this assumption. For instance, the user may have a great deal of control in some instructional programs, and some simulations may allow no user control at all.

Global educational paradigm and locus of control should be seen as independent attributes. It is possible to set up a two-dimensional matrix using these two attributes, with locus of control ranging through three 'values' - program has control; user has some control; user has a great deal of control - and global educational paradigm, as in Table 9-1, ranging through four 'values' - instructional; revelatory; conjectural; emancipatory. A given program could be categorized as fitting into any one of the twelve cells in the resultant matrix. There are, however, other attributes which ought to be taken into account. These are listed below.

In addition to global educational paradigm and locus of control, other attributes useful when characterizing a program are:

- educational purpose: what educational purpose(s) can the program serve?

/- nature ...

- nature of information: with what type of information does the program work?

- mode of information production: how does the program go about producing information?

- program action mode: in what mode(s) does the program function?

- feedback mode: in what form is feedback given?

- reward paradigm: what kind of reward is given?

- teaching mode: in what form is teaching provided?

These attributes, together with the 'dimensions' which define them, are shown in Table 9-2 below:

/Table 9-2: ...

Global educational paradigm			
Instructional	Revelatory	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control	Learner has direct control	Learner has full control

Educational purpose		
Drill-and-practice	Concept building	Problem solving

Nature of information		
Facts	Concepts	Procedures

Mode of Information production		
Retrieval procedures	Object-level procedures	Meta-level procedures

Program action mode		
Requests information	Provides information	Processes information

Feedback mode			
No feedback	Right/wrong	Provides answers	Explains answers

Reward paradigm					
Assessment	Intrinsic	Winning			Personification
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response	Direct Explanation	Instructional Embodiment	Problem solving	Provocation

Table 9-2: A Multi-dimensional Classification Scheme for Computer Programs used in Mathematics Education

The instructional, revelatory, conjectural and emancipatory dimensions of the global educational paradigm attribute are described in section 9.4 above. Where they are not self-evident, descriptions of other dimensions are listed below:

locus of control

the learner either has: no control over program execution; indirect control in the sense that the program responds to his input and executes accordingly; direct control in the sense that he can exercise options which determine the way the program runs; or full control in the sense that the program serves him as a problem solving tool, working entirely according to his instructions;

mode of information production

information is either produced via: straight retrieval of data stored within the program or in a data file; object-level procedures in the form of routines built into the program and encoding knowledge about the facts of the underlying domain; or meta-level procedures which encode control or strategic knowledge and enable the formulation of explicit problem solving strategies (see section 4.2.1.2, Chapter 4);

feedback mode

the program might give no feedback at all, or it might give feedback after judging the learner's input in any of the following ways: an indication that a response is either right or wrong; an indication of what the right answer is; or generation of the answer step-by-step with explanations;

/reward paradigm ...

reward paradigm

the program might reward the learner simply by assessing his response(s) and possibly awarding scores; reward might, on the other hand, be purely intrinsic (for example, satisfaction at having completed a task successfully); if the program is in the form of a game, the reward might lie in the thrill of winning against another player or against the computer, or it might lie in the pleasure of bettering one's own previous best score; reward can also take the form of congratulatory messages offered by the program - a personification of the computer;

teaching mode

the program might offer teaching in any of the following ways: the stimulus-response approach, that is, providing the same stimulus over and over until a correct response is learned; the provision of direct explanations in response to a learner's requests for aid in problem solving; reacting to a learner's attempts to solve a problem after being instructed to do so by the program, that is, problem solving as a teaching mode; teaching via instructional embodiment, for example, using analogs such as equations embodied as 'invaders' in the program *EQUATORS* (see section 6.2.5), or using schematics such as the graphical representation of number machines in the program *NUMBER MACHINE* (see section 6.3.3); teaching can also occur by provocation, that is, by provoking the learner to develop problem solving procedures (this is well illustrated by the program *AUTOFRAC* (see section 3.3.2.2)).

/9.5.1 Application of ...

9.5.1 Application of the Multi-dimensional Classification Scheme

The classification scheme presented above was developed from a prototype by testing the effectiveness of this prototype in establishing attribute profiles for a number of programs described in this thesis. Where an attribute was found to be missing or in some way inadequate, the necessary modifications were made until the prototype was considered to have become a fully fledged classification scheme.

The effectiveness of this scheme can be judged by considering the attribute profiles it produces for a selection of programs. This is done below, the programs chosen being those of the *SASEC, SERGO* and *TOAM* collections, plus *DIRECTED* and *AUTOFRAC*, presented as drill-and-practice programs in Chapter 3. What follows is a discussion of some of the salient features of the attribute profiles of these programs. Comparisons are drawn between their classifications according to the multi-dimensional scheme and those according to the original, more primitive, linear scheme. *FRAC* (Chapter 3), *SKETCHER* (Chapter 4), and *EQUATORS, NUMBER MACHINE*, and *GREEN GLOBS* (Chapter 6), have also been classified against the multi-dimensional scheme but, so as not to belabour the issue, no discussion on their attribute profiles is included. The actual classification forms for all of these programs are to be found in Appendix A8.

The *SASEC, SERGO* and *TOAM* program collections were classified in Chapter 3 as being simply of the drill-and-practice type. The attribute profiles generated by the multi-dimensional scheme provide much more information. Apart from having drill-and-practice as their educational purpose, these programs are instructional as far as global educational paradigm is concerned and they allow learner control only in an indirect way. The nature of the information dealt with by these programs is factual and they produce information by means of retrieval. Their program action mode involves both requesting and providing information. They give feedback in the form of 'right' or 'wrong' judgements or pre-stored answers. Reward is given via assessment and congratulatory messages (so-called personification). The teaching mode is stimulus-response.

DIRECTED was also classified in Chapter 3 as being a drill-and-practice program. The multi-dimensional scheme, however, identifies it as having concept building as its educational purpose. It fits into the revelatory global educational paradigm. It allows direct user control in the form of options to select. The information it deals with is in the form of facts and concepts. It produces information by means of object-level built-in procedures. Its program action mode involves requesting, providing and processing information. Feedback is via the provision of pre-stored answers and also by demonstration of the generation of answers. Reward involves both assessment and personification. Teaching takes the form of instructional embodiment of the analog type, a robot animating and demonstrating the relevant mathematical processing.

AUTOFRAC, classified as a drill-and-practice program in Chapter 3, is now seen to involve concept building as well. It is revelatory. It allows direct learner control in the sense that the learner can choose to perceive the program as both requesting and providing information or as just providing information. In the former case the learner is provoked into trying to find answers to problems posed by the program. In the latter he would make no attempt to solve problems, waiting simply for the program to provide the answers. *AUTOFRAC* deals with both facts and concepts. It produces information by means of object-level built-in procedures. Its program action mode is information processing. Feedback is in the form of answer provision. Reward is intrinsic and teaching is by provocation.

/9.6 Concluding Remarks ...

9.6 Concluding Remarks

The main purpose of this study was to provide a conceptual framework, an analytical tool, which could facilitate non-prejudice in software evaluation and use, by promoting understanding of a program through its classification before it is judged. It is hoped that the multi-dimensional scheme described above contributes towards the achievement of this purpose and that, using it, the teacher would be made aware of the range of potential uses of any given item of software, so that teaching opportunities would not be missed.

As mentioned in section 2.2.1 of Chapter 2, Slaughter (1989) has developed a cataloguing system for microcomputer diskettes containing educational software, in particular that which can be used in mathematics education (see Appendix A9). It is intended that this system should be enhanced by the inclusion of a summarized attribute profile for each software item.

CHAPTER 10 COMPUTER INSPIRED MATHEMATICS CURRICULUM CHANGES

CHAPTER 10

COMPUTER INSPIRED MATHEMATICS CURRICULUM CHANGES

The computer has the potential to influence mathematical activities across the full spectrum, from the most elementary levels right up into the realms of high-powered research. In the words of Howson and Kahane (eds.) (1986a):

> **"It is clear to us that the computer is having, and will continue to have, a significant impact on the directions of mathematics research and that computers will not only be commonly used to arrive at conjectures but also to assist in finding proofs."**

A famous example of the use of computers in mathematics research is Appel and Haken's (1976) proof of the 4-colour problem by exhaustive examination of the large but finite set of all possible cases. Another fascinating computer-generated result is that by Lander and Parkin (1967):

$$27^5 + 84^5 + 110^5 + 133^5 = 144^5,$$

disproving Euler's 200 year old sum of powers conjecture.

Computers are making a major impact in many branches of mathematics, amongst the better known of which are group theory, combinatorics and number theory. Simple examples are the search for large primes and the factorization of large integers. The latter, although seemingly dull at face-value, is an important aspect of cryptography.

/In the light ...

In the light of developments like these, it becomes apparent that mathematics educationists will have to analyze the curriculum at school and university level with a view to incorporation of computer-based mathematical activities. Fey et al. (1984) conclude that:

> **"Looking at the current capabilities of widely available computing equipment and software and making only modest projections of likely future developments, one is forced to question the content and emphases in school mathematics programs today. Must students still acquire the traditional collection of mathematical skills and ideas ...? Are there new skills or understandings that should take prominent roles in courses preparing students for mathematical demands that lie in the twenty-first century?"**

The computer would not be of much value in the mathematics classroom unless it was carefully integrated into the curriculum. The teacher who wished to exploit the educational potential of the computer would have to be perfectly clear in his own mind as to how, where and why it should be used. The computer, rather than replacing conventional problem-solving activities supervised by the teacher, could form a powerful tool for augmenting and enriching them. It could act as a mathematical 'laboratory' within which, under the guidance of the teacher, the pupil could experiment with mathematical concepts and make his own discoveries. In this way mathematics could become a personalized, exciting and highly meaningful activity.

Several possible applications of the computer in secondary school mathematics have been explored in the previous chapters. Let us turn now to a consideration of some of the curriculum changes that might be inspired by these applications.

It should be noted that the ideas presented in this chapter are strongly influenced by the seminal work of Howson and Kahane (eds.) (1986a) as well as that of Fey and Heid (1984)

/and Fey ...

and Fey et al. (1984).

10.1 Computer Enhanced Algebra

The existing syllabus (for example, Cape Education Department, 1985) - as does even the proposed new syllabus (Department of Education and Culture, 1989) - lays stress on procedures for transposing algebraic expressions and solving equations and inequalities. Now, it would seem that in much the same way that the electronic calculator has become an accepted tool for augmenting the numerical processing power of the pupil, a system such as *muMATH* could be used to augment his algebraic processing capacity. The calculator, by doing all the necessary routine and often rather mechanical numerical processing, enables the pupil to become involved in creative and more meaningful problem solving. The computer can play just this role in the domain of algebra. In the same way that the calculator has assumed a central position in the planning of the proposed new syllabus, the computer will have to be taken into account when this new syllabus is updated perhaps in only a few years' time - and this applies not only to algebra, but to all components of the mathematics syllabus.

Although *muMATH* is at present restricted to running on microcomputers such as the IBM PC, it is likely that it and similar systems will eventually (perhaps within just a few years) become commonly available on relatively inexpensive pocket-sized computers. Today's computer is tomorrow's calculator! This means that a sizeable and ever growing number of school pupils will be able to possess their own exceptionally powerful, versatile and sophisticated computers. In the same way that mathematics and those who teach it 'survived' the inventions of Arabic numerals, multiplication tables, logarithms, ..., and pocket calculators! we shall have to come to terms with, harness and exploit the advent of computer algebra.

/It is possible ...

It is possible that pupils using a computer algebra system such as *muMATH* would be enabled to acquire a thorough and, in the practical sense, more readily applicable knowledge of aspects of mathematics, which at present are often assimilated and perhaps inadequately learned through a process of repetitive imitation - faithfully and sometimes mechanically following examples provided by an instructor. Is it not possible that natural and intuitive mathematical powers are in fact inhibited by the formal discipline of learning and memorizing computational rules and formulae? Having accepted that the regular use of a calculator does not necessarily stultify a pupil's ability to do mental or even written arithmetic - it can enhance his powers dramatically - we should be giving serious consideration to the possible role of the computer in the mathematics curriculum, particularly with respect to algebra.

Fey et al. (1984), reporting on preliminary investigations with *muMATH* at the University of Maryland, claim that:

> **"Students who have previously been stymied by the difficulty of acquiring various manipulative skills have learned and successfully used advanced mathematical ideas."**

Another point that should be made when discussing actual classroom implementation of a powerful computer algebra system such as *muMATH* is that it could be used in solving problems which, although not conceptually difficult, are not practically possible because of their complexity. The pupils could use the system to tackle problems relating to *real* situations and involving, for example, large sets of simultaneous equations containing 'awkward' coefficients. In other words, the system could be used in the investigation of *real* problems which, because of their complexity, are normally omitted from conventional mathematics courses in which the parameters of problems are made artificially simple for the sake of calculation tractability.

Perhaps our current emphasis on the mastery of calculational skills before applications and problem solving is misdirected. In the words of Fey et al. (1984):

> **"... if one allowed computational aids in the manipulative aspects of answer finding, courses would not have to be organized to develop extensive technical skills before encountering realistic or appealing problem materials."**

Using an approach like this, the emphasis could be shifted to the more creative tasks of studying the interplay of conditions, thinking about functional relationships, interpreting graphical information, etc. Attention could be given to the crucial, yet all too often neglected, process of mathematical modelling. In other words, by reducing the time spent on mastering manipulative skills, time could be made free for applied problem solving.

Z. Usiskin (1980) in his analysis of topics that should <u>not</u> be in the first-year algebra curriculum of college-bound students in the USA makes the following rather illuminating remark regarding the manipulation of algebraic fractions, a topic normally given prominence in secondary school syllabuses:

> **"The addition, multiplication, and simplification of complex fractional expressions is a skill with utility for only a very limited number of students far in their mathematical future."**

There are essentially two main algebraic problem solving methods: formal - symbolic manipulation - and numerical. The computer can enhance the application of each of these considerably.

/10.1.1 Computer Symbol ...

10.1.1 Computer Symbol Manipulation

Computer algebra or symbol manipulation, as discussed above and in Chapter 7, involves the use of computers to carry out the routine algebraic manipulation in, for example, expanding and simplifying expressions, factorizing expressions, solving equations, and differentiating or integrating expressions. It is not limited to performing calculations involving constant values; it works in fact with variables and produces results in terms of these variables. Using, for example, the *muMATH* system the computer becomes a high-powered symbolic calculator with the capability of performing virtually any of the algebraic transformations listed in the secondary school syllabus.

Fey et al. (1984) have made a study of the impact of computing on secondary school mathematics curricula. They propose the following guidelines for change when considering the common core algebra syllabus:

"1. Student ability to manipulate algebraic expressions into alternative forms needs to cover only the simpler cases of useful operations - those that are done more efficiently by hand than by machine, and those that are needed to give students an adequate understanding of the computer-generated results. For instance:

a) Properties of exponents like $x^n.x^m = x^{n+m}$ should be taught. Simple applications like

$$3^5.3^{10} = ?$$

or

$$(a+2)^2(a+2)^5 = ?$$

seem appropriately done by hand, but there seems little need for student skill at simplifications like

$$\frac{40x^3y^2z^3}{x^5yz^9}\cdot\frac{xy^8z^{-5}}{6y^{10}z^{-3}}$$

b) Simplification of radicals like $\sqrt{[20x^3y^6]}$ and the procedures for rationalizing denominators seem of little importance.

c) Factoring of quadratic trinomials like x^2+5x+6 and certainly more complex variations, seems unimportant; factoring using a single application of the distributive property, as in $ab+ac = a(b+c)$, seems valuable for students to learn.

d) Combination and simplification of rational expressions beyond the simplest cases seem of little importance. For instance,

$$\frac{a}{b} + \frac{c}{d} = \frac{ad+bc}{bd}$$

should be known by most students, but skill with expressions like

$$\frac{2x+2}{x+1} - \frac{x+4}{x+3} + \frac{x+2}{x+3}$$

seems of far less importance.

As a general rule, it seems reasonable that students should be able to perform, by hand, one-step applications of the basic properties governing

/algebraic expressions ...

algebraic expressions.

2. Likewise, it seems reasonable that students should learn equation-solving procedures that cover those cases most efficiently done by hand or those that provide a minimum understanding for the use of computer results. For instance:

   a) Linear equations and inequalities in one variable should be readily solved by all students. Some procedure for solving quadratics (probably the quadratic formula) would open the access to a rich supply of important applications. There seems to be little need for students to learn all the traditional methods for solving polynomial equations or any but the simplest examples of rational and algebraic equations.

   b) The notion of a system of equations and some basic tools for solving systems remain important, but techniques for dealing with complex cases seem of little importance."

This list of proposed changes can, of course, be criticized on several counts - as could any other list for that matter - but it does at least provide a concrete starting position for the necessary debate. The authors themselves highlight three ensuing issues requiring urgent research:

- How much hand-manipulation skill can be regarded as sufficient?
- Will the deletion or diminishing of certain topics in high school algebra have a detrimental effect on other aspects of the mathematics curriculum?
- Will simplifying the demands of school algebra stunt the development of the habit

of working with precision and accuracy and the development of ability in abstract reasoning?

It would not be difficult to list several pros and cons in arguing about each of these questions, but solid research is necessary in order to find valid answers.

10.1.2 Numerical Methods

As indicated in Chapters 7 and 8 the computer can be used to good effect in implementing numerical methods for algebraic problem solving. Apart from writing one's own fairly simple programs for this, there are several well developed, powerful and easily available program packages ideal for the purpose. These include spreadsheets, *TK!Solver* (discussed in section 7.2.1 of Chapter 7) and systems tailored specifically to the solving of linear programming problems.

Optional modules on numerical methods have been incorporated in the proposed new senior secondary syllabus (Department of Education and Culture, 1989) at the standard 10 level for implementation in 1994. These include the following topics:

Errors: sources of error; absolute and relative error; error propagation.

Significant digits: round-off and truncation errors.

Solution of non-linear equations:

(i) What is meant by acceptable accuracy?

(ii) Interval halving; Regula Falsi.

(iii) Picard iteration method.

(iv) Newton-Raphson method.

(v) Secant method.

Calculation of the area under a curve by:

(a) using the Trapezium rule;

(b) using Simpson's rule;

(c) "sandwiching" the area between the upper and lower sums.

Calculation of areas between:

(a) a curve and the x-axis;

(b) two curves;

(c) a curve and the y-axis.

Volumes of revolution about the x- and y-axes.

These topics all lend themselves admirably to computer implementation and, again, hard research is required to test the efficacy of the computer's role in this aspect of the curriculum.

10.1.3 Possible New Sequences for Algebra Topics

Given that the use of the computer dispenses with the need for pupils to develop technical skills before applications and problem solving can be tackled, it becomes possible to re-sequence topics in the algebra syllabus in a novel and perhaps more productive way. As an example, Fey et al. (1984) suggest the following inverted order of approach for dealing with quadratic equations:

"Stage 1: Where do quadratic equations arise?

- Scientific principles like projectile motion.
- Nonlinear revenue, profit, demand, or cost functions.
- Fitting rules to curvilinear data.

Stage 2: How does one solve an equation like:

$ax^2+bx+c=0$?

- Guess-and-test successive approximation.
- By hand to illustrate ideas of solution possibilities
    - By graph
    - By computer for refined search.
- Formal solution by computer
    - *TK!Solver*
    - *muMATH*.

Stage 3: Formal analysis of solutions to the quadratic:

- Number and type of possible roots.
- Relations of coefficients and roots.
- Formal manipulations, including factoring and quadratic formula."

Perhaps the biggest advantage to such a scheme, as summarized by Fey et al. (1984), is that:

"No longer must all students struggle through years of preparation for a promise of applications in the future."

Another important consideration is that teaching of the formal facets of a topic could now be delayed, so that only those pupils with the necessary aptitude and interest need be concerned with them, and at a more appropriate stage than is usually the case at present.

10.2 Computer Enhanced Geometry

The geometric capabilities of even small microcomputer systems are impressive. There is available a powerful array of two-dimensional graphics representations and a more limited but rapidly developing collection of three-dimensional applications. Inexpensive graphics tablets connected to computers even provide the facility for calculating the area of any closed geometric shape simply by tracing its shape on the tablet with an electronic pen. Transformation geometry programs, such as the WAMSOFT *GEOMETRY I* discussed in Chapter 7, enable us to display a shape and then transform it by rotating, translating, reflecting, expanding, etc. More sophisticated systems of programs such as *MATHCAD* allow us to view a geometric solid from one perspective and then dynamically transform it to be viewed in another position. Programs such as the *GEOMETRIC SUPPOSER* (described in section 7.5.1 of Chapter 7) help us to compute the measures of the missing parts of given figures. In addition, of course, the Logo system opens up possibilities for exploring geometric concepts at elementary and advanced levels in a simple, visually dynamic and effective way.

/10.2.1 Possible New ...

10.2.1 Possible New Approaches in School Geometry

10.2.1.1 Concept Investigations via Graphics

Most of the opportunities for computer inspired curriculum innovation in geometry arise from the computer's very powerful graphical capabilities. As stated by Fey et al. (1984):

> **"The power of computers to deal with complex, dynamic geometric settings forces students to acquire more sophisticated geometric understanding themselves. It seems unlikely that someone whose geometry knowledge is limited to measuring simple plane figures and proving the congruence or similarity of triangles will be able to contribute to, or cope with, the opportunities and challenges provided by computer graphics."**

There is strong motivation for breaking away from the traditional, highly artificial approach to geometry involving deduction of geometric facts and principles from a carefully planned set of axioms and definitions, that is, standard Euclidean Geometry. This is not the way mathematicians investigate and develop their subject. It is important for pupils first to gain an intuitive understanding of geometric concepts and the relations among structures so that they can reason intelligently using those ideas when attempting to solve problems. There is a growing body of research evidence to support the view that the development of geometric intuition and the conjecturing spirit are essential pre-requisites for geometric problem solving. Theory building should be the last rather then the first phase in this logical progression.

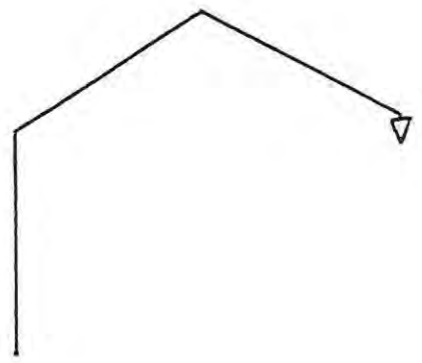
Like the *GEOMETRIC SUPPOSER* another computer program supporting the intuitive, exploratory approach is that by Purchase (1986), which was investigated by von Ludwig (1986). This program facilitates pupil explorations of similarity theorems.

/Perhaps ...

Perhaps the epitome of a programming system which promotes the building of pupils' visual intuition and conjecturing of theorems is the turtle geometry in Logo. Explorations within this medium can lead pupils beyond the intuition gained from manipulating shapes to the conjecture of major theorems. For example, at a simple level, the very common Logo entry point for beginners embodied in the instructions

FORWARD 100 RIGHT 60

repeated three times in an attempt to draw an equilateral triangle, but leading instead to



and the realization that one is working with exterior angles, results usually in an affirmation, if not discovery, of the fact that the sum of the interior angles of a triangle is 180°. Fey et al. (1984) give another elementary example in which pupils, knowing only what a parallelogram looks like, can achieve an illuminated awareness of the symmetry and angle-side relations in any parallelogram by developing a procedure for constructing a parallelogram such as the following:

```
TO PARM :SIDE1 :SIDE2 :ANGLE
  REPEAT 2[FD :SIDE1 LT :ANGLE FD :SIDE2 LT 180 - :ANGLE]
END
```

Several other examples of Logo supported geometrical concept exploration are given by the author (Marsh, 1986) and the author in collaboration with a colleague (Marsh and Halpin, 1986a). These include programs which enable investigation of the effects of drawing the medians of any triangle or joining the midpoints of the sides of any quadrilateral, and a program for drawing the circumcircle of any polygon.

de Villiers (1989) pursues the Logo-based exploratory approach very much further. He develops generalizations of figures produced by the familiar Logo procedure 'TO POLY'. He attempts to formulate a workable definition of these generalized figures and to establish a useful classification of them. In the process he makes personal discoveries of some of the important concepts involved in the topology of curves in the plane.

Perhaps emphasis ought to be given, if it has not already been done implicitly above, to the centrality of the notion that a circle contains 360° and the way that turtle geometry can enhance the development of insight in this connection. As Fey et al. (1984) explain:

> **"Students, even very young children, who work with Turtle geometry and who write their own Logo procedures, soon begin to see the significance of measures of angles that are factors of 360° or some multiple of 360°. Designs formed by recursive procedures in which striking geometric patterns are formed are seen to depend on how many times the procedure must be repeated before a multiple of 360° is reached ... Relationships between certain polygons, such as the equilateral and 30°-60°-90° right triangles and the regular hexagon soon become clear."**

/10.2.1.2 Transformation Geometry ...

10.2.1.2 Transformation Geometry

As mentioned in Chapter 7, in the discussion of the WAMSOFT *GEOMETRY I* programs, the computer opens up exciting possibilities for revitalizing our rather staid and steady, and certainly very static, geometry syllabus by the infusion of dynamic, and in many ways, more relevant, aspects of transformation geometry. Fey et al. (1984) point out that:

> **"... the transformation approach makes geometry an appealing, dynamic subject that will develop spatial visualization ability and also the ability to reason."**

Apart from the WAMSOFT programs there are several other software systems which provide viable support for this purpose. Examples are those by Shilgalis and Thompson. T. Shilgalis (1982) developed a BASIC program which enables all the standard transformations of translation, rotation, reflection, glide reflection and dilation singly or as composites. P. Thompson (1982a, 1982b), using Logo, designed a similar but more comprehensive system for the same purpose. An attractive feature of systems such as these is that they can be used not only for the provision of practice in working with the relevant concepts, but more particularly as vehicles for introducing the concepts in the first place.

It should also be mentioned that transformation geometry yields elegant, simple and direct proofs of many of the standard Euclidean theorems currently taught at the secondary level. Cases in point are the midpoint theorem and the theorem concerning the angle at the centre of a circle being twice the angle at the circumference subtended by the same arc.

When discussing computer implemented transformation geometry, a related issue is the possible greater attractiveness and added meaning given to coordinate and vector methods when handled by computer. The embodiment of coordinates in vectors and the subsequent transformation methods by matrices are of vital importance to linear algebra and its applications in, for example, business, communications, statistics and biology. The associated processing, although straightforward in broad concept, usually involves very complex manipulations and these can be done accurately and efficiently by computer, freeing the student to develop skills in interpreting results; the results generated for *real* problems, not the usual artificially simplified situations that mathematics teachers have been compelled to deal with.

Referring to vectors again, perhaps it should be noted that the Logo turtle is in many ways an ideal vehicle for investigating and reinforcing the underlying concepts. The instructions by which the turtle is moved are specifications of vectors in the plane. The addition and subtraction of vectors and the multiplication of a vector by a scalar are easily illustrated using the dynamic Logo approach. In addition, in much the same way that many of the classical theorems of Euclidean geometry can be proved by innovative, elegant and efficient transformation geometry methods, they can be proved in appealing ways by vector methods, using the computer as a demonstration medium.

10.2.2 Possible New Topics in School Geometry

The new approaches to geometry, made possible by the introduction of the computer into the curriculum, open up several new topic possibilities. As with the algebra, there is a need for thorough research in this area of curriculum development. Fey et al. (1984) list the following computer supported topics for consideration:

- the use of transformations to explore congruence and similarity;

- the use of coordinate methods;
- the use of vector methods;
- computer-based explorations to develop geometric intuition and to stimulate the discovery of important facts and principles;
- the discrete geometry of graphs and networks;
- solid geometry and its relation to the physical world;
- new approaches to learning logic and proof by algorithm design and computer programming;
- the merging of trigonometry into the geometry course.

An obvious caveat in this connection for curriculum developers is that proposed changes in content of, and approach to, school geometry would have to be carefully analyzed in order to judge their effects on other aspects of the mathematics curriculum in particular, and the overall school curriculum in general. Given the impending introduction of our proposed new senior secondary syllabus (Department of Education and Culture, 1989) with its modular structure, the time seems opportune for sustaining this new and encouraging impetus in curriculum development by experimenting with the ideas discussed above.

10.3 Computer Enhanced Calculus

10.3.1 The Role of Calculus

The importance of calculus as an aspect of the secondary school mathematics curriculum is a well established fact. Curriculum planners recognize the central role played by calculus in modelling change in the physical world and enabling the study of continuous phenomena. Such disparate fields as physics, engineering, the life sciences, the social sciences, business and architecture are all influenced and facilitated by the application of calculus strategies and techniques. Within mathematics itself calculus fulfils a synthesizing function, melding together algebra, geometry and trigonometry, encompassing functions, graphs and co-ordinate methods. Effective learning of calculus aids the development of geometric intuition, the establishment of a dynamic view of geometry.

There is a difference, however, between theoretical ideals concerning calculus and the actual implementation of the curriculum. Calculus courses, generally, are presented in a rather formal way, there being very little reliance on intuition in the treatment of such concepts as limits, derivatives and integrals. Very little preliminary numerical experience is encouraged. The problem solving approach adopted is largely algorithmic, with an emphasis on formula manipulation rather than concepts. The applications considered are often of a contrived nature: ladders sliding down walls; water flowing into and out of tanks; farmers fencing rectangular fields; etc. There is not much opportunity for building mathematical models for *real* problem solving.

10.3.2 Calculus Supported by Computer

The computer is a powerful tool for calculus problem solving. There is readily available and well developed software for a variety of applications, including the computing and displaying of critical features of graphs, the solution of differential equations, numerical integration and algebraic manipulation of relevant formulae. There are even hand-held microcomputers (or ultra-sophisticated electronic calculators) which can calculate values of elementary functions and display high-resolution graphs.

/Another dimension ...

Another dimension to the use of the computer for calculus purposes is the opportunity it provides for exploring differential geometry in an elementary way. At the heart of calculus is the study of instantaneous motion and an expedient device like the Logo turtle facilitates this process by requiring of the user driving the turtle to refer "only to the difference between where the turtle is now and where it shall momentarily be." (Papert, 1980). The essence of this approach is that growth is described by what is happening at the growing tip. The idea is well illustrated by a differential approach to drawing a circle as embodied in the Logo procedure:

```
TO CIRCLE
  IF HEADING > 0 [STOP]
  FORWARD 1 RIGHT 1
  CIRCLE
END
```

10.3.3 Possible New Approaches in School Calculus

As with the potential influence of the computer on the curriculum in algebra and geometry, there are fundamental considerations regarding the planning of a calculus curriculum which capitalizes on the use of the computer. Perhaps, for example, students should be expected to do by hand only simple cases of basic procedures. Complex procedures could be done by computer, with interpretation and understanding being of paramount importance. Fey et al. (1984) suggest the simple cases as being:

"1. those that can clearly be done more efficiently by hand;

2. those that are needed to illuminate important aspects of theory;

3. those that are needed routinely in simple applications."

/In their words ...

In their words:

> **"... students, using the computing power available to them, could concentrate on the less automatic features of problem solving: setting up the problem, analyzing and interpreting results, creating and solving realistic problems, and learning to apply the concepts of calculus to the dynamics of ordinary situations."**

This approach is also advocated by Howson and Kahane (eds.) (1986a) who contend that:

> **"In the teaching of calculus to all students the need is clear for a shift from an emphasis on calculational technique to one which emphasizes the development of mathematical insight."**

Software in the form of graph plotting programs, spreadsheet systems, algebraic manipulation systems, etc., could enhance the understanding of basic concepts and the development of intuition. The computer could be used to provide dynamic illustrations of the traditionally developed theory surrounding functions and graphs, limits, derivatives and integrals. It could facilitate the investigation of *real* applications involving the attendant practical questions of estimation and approximation using techniques such as the trapezoidal rule and Simpson's rule.

As in the case of algebra, the computer offers the opportunity in calculus for the re-sequencing of activities, starting with applications and moving onto techniques and concepts.

Once again the research issues are fundamental. The major themes unifying calculus concepts and strategies must be identified. Cross-curriculum calculus requirements must be analyzed. An answer must be sought to the question: "Is the ability to perform mechanical procedures a prerequisite for understanding and problem solving?" Lastly, but perhaps most importantly, thorough testing of computer efficacy in the calculus classroom must be carried out.

10.4 Implications for Teachers, of Computers in the Mathematics Curriculum

The opportunities for mathematics curriculum innovation offered by the computer are vast and exciting. The introduction of computer activities into the mathematics curriculum, however, will have to be planned very carefully. Traditional teaching approaches, even down to the physical lay-out within classrooms, will have to undergo major revision. Teachers will, of course, require intensive in-service training. As Howson and Kahane (eds.) (1986a), drawing on Winter (1985), point out:

> **"To take full advantage of the use of computers in teaching [mathematics] it will be necessary to change the standard classroom environment. Classrooms need to be provided with large monitors or screens on which the monitors may be projected. Outside the classroom, students [will] need administratively easy and user-friendly access to computers and software. Teachers will need private computer facilities to be used to prepare course material. A pre-requisite for this is in-service training so that teachers may become comfortable with computers and then fluent in their use ..."**

Computers can greatly assist in extending the range of pupils' mathematical activities: they can support observation, exploration, the forming of insights and intuitions, the making of predictions, the testing of hypotheses, etc. It is, however, necessary to find an appropriate balance between these experimental activities and those which are more formal such as proving, generalizing and abstracting.

Classroom use of the computer places considerable demands on the teacher. Obvious requirements as detailed by Howson and Kahane (eds.) (1986a) are that:

/"... teachers ...

> **"... teachers should not only acquire new knowledge, skills and confidence in the use of hardware and software, but ... they should also radically change their present aims and emphases, and accept a lessening in the degree of control which they presently exert over what happens in their classrooms."**

If the computer is to be incorporated into the mathematics curriculum, there are substantial resultant implications for pre- and in-service training of mathematics teachers.

CHAPTER 11 CONCLUSIONS AND RECOMMENDATIONS

CHAPTER 11

CONCLUSIONS AND RECOMMENDATIONS

This study set out to establish a framework for software design, selection and use in the field of computer applications in secondary school mathematics. This aim has been accomplished in that:

- the various fundamentally different modes of using computers in mathematics education have been identified and described;

- programs exemplifying the various modes have been identified and their possible educational uses described and, in many cases, evaluated;

- standards for mathematical software evaluation have been developed;

- various perspectives on the computer in mathematics education have been broadened and synthesized into a composite whole;

- possible mathematics curriculum changes implied by the availability and use of computers have been identified and described.

The framework thus provided is intended as a means for helping secondary school mathematics teachers in South Africa to exploit the educational advantages offered by computers purposefully, meaningfully and professionally.

/11.1 Modes ...

11.1 Modes of Computer Use in Mathematics Education and their Classification

The various fundamentally different modes of computer use in mathematics education identified in this study are:

- drill-and-practice;
- traditional CAI;
- 'intelligent' tutoring;
- computer guided teaching (CGT);
- games;
- simulations;
- pre-programmed computational environments;
- programming.

These modes are characterized by fundamental differences but they are not mutually exclusive. Each mode has been described and exemplified in some detail and an attempt has been made to classify these modes into a conceptual framework promoting optimal use of computers in mathematics education.

A multi-dimensional classification scheme has been proposed. The attribute profiles generated by this scheme enable fine perceptions of the potential uses and educational value of items of software. It is, however, suggested that the need exists for further research to be carried out in this direction. In this connection, it should be mentioned that an extensive literature search (through the ERIC data-base) did not yield any references dealing specifically with the classification of educational software.

/11.1.1 Suggestions ...

11.1.1 Suggestions for Enhancing the Proposed Classification Scheme

Two suggestions of possible ways in which the classification scheme proposed in this thesis could be enhanced are given here. Firstly, more cognizance could be taken of the triangular relationship between teacher, pupil and computer (see section 5.3 of Chapter 5). The following aspects of the process of teaching and learning mathematics could be investigated explicitly:

- the design and setting of questions
  - by the teacher;
  - by the computer;

- the production of answers (making a plan; executing the plan)
  - by the pupil;
  - by the computer (setting the pupil free to concentrate on problem solving);

- the evaluation of answers
  - by the teacher;
  - by the pupil;
  - by the computer (setting the teacher free for more important teaching tasks).

Secondly, a facet which could be taken into account when analyzing modes of computer use in mathematics education is that of the type of thinking engendered. Any computer application in mathematics education necessarily involves three types of thinking or intelligent activity: thinking about the computer (software thinking); thinking mathematically; and thinking pedagogically.

/The pupil ...

The pupil may be involved in mathematical and even in software thinking. He may think mathematically to answer questions posed by a program, or to solve a programming problem when investigating mathematical concepts using, say, Logo. He may use software and mathematical thinking in order to write a program for solving a mathematical problem.

The teacher and the external author (program designer) may be involved in all three forms of thinking.

A distinguishing feature of learning and/or teaching with the aid of computer programs, however, is the fact that a program may include routines which effectively simulate both mathematical and pedagogical thinking.

11.2 Computer Software Resources for Mathematics Education

Many items of computer software suitable for use in mathematics education have been identified, described and classified. Several of these have also been evaluated. Examples representative of each mode of application have been presented.

The field of computer-based mathematics education is still in its infancy and if it is to become a viable facet of the curriculum, the need exists for appropriate good quality software to be produced in much larger quantities and more rapidly than is the case at present. The conceptual framework developed in this thesis could serve as a blueprint against which software could be created and evaluated. It is recommended that further research be undertaken in an attempt to implement this proposal and assess its feasibility.

/11.3 The Impact ...

11.3 The Impact of Computers on the Mathematics Curriculum

Numerous possible mathematics curriculum changes occasioned by the availability and use of computers were identified in the previous chapter. In summary, these potential changes involve a reduction in emphasis on mechanistic manipulation of expressions in favour of a strong emphasis on *real* investigations and problem solving supported by the computer. The syllabus and the activities surrounding it would have to change substantially to accommodate and take full advantage of the computer as a mathematical problem solving tool.

The computer has a potential role to play in the enhancement of schooling in algebra, geometry and calculus. There is a need, however, for 'hard' research, particularly in this country, to assess the viability of integrating the computer into the mathematics curriculum. Extensive trials and tests involving the use of the computer in the mathematics classroom and laboratory will have to be carried out. A recommendation arising from the present study is that substantial research effort be mobilized in order to establish a firm foundation upon which computer inspired mathematics curriculum innovations can be built.

It should be noted that in contrast to the possible beneficial effects of the computer in the mathematics curriculum, there is the danger that it can play a detrimental role. An example might be the rigid algorithmic, right-to-left vertical addition of 2- and 3-digit numbers demanded by the *TOAM* and *SERGO* systems. Large sums of money have been and are being spent in this country on installations of these systems, and these investments could inhibit curriculum development. Any program requiring the pupil to follow a predetermined approach is potentially anti-constructivist and as such unsuitable in the light of the current trend emphasizing freedom of computational and problem solving approach.

/11.4 Some Questions ...

11.4 Some Questions Suggested for Investigation

Several questions arising from this thesis are suggested for investigation. Inter-alia:

- Will the teacher's personal accountability (measured in, say, examination results) be lowered if (s)he becomes merely a partner, with the microcomputer, in the teaching process?

- Will not this 'high-tech' approach accelerate the widening of the gap between the first world and third world components of the educational system in our country?

- Will the possible advantages of using microcomputers (say in avoiding excessive manipulation and in problem solving) save enough time to avoid displacing mathematical content to provide time for programming activities?

- Should the teaching world use its scarce supply of mathematics teachers to teach programming - and should we use limited mathematics teaching time?

- How can we be sure that computer problem solving activities will transfer to other cognitive areas?

- Finally, and perhaps most importantly, a concern shared by Noble (1990), in the words of Howson and Kahane (eds.) (1986b):

"If the calculator removes the necessity for acquiring and perfecting arithmetic skills, and the micro those associated with polynomial algebra and the elementary calculus, where - and when - is the [would be] professional mathematician to gain skills which (s)he will need?"

Research into these and other, related, questions could serve to provide valuable additional information in the field of knowledge to which this project has attempted to contribute.

11.5 A Call for In-Service Training for Teachers, in Computers in Mathematics Education

It would seem appropriate to end this treatise by calling for the establishment of in-service training courses for teachers, in the field of computer applications in mathematics education. The power of the computer as an agent for the enhancement of secondary school mathematics education will not be harnessed and exploited until a sufficient number of teachers have developed the necessary awareness and expertise. It is recommended that in-service training courses be instituted as and where possible to meet this need. Teachers of mathematics should be shown that there is more to computer-aided mathematics than drill-and-practice and traditional CAI. They should be introduced to other applications such as 'intelligent' tutoring, computer guided teaching, games, simulations, pre-programmed computational environments, and perhaps even elementary programming.

REFERENCES

Abelson, H. and Di Sessa, A.A. (1980). Turtle Geometry: the computer as a medium for exploring mathematics. The M.I.T. Press, Cambridge, Massachusetts.

Appel, K. and Haken, W. (1976). The solution of the 4-colour map problem. Sci. American (Oct.), 108-121.

Appel, M. and Appel, S. (1987). A critique of CAI: the case of SERGO. South African Journal of Education, 7(4).

Arganbright, D.E. (1984). Mathematical applications of an electronic spreadsheet. Proposal for 1984 NCTM Yearbook

Baldwin, D. (undated). Sequences Set. Chalksoft Ltd. (address unknown).

Bana, C., et al. (1981). In Lewis, R. and Tagg, D. (eds.) (1981). Using computer science in order to teach mathematics. Computers in Education. North-Holland Publishing Company.

Bell and Hyman. Denmark House, 37-39 Queen Elizabeth Street, London SE1 2QB.

Blease, D. (1986). Evaluating educational software. Croom Helm, London.

Brown, J.S. (1979). Fundamental research in technology in science education. Technology in Science Education: The next ten years.

Brown, I. (1984). Looking for quality in software. Educational Computing (Jan.).

Brown, J.S. and Burton, R.R. (1978). Diagnostic models for procedural bugs in basic mathematical skaills. As referenced by Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent Tutoring Systems. Academic Press, London.

Bundy, A. (1983). The computer modelling of mathematical reasoning. Academic Press, London.

Bundy, A. and Welham, R. (1984). Using meta-level inference for selective application of multiple rewrite rules in algebraic manipulation. Proceedings of the Fifth Conference on Automated Deduction. Lecture Notes in Computer Science, Springer-Verlag.

Burrell, D. (1986). Computers in teaching mathematics. Department of Computer Science, Rhodes University, Grahamstown.

Burton, R.R. and Brown, J.S. (1976). A tutoring and student modelling paradigm for gaming environments. As referenced by Sleeman, D. and Brown J.S. (eds.) (1982). Intelligent Tutoring Systems. Academic Press, London.

CAMBRIDGE MICROSOFT. Cambridge University Press, Cambridge.

Cape Education Department (CED) (1985). Senior Secondary Course Syllabuses for Mathematics Higher and Standard Grades. Provincial Admin. of the Cape of Good Hope, Wale Str., Cape Town.

Coburn, P. et al. (1982). Computers in Education. Addison-Wesley.

Cohen, L. and Manion, L. (1986). Research Methods in Education (Second Edition, Reprinted). Croom Helm, London.

Collett, P. (1985). Kingswood College, Grahamstown.

CONDUIT. Computing Ideas for Higher Education. The University of Iowa, Iowa City.

Conference Board of the Mathematical Science Committee on Computer Education. (1972). Recommendations regarding computers in high school education (Washington, DC).

Coupland, J. (1984). In Phillips, R. (ed.) (1984). See ITMA.

Croft, G. and Evans, S. (1985). Educational software review project. Computer Education (Feb.).

Department of Education and Culture (DEC): House of Assembly. (1989). Working Document for Mathematics, Standards 8, 9 and 10, Standard and Higher Grades.

Department of Education and Science (DES). (1985). Mathematics from 5 to 16. HMSO, London.

de Villiers, M. D. (1989). From 'TO POLY' to generalized poly-figures and their classifications: a learning experience. Int. J. Math. Educ. Sci. Technol., Vol. 20, No. 4, 585-603.

Dromey, R.G. (1982). How to solve it by computer. Prentice-Hall International Series in Computer Science, Englewood Cliffs.

Dugdale, S. (1982). Green Globs: A microcomputer application for graphing of equations. Mathematics Teacher, Vol. 75, No. 3, NCTM, USA.

Dugdale, S. and Kibbey, D. (1983). Graphing equations. CONDUIT, Iowa City.

Dwyer, T.A. (1974). International Journal of Man-Machine Studies, Vol. 6, p.137.

Dwyer, T.A. (1975). Soloworks: computer-based laboratories for high school mathematics. School Science and Mathematics (Jan.), 75, 1, 93-99.

Eisner, E.W. (1985). The educational imagination: on the design and evaluation of school programs (second edition). Macmillan Publishing Co.

Ellingham, D. and Haydon, R. (undated). Primary education and computers: evaluation of microcomputer programs for primary schools. Micro-Electronics Programme, United Kingdom.

Elliott, P.C. (1976). Programming - an integral part of an elementary mathematics method course. International Journal of Mathematics Education, Science and Technology, Vol. 7, No. 4, pp 447-454.

Engelman, C. (1968). MATHLAB 68. Proceedings of the IFIP (International Federation for Information Processing) Congress 68, Software 1, North-Holland Publ. Co., Amsterdam, B91-95.

Ennals, J.R. (1984). In Yazdani, M. (ed.) (1984). New horizons in educational computing. Ellis Horwood Series in Artificial Intelligence, Ellis Horwood Ltd., Chichester, England.

Feurzeig, W., Papert, S., et al. (1969). Programming languages as a conceptual framework for teaching mathematics. Rep. No. 1899, Bolt Beranek and Newman Inc., Cambridge, Massachusetts.

Fey, J., and Heid, M.K. In Hansen, V. (ed.) (1984). Computers in mathematical education. NCTM Yearbook.

Fey, J., et al. (1984). Computing and mathematics: the impact on secondary school curricula. Report of a conference sponsored by the National Science Foundation (for the) NCTM, University of Maryland.

Fraser, R., et al. (with Wells, C. and Allnut, M.) (undated). See ITMA.

Genesereth, M.R. (1978). Automated consultation for complex computer systems. Ph.D. unpublished, Harvard University. In Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent tutoring systems. Academic Press, London.

GEOMETRIC SUPPOSER. Sunburst Communications, 39 Washington Ave., Pleasantville, New York.

Ginsburg et al. (1969). Piaget's theory of intellectual development. Prentice-Hall, Englewood Cliffs.

GRAPHIC CALCULUS. See Tall, D. (1986).

Greenslade, T. (undated). See ITMA.

Halpin, M. (1986). Logic for problem solving. Department of Mathematics (Pure and Applied), Rhodes University, Grahamstown. Working Paper.

Halpin, M. and Marsh, T.A. (1988). A computer inspired strategy for teaching equation solving. South African Journal of Education, 8(2).

Hansen, V. (ed.) (1984). Computers in mathematical education. NCTM Yearbook.

Hativa, N. (1984). Computer-Guided Teaching: an effective aid for group instruction. Computers in Education, Vol. 8, No. 3, pp 293-303. Pergamon Press, Great Britain.

Higgo, J. et al. (1985). 132 Short programs for the mathematics classroom. The Mathematical Association and Stanley Thornes (Publishers) Ltd.

Howson, A.G (1985). The impact of computers on mathematics education. The Journal of Mathematical Behaviour, Vol. 4, No. 3.

Howson, A.G. and Kahane, J-P. (eds.) (1986a). The influence of computers and informatics on mathematics and its teaching. ICMI Study Series, Cambridge University Press.

Howson, A.G. and Kahane, J-P. (eds.) (1986b). School mathematics in the 1990's. ICMI Study Series, Cambridge University Press.

Human, P. and Joubert, C. (undated). See NASOU.

ITMA. Investigations on Teaching with Microcomputers as an Aid. ITMA Collaboration, College of St. Mark and St. John, Derriford Road, Plymouth, Devon, England.

Ives, B. (undated). See ITMA.

Jones, K.L. and Lamb, C.E. (1985). Programming in mathematics education: an essential component in the development of reasoning skills. In Duncan, K. and Harris, D. (eds.) (1985). Computers in Education. Elsevier Science Publishers, B.V. (North-Holland), Amsterdam.

Kansky, R.J. (1982). The many faces of instructional computing. Computers in Education. Proceedings of the South African Congress on Computers in Education, Stellenbosch University.

Kelman, P., et al. (1983). Computers in teaching mathematics. Addison-Wesley.

Kemmis, S., Atkin, R. and Wright, E. (1977). How do students learn? Working papers on computer assisted learning. Occasional paper no. 5, Centre for Applied Research in Education, University of East Anglia.

Kimball, R.B. (1981). A self-improving tutor for symbolic integration. In Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent Tutoring Systems. Academic Press, London.

Kite, D. and Forecast, C. (1983). Maths Topics 1. CAMBRIDGE MICROSOFT, Cambridge University Press, Cambridge.

Kowalski, R. (1984). In Yazdani, M. (ed.) (1984). New horizons in educational computing. Ellis Horwood Series in Artificial Intelligence, Ellis Horwood Ltd.

Kurtz, T.E. (1970). The computer as pupil: the Dartmouth Secondary School Project. Dartmouth College, Hanover NH. Final report to the NSF on Grant GW-2246.

Lander, L.J. and Parkin, T.R. (1967). A counter-example to Euler's sum of powers conjecture. Math. Comp., 21, 101-103.

Lawler, R. (1985). Computer experience and cognitive development. Ellis Horwood.

Luehrmann, A. (1972). ACM SIQCUE Bulletin 6, 10.

MacDonald, C., Smith, P. and Metrowich, T.P. (1986). The attitude of Soweto primary school pupils towards mathematics and TOAM-CAI. Discussion Paper Series in Adult and Continuing Education, University of the Witwatersrand. Discussion Paper No. 12, February 1986.

MacDonald, C. (1986). The response of C.A.A.R.P. teachers to selected aspects of the TOAM innovation: 1984. Discussion Paper Series in Adult and Continuing Education, University of the Witwatersrand. Discussion Paper No. 14, February 1986.

Maddison, A. (1982). Microcomputers in the classroom. Hodder and Stoughton, London.

Marsh, T.A. (1985a). Computer aided mathematical experimentation and problem solving. Proceedings of the Computers in Education and Training Conference, Human Sciences Research Council, Pretoria.

Marsh, T.A. (1985b). Computer algebra in the mathematics classroom. South African Journal of Education, 5.

Marsh T.A. (1986). Exploring advanced school mathematics with Logo. Proceedings of the International Conference: To Educate the Human Potential, Johannesburg College of Education.

Marsh T.A. and Halpin, M.N. (1986a). Mathematical problem solving with Logo: an introductory workshop. Proceedings of the 8th National Congress on Mathematics Education, Stellenbosch University.

Marsh, T.A. and Halpin, M.N. (1986b). An intelligent computer tutoring system for rational function curve sketching. South African Journal of Education, 6(4).

Marsh, T.A. (1988). Computer programs as tools for mathematical problem solving. Proceedings of the 9th National Congress on Mathematics Education, University of the Orange Free State, Bloemfontein.

Marsh, T.A. and Marsh, C.J.A. (1989). A report on an evaluation of the SERGO CAI system for mathematics. Education Department, Rhodes University, Grahamstown.

Marsh, T.A. (1989a). Microcomputer applications in mathematics education. Pythagoras (Journal of the Mathematical Association of Southern Africa), No. 19.

Marsh, T.A. (1989b). The evaluation of computer software for use in mathematics education. Proceedings of the Thirteenth National Convention of Teachers of Mathematics, Physical Science and Biology of South Africa. Pretoria Technikon, Pretoria.

Marsh, T.A. (1989c). The microcomputer in secondary school mathematics: a classification of possible applications, with examples. South African Journal of Education, 9(4). (To appear).

Marsh, T.A. and Schafer, M. (1989). "Green Globs" in the classroom - a case study. Pythagoras (Journal of the Mathematical Association of Southern Africa), No. 20.

MATHCAD. Mathsoft Inc., Kendall Square, Cambridge, Massachusetts.

microPLATO. Control Data South Africa, Sandton, Johannesburg.

Milliken Computer Courseware. (Address unknown).

Miller, A.R. (1984). TK!Solver (a software review). Byte Magazine, December 1984, p.263.

Milner, S. (1974). The Annual Meeting of the American Educational Research Association, Chicago, Illinois.

Minsky, M. (1970). Comput. Mach., 17, 197.

Mullan, T. (1982). Microscope (March), 36-37.

muMATH/muSIMP-83. The Soft Warehouse, Honolulu, Hawaii.

Murakami, H. and Hata, M. (1985). The progess of computers and mathematical education in Japan. IREM, Universite Louis Pasteur, Strasbourg.

Murakami, H. and Hata, M. (1986). Mathematical education in the computer age. In Howson, A.G. and Kahane, J-P. (eds.) (1986). The influence of computers and informatics on mathematics and its teaching. ICMI Study Series, Cambridge University Press.

NASOU. Nasionale Opvoedkundige Uitgewery Beperk, Box 5197, Cape Town.

National Curriculum Council (1988). Mathematics for ages 5 to 16: proposals of the Secretary of State for Education and Science and the Secretary of State for Wales. Department of Education and Science and the Welsh Office.

NCTM. National Council of Teachers of Mathematics. Association Drive, Reston, Virginia.

Noble, A. (1990). The Power of School Mathematics. Institute for Mathematics and Science Teaching of the University of Stellenbosch, Stellenbosch.

Noss, R. (1986). Constructing a conceptual framework for elementary algebra through Logo programming. Educational Studies in Mathematics, Vol. 17, pp 335-357. Reidel Publishing Company.

Oberem, G. (1983). Intelligent tutoring - an overview. The Second Symposium on Computer Science Theory and Practice, Rhodes University, Grahamstown.

Open University (1984). Micros in schools: educational software. The Open University Press, Milton Keynes, UK.

O'Shea, T. (1981). A self-improving quadratic tutor. In Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent Tutoring Systems. Academic Press, London.

Pair, C. (1979). La Construction des programmes. Rairo Informatique, 13, 2, 113-118.

Papert, S. (1972). International Journal of Mathematics Education, Science and Technology, 3, 249.

Papert, S. (1973). New Educational Technology, 01, 1.

Papert, S. (1975). Teaching children. Journal of Structural Learning, 4, 219-229.

Papert, S. (1980). Mindstorms: Children, Computers and Powerful Ideas. Basic Books, New York.

Parlett, M. and Hamilton, D. In Tawney, D. (ed.) (1976). Curriculum evaluation today: trends and implications - a school council research study, 84-101. Macmillan.

Pavelle, R., Rothstein, M. and Fitch, J. (1981). Computer algebra. Sci. American, (Dec.) 245(6).
PC PLANNER. (1986). Sagesoft, Regent Centre, Gosforth, Newcastle upon Tyne, England.

Perkins, M. and Perkins, P. (1987). Advanced Mathematical Software for Exploring Functions and Graphs. Bell and Hyman, Denmark House, 37-39 Queen Elizabeth Street, London SE1 2QB.

Personal Computing Today (May 1984). Phillips Publishing Inc., Montrose Rd., Potomac MD.

Phillips, I.E. (1989). Spreadsheets and maths? The sum to infinity of a geometric series. Department of Mathematics, University of the Witwatersrand. Working Paper.

Phillips, R. (undated). See ITMA.

Phillips, R. (ed.) (1984). See ITMA.

PLATO. Control Data Corporation. P.O. Box 0, Minneapolis, Minnesota, USA.

Polya, G. (1973). How to solve it. Princeton University Press, Princeton, NJ.

Primary Teaching and Micros (January 1984). Scholastic Publications, Leamington Spa, Warwickshire, GB.

Purchase, H. (1986). A natural language interface to a geometry tutoring system. Department of Computer Science, Rhodes University, Grahamstown.

QUEST. Robin Stead Associates, Claremont, Cape Town.

Riordan, D. (1987). Introduction to Logo. Technical Document 87-2, Department of Computer Science, Rhodes University, Grahamstown.

Ross, P. and Howe, J.A.M. (1981). Teaching mathematics through programming: ten years on. In Lewis, R. and Tagg, D. (eds.). Computers in Education. North-Holland, Amsterdam.

SERGO. Sentrum vir Rekenaargesteunde Onderrig (Centre for Computer Assisted Instruction) (Pty.) Ltd., 1077 Arcadia Str., Hatfield, Pretoria.

Shilgalis, T. (1982). As referenced by Fey, J. and Heid M.K. In Hansen, V. (ed.) (1984). Computers in mathematical education. NCTM Yearbook.

Sicks, J.L. (1985). Investigating secondary mathematics with computers. Prentice-Hall, Inc.

Slaughter, A. (1989). A cataloguing system for microcomputer diskettes, in particular for those used for mathematics computer aided learning. Education Department, Rhodes University, Grahamstown.

Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent tutoring systems. Academic Press, London.

Sleeman, D. and Smith, M.J. (1981). In Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent tutoring systems. Academic Press, London.

SLIMWAM2. (undated). Some (more) lessons in mathematics with a microcomputer. The Association of Teachers of Mathematics, King's Chambers, Queen Str., Derby, England.

SMILE (1985). The Next 17 (Microsmile 2). Centre for learning resources, 257 Kennington Lane, London SE11 5QZ.

Smith, C. (1990). Calculus and computers. Proceedings of the 10th National Congress on Mathematics Education. Durban. The Mathematical Association of Southern Africa.

Stanfield, J., Carr, B. and Goldstein I. (1976). In Sleeman, D. and Brown, J.S. (eds.) (1982). Intelligent tutoring systems. Academic Press, London.

Steen, L.A. (1981). Computer Calculus. Science News (Apr.), 119, 16, 250-1.

Stoutemyer, D. and Rich, A. (1983). The muMATH/muSIMP reference manual. The Soft Warehouse, Honolulu, Hawaii.

Straker, A. (1982). A check-list for teachers. Microscope 6.

SUPERGRAPH. See Tall, D. (1985).

SuperPILOT. (1981). Apple Computer Inc., Bandley Drive, Cupertino, California.

Tall, D. (1985a). Supergraph. Glentop Publishers Ltd., London.

Tall, D. (1985b). Understanding the calculus. Mathematics Teaching (March), 49-53. The Association of Teachers of Mathematics, King's Chambers, Queen Str., Derby, England.

Tall, D. (1985c). Visualizing calculus concepts using a computer. IREM, Universite Louis Pasteur, Strasbourg.

Tawney, D. (ed.) (1976). Curriculum evaluation today: trends and implications: a second collection of papers from members of the Schools' Council project evaluators' group on aspects of their work.

Taylor, R.P. (ed.) (1980). The computer in the classroom: tutor, tool, tutee. New York: Teachers' College Press.

TenCore (copyright 1987, 1988). Computer Teaching Corporation, 1713 South Neil Street, Champaign, Illinois, USA.

Thompson, P. (1982a, 1982b). As referenced by Fey, J. and Heid, M.K. In Hansen, V. (ed.) (1984). Computers in mathematical education. NCTM Yearbook.

TK!Solver. See Miller (1984).

TOAM. Centre for Educational Technology, Ramat Aviv, Israel.

Usiskin, Z. (1980). As referenced by Fey, J. and Heid, M.K. In Hansen, V. (ed.) (1984). Computers in mathematical education. NCTM Yearbook.

van Hille, G.E. (1986). A preliminary investigation into the use of computers in the teaching of mathematics. M.Ed. unpublished, Rhodes University, Grahamstown.

von Ludwig, W. A. P. (1986). Die ontwikkeling van die bekwaamheid om meetkundige figure verbaal te beskryf. Unpublished Ph.D. thesis, Stellenbosch University.

Wain, G.T. and Flower, S.M. (1989). Mathematics homework on a micro. Leeds University.

WAMSOFT. (1984). WAMSOFT Productions Development Centre, WACAE, Perth, Australia.

WCCE70. (1970). van der Aa, H.J. (ed.). Computers and Education; an international bibliography on computer education. A special publication on occasion of the IFIP (International Federation for Information Processing) World Conference on Computer Education, prepared by the editors of New Literature on Automation, Netherlands Centre for Informatics, Amsterdam. Science Associates/International, New York.

WCCE75. (1975). Lecarme. O. and Lewis, R. (eds.). Proceedings of the 2nd World Conference on Computers in Education, IFIP (International Federation for Information Processing) TC-3, North-Holland Publ. Co., Amsterdam.

WCCE81. (1981). Lewis, R. and Tagg, D. (eds.) Computers in Education. Proceedings of the IFIP (International Federation for Information Processing) TC-3 3rd World Conference on Computers in Education, Lausanne, Switzerland. North-Holland Publishing Co., Amsterdam.

WCCE85. (1985). Duncan, K. and Harris, D. (eds.) Computers in Education. Proceedings of the IFIP (International Federation for Information Processing) TC-3 4th World Conference on Computers in Education, Iowa City, USA. North-Holland Publishing Co., Amsterdam.

Wigley, A. (undated). See ITMA.

Wilf, H.S. (1982). The disk with the college eduaction. Amer. Math. Mthly. (Jan.), 89, 1, 4-8.

Winter, M.J. (1985). Using computers with undergraduate mathematics students in college algebra, elementary calculus and teacher-training courses. IREM, Universite Louis Pasteur, Strasbourg.

Yazdani, M. (ed.) (1984). New horizons in educational computing. Ellis Horwood Series in Artificial Intelligence, Ellis Horwood Ltd.

APPENDICES

APPENDIX A1

Computer Software for Mathematics Education

Evaluation Form for Teachers

0. Name of Program being Evaluated:

1. Evaluator's Particulars

Name:

School:

Position:

Academic & Professional Qualifications:

Years of Teaching Experience:

2. Software Objectives

In your opinion does this program enable the criteria listed below to be met? Please rate the extent to which each criterion is met on the following four point scale:

1 indicates "not at all"
2 indicates "slightly"
3 indicates "moderately"
4 indicates "very much so"

(Give the rating of a criterion as N/A if you feel that it is not applicable in the context of this particular program.)

	Criterion	Rating
2.1	Develops problem-solving skills	....
2.2	Develops investigatory ability	....
2.3	Encourages use of appropriate computation skills	....
2.4	Encourages pupil to make predictions	....
2.5	Enables testing of predictions	....
2.6	Promotes awareness of inter-relationships of mathematical concepts	....

2.7	Communicates effectively in words	....
2.8	Communicates effectively in symbols	....
2.9	Communicates effectively in diagrams	....
2.10	Develops a positive attitude to mathematics as an interesting and attractive subject	....

2.11 What improvements can you suggest?

3. Technical Design

In your opinion does this program meet the following criteria? Please rate the extent to which each criterion is met on the aforementioned four-point scale.

	Criterion	Rating
3.1	Instructions to the user are clear	....
3.2	The user has good control over the execution of the program	....
3.3	Progress through the program is logical and smooth	....
3.4	Graphics are used effectively	....
3.5	The user has freedom to explore mathematical concepts at various stages	....
3.6	Any 'help' information provided by the program is relevant and useful	....

3.7 What improvements can you suggest?

4. Subject Content

Please comment verbally on each of the following points as they relate to the subject content of the software:

4.1 Degree of accuracy

4.2 Underlying mathematical concepts

4.3 Degree of motivation for the pupil

4.4 Applicability to the present syllabus

4.5 Applicability to other school subject areas

4.6 What improvements can you suggest?

5. Documentation for Teachers

Please comment verbally on the instruction manual, if there is one, with regard to the following points:

5.1 Clarity

5.2 Ease of reference

5.3 Usefulness

5.4 Accuracy

5.5 What improvements can you suggest?

6. Conclusions

Please comment on the viability of using this software as a teaching/learning tool:

6.1 Content

6.2 Usability

6.3 General

Thank you for your cooperation!

APPENDIX A2

Computer Software for Mathematics Education

Evaluation Form for Pupils

0. Name of Program being Evaluated:

1. Pupil's Details

Name:

School:

Standard:

Please say whether you are a boy or a girl:

Please answer the following questions completely honestly:

2. Computer Familiarity

2.1 Have you used a computer before?

2.2 Have you played computer games?

2.3 Have you tried Logo?

2.4 Have you tried computer aided maths. learning before?

If so, please give details:

3. Attitude to Using Computers

3.1 Do you like working on a computer?

3.2 Would you like to do more computer aided maths.?

Please explain why?

4. Attitude to Mathematics

4.1 Do you normally do well in maths?

4.2 Do you like maths?

5. The Session

5.1 Did you enjoy the computer aided maths. session?

5.2 Did the person supervising the session explain properly how the computer program should be used?

5.3 Did you understand what you had to do?

5.4 Did you have to ask for help at all during the session?

5.5 Did the work with the computer help you to understand your maths. better?

6. General Information

6.1 What did you like most about the session?

6.2 What did you not like about the session?

6.3 Any other comments?

Thank you very much for your help!

APPENDIX A3

Teacher Evaluation of Software for use in Mathematics Education

Name:

Years of Experience Teaching Mathematics:

Main Level of this Experience:

Do you have an interest in possible application of computers in mathematics education?:

What is your opinion of the potential role of computers in mathematics education?:

Please describe any experience you have in this area:

Name of Program being Evaluated:

How would you classify the program? (tick appropriate category(ies)):

a teaching program (attempts to introduce new concepts)
a drill-and-practice program
a game
a simulation
other (name category)

Do you see any potential use for this program?:

Explain your answer:

Which aspect(s) of the syllabus does the program involve, if any?:

Given the necessary facilities (a computer or computers and the software in your classroom) would you use this program in your teaching?:

Explain your answer:

Can the teaching of the concepts involved be done just as well without the aid of a computer?:

Explain your answer:

Does this program lead you to think of other (non-computer) strategies for teaching the same or similar material?:

Explain your answer:

Any other comments:

Thank you for your help!

APPENDIX A4

SERGO System Evaluation Form for Teachers

1. Evaluator's Particulars

School:

Position:

Academic/Professional Qualifications:

Years of Teaching Experience:

In what context did you use the SERGO system?

2. Software Objectives

What do you understand the objectives of the SERGO system to be?

In your opinion does the SERGO software meet the following criteria? Please rate the extent to which each criterion is met on the following four point scale:

1 indicates	"not at all"
2 indicates	"slightly"
3 indicates	"moderately"
4 indicates	"very much so"

	Criterion	Rating
1.	Develops problem-solving skills	....
2.	Develops investigatory ability	....

3.	Encourages use of appropriate computational skills	....
4.	Encourages pupil to make predictions	....
5.	Tests predictions	....
6.	Encourages approximation and estimation	....
7.	Encourages awareness of inter-relationships of mathematical concepts	....
8.	Applies mathematics to everyday situations	....
9.	Communicates effectively in words	....
10.	Communicates effectively in symbols	....
11.	Communicates effectively in diagrams	....
12.	Develops a positive attitude to mathematics as an interesting and attractive subject	....

13. What improvements, if any, can you suggest?

3. Technical Design

In your opinion does the SERGO system meet the following criteria? Please rate the extent to which each criterion is met on the aforementioned four-point scale.

	Criterion	Rating
1.	The teacher is able to preview the lessons with ease and efficiency	....
2.	Instructions to the pupils are clear	....
3.	Instructions to the teacher are clear	....
4.	Progress through a lesson is logical and smooth	....

5.	Graphics are used effectively	....
6.	The teacher has good control of how the program is executed by the pupil	....
7.	The pupil has freedom to explore mathematical concepts at various stages	....
8.	The 'help' information is relevant and useful	....
9.	The teacher receives all relevant information about the pupil's progress	....
10.	Pupils' records are easy to obtain and to manage	....

11. What improvements, if any, can you suggest?

4. Subject Content

Please comment verbally on each of the following points as they relate to the subject content of the SERGO lessons:

4.1 Degree of accuracy

4.2 Underlying mathematical concepts

4.3 Degree of motivation for the pupil

4.4 Applicability to the present syllabus

4.5 Applicability to other school subject areas

4.6 What improvements, if any, can you suggest?

5. Documentation for Teachers

Please comment verbally on the SERGO instruction manual with regard to the following points:

5.1 Clarity

5.2 Ease of reference

5.3 Usefulness

5.4 Accuracy

5.5 What improvements, if any, can you suggest?

6. Final Conclusions

Please feel free to make any comments or suggestions relating to the SERGO system with regard to the following points:

6.1 Content

6.2 Usability

6.3 General

Thank you very much for your cooperation - it is greatly appreciated!

APPENDIX A5

SERGO System Evaluation Form for Pupils

1. Pupil's details

School:

Standard:

Are you a boy or a girl?

Please answer the following questions completely honestly:

1. Computer familiarity

1.1 Have you used a computer before?

1.2 Have you played computer games?

1.3 Have you tried Logo?

1.4 Have you used computer lessons before?

2. Attitude to using computers

2.1 Do you like working on a computer?

2.2 Would you like more computer lessons?

3. Attitude to mathematics

3.1 Do you do well in mathematics?

3.2 Do you like mathematics?

4. The lesson

4.1 Did you enjoy the computer lesson?

4.2 Did your teacher explain how to use the computer lesson properly?

4.3 Did you have to call your teacher to help you with something?

4.4 Did the computer lesson keep you busy all the time?

4.5 Did you understand what you had to do?

4.6 Did you discuss the computer lesson with your friends?

4.7 Did you need to use pen and paper or a calculator while you were doing the computer lesson?

5. General Information

5.1 What did you like most about the lesson?

5.2 What did you not like about the lesson?

Thank you very much for your help!

APPENDIX A6

Kingswood College Curve Sketching Workshop

Instructions for use curve sketching program

Switch computer on

Wait for instruction: "Insert system disk"

Insert disk into drive A

Press ENTER

Press CAPSLOCK

Type LOAD(CURVESK); (N.B. semi-colon)

Press ENTER

Type SKETCH(); (N.B. semi-colon)

Press ENTER

Follow instructions given by program

Worksheet

Curve Sketching Problems
From "Classroom Mathematics Std. 10" by Laridon et al.

p. 210, Example 2

$$y = x^3 - 4x^2 + 4x$$

p. 211, Example 3

$$y = -(x + 1)^3 + 8$$

p. 212, Exercise 6.5

SG * 2.i) $y = 1 - x^3$

* 2.k) $y = x^3 + 3x^2$

p. 213, Exercise 6.5

HG ** 4.c) $y = x + \frac{1}{x}$

** 4.e) $y = \frac{1}{x^2}$

Evaluation Questionnaire

1. Do you take mathematics on the HG or the SG?:
2. Do you find the section on curve sketching easy, OK or difficult?
3. Did you find the curve sketching program easy to use?:

   If not, explain why:

4. Do you think that it could help you to learn curve sketching?:

   If yes, explain why:

   If no, explain why:

5. Would you like to make more use of the computer as a mathematical problem solving aid?:
6. Which other aspects of the syllabus lend themselves to computer exploration?:
7. General comments:

Thank you!

APPENDIX A7

St. Andrew's College Standard 9 Learning Conference

Computer Algebra Workshop

Worksheet

1. Solve simultaneously the following pairs of linear equations:

(a) $-x+y = 1 ; 2x+3y = 3$

(b) $-x+y = 1 ; 2x+3y = 4$

(c) $-x+y = 1 ; 2x+3y = 5$

(d) $-x+y = 1 ; 2x+3y = 6$

What do you notice about the solutions?

Explain the pattern.

Predict the solution to $-x+y = 1 ; 2x+3y = 9$

Check your answer.

Sample muMATH input: LINEQN([-X+Y==1,2X+3Y==3],[X,Y]); (N.B. semi-colon)

2. Solve the following quadratic equations:

(a) $2(x-3/2)^2 - 3 = 0$

(b) $2(x-3/2)^2 - 1 = 0$

(c) $2(x-3/2)^2 = 0$

(d) $2(x-3/2)^2 + 1 = 0$

What do you notice about the solutions?

Explain.

Sample muMATH input: SOLVE(2(X-3/2)^2-3==0,X); (N.B. semi-colon)

3. Find two factors of each of the following expressions by using the Remainder Theorem:

(a) x^3-7x+6

(b) $2x^3+7x^2-7x-12$

(c) x^4+x^2-8x-8

(d) $(x^5/2)-x^2-(x/2)+1$

Sample muMATH input: EVSUB(X^3-7X+6,X,1); (N.B. semi-colon)

4. Predict approximately the number of digits in the expanded form of each of the following factorials and check your answers by muMATH computation:

(a) 10!

(b) 20!

(c) 50!

(d) 100!

How many digits do you think 500! contains?

Check.

Sample muMATH input: 10!; (N.B. semi-colon)

Evaluation Questionnaire

As in Appendix A2

APPENDIX A8

Multi-dimensional Classification of the *SASEC* Programs

Global educational paradigm			
Instructional ✓	Revelatory	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control	Learner has full control

Educational purpose		
Drill-and-practice ✓	Concept building	Problem solving

Nature of information		
Facts ✓	Concepts	Procedures

Mode of Information production		
Retrieval procedures ✓	Object-level procedures	Meta-level procedures

Program action mode		
Requests information ✓	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers ✓	Explains answers

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification ✓
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response ✓	Direct Explanation	Instructional Embodiment	Problem solving	Provocation

Multi-dimensional Classification of the *SERGO* System

Global educational paradigm			
Instructional ✓	Revelatory	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control	Learner has full control

Educational purpose		
Drill-and-practice ✓	Concept building	Problem solving

Nature of information		
Facts ✓	Concepts	Procedures

Mode of Information production		
Retrieval procedures ✓	Object-level procedures	Meta-level procedures

Program action mode		
Requests information ✓	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers ✓	Explains answers

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification ✓
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response ✓	Direct Explanation	Instructional Embodiment	Problem solving	Provocation

Multi-dimensional Classification of the *TOAM* System

Global educational paradigm			
Instructional ✓	Revelatory	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control	Learner has full control

Educational purpose		
Drill-and-practice ✓	Concept building	Problem solving

Nature of information		
Facts ✓	Concepts	Procedures

Mode of Information production		
Retrieval procedures ✓	Object-level procedures	Meta-level procedures

Program action mode		
Requests information ✓	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers ✓	Explains answers

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification ✓
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response ✓	Direct Explanation	Instructional Embodiment	Problem solving	Provocation

Multi-dimensional Classification of the Program *DIRECTED*

Global educational paradigm

Instructional	Revelatory ✓	Conjectural	Emancipatory

Locus of control

Program has full control	Learner has indirect control	Learner has direct control ✓	Learner has full control

Educational purpose

Drill-and-practice	Concept building ✓	Problem solving

Nature of information

Facts ✓	Concepts ✓	Procedures

Mode of Information production

Retrieval procedures	Object-level procedures ✓	Meta-level procedures

Program action mode

Requests information ✓	Provides information ✓	Processes information ✓

Feedback mode

No feedback	Right/wrong	Provides answers ✓	Explains answers ✓

Reward paradigm

Assessment ✓	Intrinsic	Winning			Personification ✓
		Opponent	Computer	Self	

Teaching mode

Stimulus-response	Direct Explanation	Instructional Embodiment ✓	Problem solving	Provocation

Multi-dimensional Classification of the Program *AUTOFRAC*

Global educational paradigm			
Instructional	Revelatory ✓	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control	Learner has direct control ✓	Learner has full control ✓

Educational purpose		
Drill-and-practice ✓	Concept building ✓	Problem solving

Nature of information		
Facts ✓	Concepts ✓	Procedures

Mode of Information production		
Retrieval procedures	Object-level procedures ✓	Meta-level procedures

Program action mode		
Requests information	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong	Provides answers ✓	Explains answers

Reward paradigm					
Assessment	Intrinsic ✓	Winning			Personification
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response	Direct Explanation	Instructional Embodiment	Problem solving	Provocation ✓

Multi-dimensional Classification of the Program *FRAC*

Global educational paradigm			
Instructional ✓	Revelatory	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control ✓	Learner has full control

Educational purpose		
Drill-and-practice ✓	Concept building	Problem solving

Nature of information		
Facts ✓	Concepts ✓	Procedures

Mode of Information production		
Retrieval procedures ✓	Object-level procedures	Meta-level procedures

Program action mode		
Requests information ✓	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers ✓	Explains answers

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response ✓	Direct Explanation	Instructional Embodiment	Problem solving	Provocation

Multi-dimensional Classification of the Program *SKETCHER*

Global educational paradigm			
Instructional	Revelatory	Conjectural ✓	Emancipatory ✓

Locus of control			
Program has full control	Learner has indirect control	Learner has direct control	Learner has full control ✓

Educational purpose		
Drill-and-practice	Concept building ✓	Problem solving ✓

Nature of information		
Facts ✓	Concepts	Procedures ✓

Mode of Information production		
Retrieval procedures	Object-level procedures	Meta-level procedures ✓

Program action mode		
Requests information	Provides information ✓	Processes information ✓

Feedback mode			
No feedback	Right/wrong ✓	Provides answers ✓	Explains answers ✓

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response	Direct Explanation ✓	Instructional Embodiment	Problem solving	Provocation

Multi-dimensional Classification of the Program *EQUATORS*

Global educational paradigm			
Instructional ✓	Revelatory ✓	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control ✓	Learner has full control

Educational purpose		
Drill-and-practice	Concept building ✓	Problem solving

Nature of information		
Facts ✓	Concepts ✓	Procedures ✓

Mode of Information production		
Retrieval procedures	Object-level procedures ✓	Meta-level procedures

Program action mode		
Requests information ✓	Provides information	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers	Explains answers

Reward paradigm					
Assessment	Intrinsic ✓	Winning			Personification
		Opponent	Computer ✓	Self	

Teaching mode				
Stimulus-response	Direct Explanation	Instructional Embodiment ✓	Problem solving	Provocation

Multi-dimensional Classification of the Program *NUMBER MACHINE*

Global educational paradigm			
Instructional ✓	Revelatory ✓	Conjectural	Emancipatory

Locus of control			
Program has full control	Learner has indirect control ✓	Learner has direct control ✓	Learner has full control

Educational purpose		
Drill-and-practice ✓	Concept building ✓	Problem solving

Nature of information		
Facts ✓	Concepts ✓	Procedures

Mode of Information production		
Retrieval procedures	Object-level procedures ✓	Meta-level procedures

Program action mode		
Requests information ✓	Provides information ✓	Processes information

Feedback mode			
No feedback	Right/wrong ✓	Provides answers	Explains answers

Reward paradigm					
Assessment ✓	Intrinsic	Winning			Personification
		Opponent	Computer	Self	

Teaching mode				
Stimulus-response	Direct Explanation	Instructional Embodiment ✓	Problem solving	Provocation

Multi-dimensional Classification of the Program *GREEN GLOBS*

Global educational paradigm			
Instructional	Revelatory	Conjectural ✓	Emancipatory

Locus of control			
Program has full control	Learner has indirect control	Learner has direct control ✓	Learner has full control

Educational purpose		
Drill-and-practice	Concept building ✓	Problem solving

Nature of information		
Facts ✓	Concepts ✓	Procedures ✓

Mode of Information production		
Retrieval procedures	Object-level procedures ✓	Meta-level procedures

Program action mode		
Requests information	Provides information ✓	Processes information ✓

Feedback mode			
No feedback	Right/wrong	Provides answers ✓	Explains answers

Reward paradigm					
Assessment	Intrinsic	Winning			Personification
		Opponent	Computer	Self ✓	

Teaching mode				
Stimulus-response	Direct Explanation	Instructional Embodiment	Problem solving ✓	Provocation

APPENDIX A9

A Cataloguing System for Microcomputer Diskettes Containing Programs for use in Mathematics Education

The system developed entails cataloguing a diskette according to the following attributes, in the order given:

- Dewey decimal classification number for the contents
- the machine type
- the software type
- the level of the user

An example is the code

512.922/10/03/04

which is interpreted as follows:

512	:	Algebra
.9	:	Pedagogical algebra
.922	:	Exponents and logarithms
10	:	BBC
03	:	CAL
04	:	Senior secondary

Note that, in this instance, software type is given as CAL (computer assisted learning). Eventually, as the system becomes more refined, specific categories such as drill-and-practice, game, etc., will be coded into the catalogue number.

Further information required for a formal catalogue entry is given according to conventional cataloguing rules, under the following headings:

1. Title and Statement of Responsibility
2. Edition
3. Publication and Distribution
4. Physical Description
5. Notes
6. Standard Numbers and Terms of Availability

An example of the use of this scheme is shown below:

Catalogue number : 510/10/03/08
(510 is mathematics)

"Area" number	**Information**
1.1	*SLIMWAM 2* (Some Lessons In Mathematics With A Microcomputer)
1.2	Microelectronics in Mathematical Education (MIME) group of the Association of Teachers of Mathematics (ATM)
2.1	
2.2	
3.1	Kings Chambers, Queen Street, Derby, DE1 3DA
3.2	A.T.M.
3.3	April 1985
4.1	1 microcomputer diskette
4.2	colour, sound, two sided, double density, soft sector, hard hole, order code: extra
4.3	133 mm
4.4.1	1 user manual
4.4.2	Dewey number : 371.39445 ISBN : 0900095512 Accession number : 239267
5.1	Computer Aided Learning
5.2	BBC microcomputer, 32K RAM, single diskdrive with 40 track mode, Acorn DFS, BASIC
5.3	English
5.4	
5.5	
5.6	
5.7.1	*MONTY* : investigating number patterns on grids
5.7.2	Sequences
5.7.3	Junior secondary / senior secondary
5.7.4	
5.8	
5.9	
6.1	
6.2	

Table 9-2: A Sample Mathematics Education Diskette Catalogue Entry