COMPUTER ALGEBRA SYSTEMS

AND

SECONDARY EDUCATION

Michael Rickhuss B.A. (Hons).
Department of Computer Science,
The Open University,
Walton Hall,
Milton Keynes MK7 6AA.

A thesis submitted for the Master of Philosophy Degree,
Open University, in June 1993.

Author number: E0089083
Date of submission: 1 October 1993
Date of award: 9 February 1994
PAGE NUMBERING AS ORIGINAL
STATEMENT

I wish to announce that this thesis 'Computer Algebra Systems and Secondary Education' has not been submitted for another degree either with the Open University or with any other university or institution.

M.G.Rickhuss (B.A. Hons) (June 1993)
ABSTRACT

Computer Algebra systems such as MACSYMA, REDUCE, SMP, MAPLE, DERIVE, Mathematica and muMath can be used as a powerful mathematics assistant in the secondary school mathematics classroom. Yet, at the present time, school teachers have declined to experiment with computer algebra systems.

This thesis is divided into three parts. The first part reviews the current literature on the use of computer algebra in education. The second part of this work describes why muMath was selected for the research contained within and how it was adapted and extended with clear pedagogical aims. Part three describes an experimental use of the resulting system, in the classroom, and analysis the results from the experiment.

This thesis contains some new evidence concerning the use of computer algebra systems in the classroom. Additionally a teaching strategy is discussed in section two of the thesis and the results of an experimental system outlined in part three tends to suggest that computer algebra systems will be as familiar to pupils of the future as the calculator is to pupils of today.
Hence, this thesis identifies current research in the use of computer algebra systems in the secondary school. It shows that research is at best poor and consequently proposes the design and implementation of a usable classroom system based on a primary computer algebra system. The resulting implementation, and use of, one such system is then evaluated. The thesis concludes with a personal opinion about the future of computer algebra systems and their uses in education.

Much of this work was completed between 1988 and 1990, and some details may now be somewhat out of date.
ACKNOWLEDGEMENTS

This thesis is the result of three years part time work. It would not have been possible without the assistance and encouragement of Dr. P. G. Thomas at the Open University.

I would also like to thank my wife and daughter for their support even at the difficult times; especially in the early hours of most mornings when the tapping of keys on a keyboard interfered with their sleep.

Lastly I should also thank the pupils of the fourth year mathematics set at Northicote High School, Wolverhampton who acted as guinea pigs for the trailing of my software.

To my parents Mr. & Mrs. O. Rickhuss for their years of support.
CONTENTS

Section 1 — The Literature Review

Chapter 1 Introduction

1 The Structure and Content of the Thesis............. 1
2 Formal Preliminaries.................................... 4
3 Objectives of the Work................................ 5

Chapter 2 Review of the Literature

1 Applications of Algebra Systems in Education........ 11
2 Computer Algebra Systems.............................. 19
3 Characteristics of Computer Algebra Systems ......... 27
4 History of Computer Algebra............................ 31
5 Computer Algebra Systems Compared.................... 33
6 Teaching Systems........................................ 40
Section 2 -- The Resulting Software

Chapter 3 Software Requirements

1 Introduction............................................... 60
2 Pupils View............................................... 64
3 Teachers View........................................... 70

Chapter 4 Description of Resulting Software

1 muMath.................................................... 79
2 Extensions to muMath.................................. 86
3 Additions made for Classroom use.................... 92
4 Mathematical Display Features....................... 99

Chapter 5 How does the software work?

1 Introduction.............................................. 103
2 Modes of Operation.................................... 105
3 Communications between Sub-Systems............... 109

Chapter 6 The Menu System

1 Introduction............................................ 136
2 The Abstract Data Type N-ary Tree.................. 139
3 Implementation Details............................... 157
Section 3 - The use of the System

Chapter 7 The Experiment

1 Design and Implementation.......................... 163
2 Resources.......................................... 165
3 Experiment Results.................................. 168
4 Follow up work..................................... 174

Chapter 8 Conclusions

1 Resulting Implications............................. 176
2 Future Work........................................ 182

References........................................... 186

Appendices

1 Statistics from the Experiment..................... 1
2 Functions Added to muMath............................ 33
3 Teaching Materials.................................. 36
4 ExPress Menu Structure.............................. 37
5 Names of mathematical operations added to muMath.... 40
6 Table of routines and their syntax............... 41

7 Algebra content of the 1990 and 1991 GCSE exams..... 45

8 Batch Files........................................ 48

9 Example of teaching materials for the experiment ... 52

10 Published Paper

An Experiment in the use of Computer Algebra in the Classroom. (1992)
Chapter 1 -- Introduction

1 The Structure and Content of the Thesis

The thesis is divided into three parts. The first part, consisting of this chapter and chapter 2, describes the content of the thesis and reviews the relevant research work in computer algebra and the uses of computer algebra in education. Section 1.2 introduces the notation and definitions that are used in subsequent chapters. Section 1.3 defines the objectives of this research.

The second part of this thesis, consisting of chapters 3 to 6, is concerned with the resulting software designed to meet the objectives of this research. The third part describes a novel use of a modified computer algebra system in the secondary school mathematics classroom and comments on the results obtained.

Chapter 2 begins by examining definitions for computer algebra systems. Section 2.1 describes the component parts of most current computer algebra programs using an approach similar to D. Harper [36]. Section 2.2 traces the developments made in the field starting from 1965. An historical approach has been followed similar to that of J.E. Sammet [71, 72, 73] and has brought the developments up to date. In section 2.3 the features of several computer algebra systems are compared.
from the point of view of their relevance in secondary education. The chapter concludes by looking at the limited amount of research into the uses of computer algebra systems in tertiary education and associated work.

Chapters 3, 4 and 5 detail the analysis, design and implementation of a computer algebra system to meet the classroom needs of pupils and teachers.

Chapter 3 compares the objectives of the work, as outlined in section 1.3, with the characteristics of current computer algebra systems. The conclusions drawn from the comparisons necessitate a proposal for a new system, which we have named ExPress, based upon an existing system (muMath [P2]), with many additional purpose-built sub-systems. In section 3.1 the requirements of the proposed system are examined from the pupils' point of view. In section 3.2 the requirements are examined from the teachers' viewpoint. These two sections give rise to the definitions and functionality of the extra sub-systems required in ExPress.

Chapter 4 considers in detail the character, merits and limitations of the Computer Algebra System, muMath. In section 4.1 a descriptive account of muMath is given from both a mathematical and a technical perspective. Section 4.2 is concerned with extensions made to the basic system to provide extra facilities for use in the secondary school classroom. Section 4.3 investigates the ill-suited aspects of
muMath for use in a classroom and proposes the design of a user-friendly front-end to muMath for ExPress. This section also discusses the relevant merits of Derive [P3], the well known Computer Algebra System, and contrasts its potential classroom uses with those of ExPress. Section 4.4 describes some of the important mathematical and educational features to be embedded in the new system. This chapter is primarily concerned with the design of an experimental system for school use.

Chapter 5 is an explanation of how the new system, ExPress, communicates externally with the user, and internally between its various sub-systems. Section 5.1 deals with the differing levels of user access to ExPress, and section 5.2 deals with the internal system communication. This, the final chapter in section 2, takes the points outlined in the previous two chapters and discusses the implementation details required to produce the system.

Chapter 6 deals with the design and implementation of the front-end for ExPress. The front-end is concerned with user input and output. This is done in two sections; one deals with the design of a Pull Down Menu as an implementation of an abstract data type, see Thomas et al [37], the other deals with the practical implementation details in Turbo Pascal [P4] and the operating system MS-DOS [P1].
Chapter 7 contains a description of the experimental testing of ExPress with a class of 14 and 15 year olds in a Wolverhampton secondary school. Section 7.1 outlines the design and practicalities of the experiment. Section 7.2 looks at the current resources found in school and those required by ExPress. Section 7.3 contains results of the experiment and draws conclusions about such systems. The final section looks to the future by discussing the life expectancy and usefulness of ExPress.

Hence, this thesis identifies current research in the use of computer algebra systems in the secondary school. The thesis shows that research is at best poor and consequentially proposes the design and implementation of a usable classroom system based on a primary computer algebra system. The resulting implementation, and use of, one such system is then evaluated.

2 Formal Preliminaries

This section deals with the abbreviations used throughout the thesis, at this point no attempt is made to define these terms. The definitions or explanations will be given when the terms are first used in the text.

Abbreviations:

CAS    Computer Algebra System
CAL    Computer Aided Learning
A new student of computer algebra (CA) is often caused a fair degree of consternation by the large number of synonymous abbreviations. J. Calmet [12] in the introductory remarks to the EUROCAM 1982 (European Computer Algebra Conference) states that almost any combination of the words algebraic, computation, manipulation and applied mathematics can be used to describe computer algebra. Others have added additional synonyms J.E. Sammet [73] formula manipulation; R. G. Tobey [91] symbolic mathematical computation and E. Ng [59] symbolic and algebraic computation. This is not an exhaustive list but for consistency this thesis uses CA and the definition of CA is based upon those given by E. Ng [59] Yun & Stoutemyer and Loos [108].

'Computer Algebra is the part of computer science in which the design, analysis, implementation and application of algorithms for performing mathematical operation symbolically are studied.'

3 Objectives of the Research

The main objective of the research was essentially very straightforward, and originated from a personal need to
improve the quality of my teaching of algebra. The goal was to investigate the possibility of teaching algebraic concepts as opposed to algebraic manipulations in a secondary school using a computer and a suitable Computer Algebra System (CAS).

The goal of using CAS as a means of teaching algebra to school children required the development of a teaching strategy. The well known method of 'Query by Example' developed by Zloof [110] was selected as the most appropriate and well tested method available. The teaching strategy envisaged was:

(1) input a mathematically question to the CAS and obtain a solution;

(2) present the solution to the pupil together with a selection of possible concepts that might have lead to the solution;

(3) ask the pupil to select the most appropriate concept;

(4) if the pupil is correct offer a reward, otherwise offer remedial help.

Work on CAS in schools is poorly documented. A large volume of work in the use of computer algebra over the last
20 to 30 years concentrates on the design, implementation and applications of CAS (see [13]). It is only recently that the power of CAS as teaching tool has been recognised and then only in a small number of universities where committed individuals have begun to take a wider view of the use of CAS. With their quality and quantity of hardware, university departments were an obvious forum for such teaching experiments, especially when one considers the machine power required to run most algebra systems. It therefore became apparent that this present work needed to examine what systems were currently available and, of these, which could realistically be used in schools.

In 1988, having evaluated several CA packages, the opinion reached was that they all suffered from being designed primarily for research purposes, although more effort is being put into the educational uses of such systems. However, insufficient thought is given to the user interface. This is an obvious disadvantage for school use and therefore demanded that a significant part of the research should be the tailoring of a system to the needs of school children and their teachers.

The literature documenting CAS teaching experiments in school by others was found to be very sparse; only one comparable investigation with school children having been attempted by Apetsberger [3]. Apetsberger's experiment was not exactly what we had in mind but served to highlight how a
CAS could be used in the classroom. Piddock's [66] use of muMath for the teaching and learning of mathematics suggested that a raw computer algebra system (muMath)

'... is an extremely powerful symbolic Algebra tool',
in need of

'... a proper environment for development'.

More importantly, Apetsberger's work demonstrated that the raw muMath system could be fairly easily extended and developed into a usable package. Of all the CAS evaluated, muMath was selected for use with children even though several major problems had to be overcome:

1. The awkward input syntax - use of the word awkward in this sense infers that the syntax used is dissimilar to that normally used in the classroom.

2. The output from muMath is one dimensional, for example

\[ 4 \times x^2 \] as opposed to \( 4x^2 \).

3. No on-line help.

4. No graphics for displaying functions.

5. Some number pattern and numerical routines used in school as an introduction to algebra are not implemented in muMath.

6. muMath provides the user with only the results of calculations and does not describe the intermediate steps that would be necessary in the classroom.
muMath's selection for the research in preference to Derive [P3], muMath's successor, was because muMath could easily be adapted to overcome these limitations whereas even though Derive contained some of these enhancements they were not always seen to be implemented in an appropriate format to meet the needs of school pupils. More seriously, Derive lacks the ability to have its functionality changed as it lacks an internal programming language. This inflexibility in Derive is not present in muMath.

If Derive and muMath are compared, both for mathematical content and user friendliness, it is quickly apparent that Derive is a much improved version of muMath. Derive has an updated user interface and contains graphing facilities. However, it is not tailored for classroom teaching use because it does not attempt to explain solutions. A method of teaching algebraic concepts was felt to be an essential feature of a CAS used in the classroom. These two main omissions from Derive led to the selection of muMath as the base system for ExPress.

Other systems such as MAPLE [P8] and MATHEMATICA [P10], were not considered suitable because they require hardware currently not available in schools. However, it was fully realised that these systems offer a number of the functions that were required in ExPress. SYMBOLATOR [P11], a recent CAS, also has the disadvantage of being fixed in its
3 Objectives of the Research

functionality with no possibility of expansion or enhancements.

It was envisaged that even though muMath contained all the mathematical knowledge we wished to use with pupils, there would be a need to make use of the knowledge in an educationally acceptable and practical way. Allowing children aimlessly to enter questions into a CAS and receive an answer did not seem to be the most appropriate method of teaching algebra. Hence a secondary objective of searching for a suitable teaching strategy emerged.

Therefore, what had started as a single aim of using CAS in school to assist with the teaching and learning of algebra was transformed into two key issues:

1. Embedding a teaching strategy into the developed system;
2. Developing and enhancing an existing CAS for use in schools;

with the subsidiary aim of

Chapter 2 -- Literature Review

As this work contains two main goals it seems sensible to present the literature review in two broad categories even though it is fully realised that they are inevitably interlinked. So this section will deal with the literature on:

(a) applications of algebra systems in education.
(b) computer algebra systems.

This division also exists in the literature since most papers either report on the design of faster and more concise algorithms or on the uses and application of the popular computer algebra systems.

1 Applications of Algebra Systems in Education

Several examples can be found where other researchers have used a raw (unmodified) CAS for teaching in higher education [11,14,18,28,39,44,47,48,60,69,101] but these are of a general interest and not wholly appropriate to the mathematical level of the average school child. However, it is felt that several points need discussion and some will be commented upon before proceeding.

Although the paper by M.Rodino et al [69] is an early report of the use of a microcomputer to aid the mathematics
learning process, the work does indicate the way forward. The comment

'... use of the microcomputer in basic mathematics courses is particularly convenient whenever a visualisation is possible ....'

needs to be considered with care as a well prepared, adequately equipped lecturer or a suitably made television program can be equally convenient and as instructional. Microcomputers are a powerful tool but cannot yet be seen as a panacea for the lack of good mathematics teaching or as a replacement for other forms of communication. The lack of intelligent thought and the inability to respond to a vast number of differing situations presented by the learner precludes computers, at the moment, fully replacing a human teacher.

Among the very important comments about the use of CAS for the teaching of mathematics in Cromer's [18] paper can be found what might be considered an odd, incongruous phrase

'... short [introductory muMath] lectures are independent of the regular courses and are intended to help people begin to learn about muMath ...'

The need to learn 'about muMath' seems unnecessary for a mathematics student; just as he/she has no need to learn how
a calculator functions internally there is no need to learn the internal workings of a computer algebra system; it should simply be viewed as a tool. Also, the method employed by Cromer, of teaching mathematics by extending the number of interactive lessons, was considered to be the wrong approach because, yet again, the method would appear to be the teaching of a computer system rather than the mathematics.

Hosack et al [39] asserts, as does F.Cromer [18], that a main advantage of CAS for pupils is the relief from the boredom of the continual pursuit for solutions to exercises through the learning of set algorithms. Time for the teaching of mathematical concepts is quoted by both authors as a significant reason for the use of CAS and we would candidly agree. All too often, unfortunately, work in the teaching environment with the aid of CAS still relies on achieving solutions to exercises. We consider that the lesser function is the use of the CAS to perform all the manipulations to solve the exercise, and the major function to be that of teaching the mathematical concepts involved in the solution. The Hosack et al paper [39] does, nevertheless, raise four very interesting questions:

1. Do the opportunities presented by CAS require a significant modification of existing mathematics curricula?
2. How will the widespread use of CAS affect the beneficial aspects of hand computations?

3. How can the use of CAS be used to retain the benefits of hand computation?

4. How should mathematics exercises change?

Point 1 could imply that the current mathematics curricula are inadequate and should be altered to incorporate CAS. Development of CAS as a tool of current curricula and teaching methods would be a better cause. This is not to say that curricula should be static; the evolution of curricula is governed by both tradition and the needs of society. Therefore, when computer algebra is widely used by a diversity of professions, society at large will demand its inclusion into the curricula. Hence, we should be publicising the uses of CAS and their benefits so that society can assist and press for the rapid integration of CAS into all mathematics syllabi, and without such publication the time scale for CAS to be included as an integral part of the syllabus will be further elongated. If society is taken to be the professional mathematicians and physicists then this is already happening as a number of university mathematics departments have CAS as integral part of their current undergraduate programmes. For example the Open University, as a major university with a wide audience, is contemplating using CAS in their Mathematics Foundation Course.
Another interpretation of point 1 could be whether there is a need to teach part of the current syllabus because the inclusion of CAS into the programmes of study will automatically modify existing mathematics curricula. Also the syllabus can be viewed as being changed for the better by enabling more 'realistic' mathematics to be taught since CAS could be employed to deal with the routine mundane algebra.

Points 2 and 3 are equally valid now as they were in 1984 and little or no evidence exists which can aid in the conclusive acceptance or categorical rejection of the sentiments expressed. Only the very recent evidence of Palmiter [63] begins to address these two issues. Point 4 stresses the need for routine drill exercises to be replaced with exercises that concentrate on the teaching of concepts. It is to this objective that this thesis aims to address itself.

J. Calmet [11] makes the very valid point that the high school is

'... probably a place very well suited for using computer algebra ...'

but then goes on to say that computer algebra is

'... a good teaching aid for non-math majors.'
It is felt that all pupils, from whatever discipline, and at whatever academic level, can benefit from CAS. Calmet seems to imply that mathematics specialists may not be aided in their learning of mathematics through the use of CAS. Or perhaps he insinuates that they ought to perform pencil and paper mathematics before using CAS. Calmet may even be trying to question the level of mathematical sophistication of current systems. Surely, if a particular system is not powerful enough for mathematics specialists, the aim should be to improve the system and not to deny pupils the use of the system.

The teaching of mathematics by discovery, as advocated by J.E. White [100] for the acquisition of strategies for finding antiderivatives, again assumes a high degree of mathematical self-confidence that is not usually found in the average school child when encountering a new concept. The notion of using a computer to test your rule, such as

\[ \int (F(x) + G(x)) \, dx = \int F(x) \, dx + \int G(x) \, dx \]

is exciting and adaptable for the use in secondary school.

It is noteworthy that all of these researchers use unmodified computer algebra systems for student experimentation but this is felt not to be the most effective method of delivering the precise algebraic aspects of the school mathematics curriculum. To a school pupil, the
complexities of learning abstract algebraic concepts and manipulations are considered a sufficiently difficult task and we should not additionally burden the pupil with learning the syntax of an unfriendly computer algebra package. It should also be noted that it is becoming more usual in universities for both mathematics students and students of other disciplines to use an algebra package or even have a significant section of a course delivered through the use of a computer algebra system.

The fact that more university mathematics courses consider CAS as an important method of delivering parts of the syllabus means that there is pressure on prospective university students to meet CAS earlier than at present and this was considered as yet another reason for the research.

The question posed in [93] 'Why Elementary Algebra Can, Should and Must be an Eighth Grade Course for Average Students', by itself illustrates why this research is important. Far too many average and above average pupils fail at algebra and maybe this novel approach of using CAS will bring more success to a larger number of pupils.

Having paid due attention to comments made by Apetsberger [3] and the lack of relevant primary evidence, or even others' experiences, the conclusion was reached that the main objective of providing a complete and definitive solution to the teaching of algebra through use of CAS was probably too
ambitious and needed to be sub-divided into a number of smaller more easily attainable short term objectives. These objectives are:-

1. Provide an on-line help system for muMath;
   (to give a 'stand-alone' system that will minimise teacher intervention).

2. Prepare a user-friendly front end for muMath;
   (effectively hiding muMath from the user).

3. Display an expression in the most natural mathematical way; this would be equivalent to the two-dimensional output routines of other CA packages.

4. Implement new mathematical routines in muSimp (the language in which muMath is coded) specifically tailored to school use.

5. Investigate appropriate methods of getting muMath to teach mathematical concepts.

It was never the intention of this work to diagnose the algebraic errors pupils make in the manner of Sleeman [81] or Brown & Burton [6]. Instead, the aim was to minimise the number of errors by providing a well thought through method
of delivery since muMath, in the background, would accomplish all the algebraic manipulations.

As teachers of mathematics become more aware of the capabilities of CAS, and as the technology exists for each pupil to own a symbolic calculator [44], it is expected that the level of algebraic achievement will improve. Just as the electronic calculator revolutionised aspects of the numerical mathematics curriculum, leaving teachers time to concentrate efforts elsewhere, it is envisaged that computer algebra systems, running either on a microcomputer or a calculator, will be equally effective in transforming the teaching of mathematics.

2 Computer Algebra Systems

Computer algebra is a branch of computer science concerned with the implementation of algorithms to perform mathematical operations on symbolic quantities. Frequently, such programs are large, complex systems containing hundreds of procedures or subroutines necessary for carrying out the required algebraic manipulations. Most of these routines are contained on a backing storage medium (disk) and only the kernel of the system is permanently resident in memory. The kernel together with the associated routines are collectively known as a Computer Algebra System (CAS). This is just one of many definitions that can be found in the literature. Loos [7] supplies another definition:
'Computer algebra is that part of computer science which designs, analyses, implements and applies algebraic algorithms'.

A short set of examples taken from the Mathematica manual exemplifies the character of all computer algebra systems.

In[1]:= p = 2
Out[1] = 2

This assigns p to have the value 2

In[2]:= 3p - p + 2
Out[2] = 6

Using the numerical value p = 2 the CAS can evaluate the expression

In[3]:= 3q - q + 2
Out[3] = 2 + 2q

q has NO value nevertheless the CAS is able to evaluate the expression SYMBOLICALLY

Over recent years, the power and versatility of CAS has been increasingly recognised, and several new systems have emerged. Research is progressing along two main avenues:

1. Increasing the power of systems by finding new algorithms, or amending and extending existing ones.

2. Widening the audience to whom CAS might appeal by taking advantage of the features offered by powerful underlying hardware.
A body of research, for example the papers [7,20,92], exists reporting work into the design and implementation of faster and more compact algebraic algorithms. One such book by J. Davenport et al. Computer Algebra [23], has drawn together recent research into an interesting readable volume. Unfortunately few such works exist of such a general nature! The volume Semi-Numerical Algorithms by Knuth [45] can easily be considered as a landmark in the reporting of algebraic algorithms and it is on this volume that more recent books have their foundations.

CAS are no longer text based; systems such as MAPLE and Mathematica have sophisticated graphic interfaces and facilities for producing complex graphical representations of two- and three-dimensional mathematical functions.

It is only over the past five years that CAS have received significant interest and there are still relatively few researchers working in the area. However, exploration of the subject can be traced back at least one hundred years. Computer algebra has foundations back in the mechanical world of the 19th. century when Charles Babbage wrote of his Analytic Engine:

'Many persons ... imagine that its processes must be arithmetical and numerical. This is an error. The engine ... might bring out its results in algebraic notation'.
It has taken the improvements in fast, cheap and reliable microcomputers to bring about the further developments in Babbage's work and an increased awareness of CAS by a larger more diverse community. The management of CAS into a viable product in a similar manner to the calculator has been relatively slow. M.MacCullum [50] identified three main causes for this slow development:

1. their existence was not widely known

2. their possible value was not appreciated

3. potential users did not have the requisite hardware.

It is against this background that computer algebra developed as a tool for use by theoretical physicists. In the research literature can be found many examples of the application of CAS in applied physics [AP1 to AP5]. In 1960 SCHOONSHIP was developed to calculate physical quantities in quantum electrodynamics. However, it was not until 1966 that the first conference on computer algebra was organised by the Association for Computing Machinery. The late 1960's and early 1970's saw the beginnings of more general and comprehensive packages. These packages were larger, slower and less convenient to use than the more specialised packages. Two such packages, MACSYMA and REDUCE, were respectively the work of Stanford University and Massachusetts Institute of Technology. They were a direct result of pure research with large mainframe computers and
encapsulated large amounts of mathematical knowledge. Over subsequent years further research has led to improvements in the algorithms and enhancement to the mathematical knowledge base. But the pressure has been for faster and larger computers. The packages were obviously designed by mathematicians and computer scientists for the primary use by other mathematicians and computer scientists to aid their research. Little thought was given to the wider usage of the systems and this is evidenced by some of the unusual mathematical notations employed.

Unfortunately MACSYMA and REDUCE have dominated the thinking of researchers when they have undertaken the design of other systems. When Stoutemyer redesigned muMath [85] with a wider audience in mind why did he almost exactly copy the syntax of MACSYMA for some commands? Even when MAPLE was under development some twenty years later the influence of these systems can be clearly seen in the similarity of the input syntax of many of the commands:

<table>
<thead>
<tr>
<th>MACSYMA</th>
<th>MAPLE</th>
<th>muMath</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOLVE(3*x-y=x,x);</td>
<td>solve(3*x-y=x,x);</td>
<td>SOLVE(3*x-y==x,x);</td>
</tr>
<tr>
<td>INTEGRATE(SIN(X),X);</td>
<td>int(sin(x),x);</td>
<td>INT(SIN(x),x);</td>
</tr>
<tr>
<td>DIFF(SIN(X),X);</td>
<td>diff(sin(x),x);</td>
<td>DIF(SIN(x),x);</td>
</tr>
</tbody>
</table>
There is an obvious advantage to having a similar syntax, experienced users of CAS can become quickly competent in the use of more than one system. However the point is that the syntax is not aimed at the novice.

The major problem with this disparity of written mathematical notation and the syntax of CAS is that keyboards do not support, at present, the usual mathematical symbols. A mathematician would write

\[
\frac{df(x)}{dx}
\]

and we would read it as,

\[
\text{'differentiate } f(x) \text{ with respect to } x'
\]

is almost perfectly matched by the computer algebra syntax `differentiate (f(x),x)`. A problem arises when we teach students to communicate mathematics the language in (1) and then require them to use the language in (2) to use CAS; this is an unnecessary burden but unavoidable with present technology. Mathematical notation has evolved over many centuries into a very succinct language, which often relies on context for its meaning. For example mathematical operations are often implicit, as in '3x' where the multiplication is implied. CAS can be built to interpret such cases of implied mathematics. Even with recent artificial intelligence research computers are limited in their ability
to reason over such a large complex field as mathematical language. Hence, the verbose nature of CAS input is a direct result of capabilities of computers and the need to eliminate input errors. In this sense perhaps mathematical notation, no matter however succinct or aesthetically beautiful, should move towards a more verbose system, such as that employed in current computer algebra systems. Surely mathematics and CAS notation should be identical, as far as possible? Nevertheless it is going to be a while before input technology allows an improved mathematical input.

The lack of suitable machines and the idiosyncrasies of packages had until very recently resulted in computer algebra being an exclusive tool of universities. Surely a user should be able to interact with such a system in a mathematical notation comfortable for the user and expect the program to respond in the user's mathematical notation. That is, the user should be able to type

\[
\text{solve} \quad 4x + 5 = 21
\]

and expect the program to reply with \( x = 4 \) rather than \( \{x==4\} \) (muMath notation).

Equally, users should be able to take the above questions and present them to the CAS as they would find them normally written in the literature and expect the system to deal with them effectively. Hence the above examples might become
Solve for \( x \) \[ 3x - y = x \]

\[
\text{solve} \int \sin x \, dx
\]

\[ \frac{d}{dx} \sin x \]

The development of hardware and software that can scan handwriting and intelligently interpret the symbols for input into a CAS no longer seems futuristic and avoids keyboard limitations.

Such mathematical notation eccentricities are trivial to the experienced user. However, the implications are of great significance for a school child encountering a computer algebra system for the first time having only recently been exposed to algebra. When using CAS pupils are effectively being asked to learn another, and to them different, mathematical notation. In essence therefore a CAS should be flexible enough for the novice and powerful enough to perform complex algebraic computations for the experienced user. Algebra systems containing very complex mathematical operations have been built but at the expense of expecting novices to use them. Is this what we want? Is it really what ought to happen? Hopefully, both answers to these questions should be an emphatic NO!
The major CA systems currently available offer roughly the same mathematical facilities. Some of the more specialised systems deal with small, self contained algebraic domains. To compare in detail the facilities offered by the major CA systems is not easy because of their differing intended audiences and machine environments. However, a commonality is discernible in that all systems typically provide the following facilities:

1. Simplification of terms e.g. $2x + x = 3x$.
2. Finding common factors e.g. $2xy - 2xz = 2x(y-z)$.
3. Substitution e.g. if $x=2$ then $5x = 10$.
4. Solution of polynomial equations upto the 4th degree.
5. Solution of linear simultaneous equations.
7. Expansion of polynomials e.g. $(x-2y)^3$.
8. Simplification of polynomials.
9. Calculation of matrices and determinants.
10. Calculation of Limits $\lim_{x \to 0} \frac{(x+1)^2}{\sin x}$.
11. Basic Calculus


14. Handling some special functions.

15. Expansion of Series.

16. The ability to program new rules, algorithms in an underlying language.

Beside these general features, each system has its own special features. For example, both Derive and muMath are capable of producing graphical representations of functions while MACSYMA includes routines for handling tensors. However, at a more fundamental level each major system varies in its ability to handle common types of problems. As an illustration, all systems are able to perform integration but the range and complexity of problems each system is able to solve differs considerably.

It must be said, however, that until recently all systems were designed primarily for research usage. This has meant that presentation, ease of use, and output of detailed solutions were NOT the primary concern of the authors as the intended audience required only results. Most of the facilities offered by the systems were direct outcomes of research into algorithms capable of producing solutions to large complex problems and an ever increasing range of problems.
Thus, no intermediate steps in the working of problems were presented for the user for study purposes, and no explanations of the resulting solution were available. This lack of output occurred because the users did not require detailed worked solutions and, more importantly, most systems solved problems in a dissimilar way to the user. The result from the input was just a raw answer. This, of course, was fine, the research user knew the manipulations / concepts involved and the intermediate steps were of little concern.

More recently, packages have begun to be developed with the emphasis on giving help, guidance and intermediate steps on the way to the answer, see M. Beeson [4]. These may be useful, even necessary, but it is questionable whether they are of value for teaching purposes. Some educational research, see Palmiter [63] into the teaching of mathematics indicates that the teaching of concepts is of more importance than the detailed and laborious manipulations of long algebraic problems.

The methods employed in CAS to solve problems are very different to those used by individuals doing identical hand calculations. An individual is taught a technique and applies what has been taught to an appropriate problem mainly by pattern matching techniques, whereas CAS applies algorithms to the problem. This subtle difference leads to CAS performing powerful mathematical algorithms on essentially simple problems, see [20, 45, 53, 91, 102]. This difference in
methods is the main reason why some CA systems are unable to produce explanations and steps in the mathematical working.

It is the recognition of the importance of acquisition of concepts that has lead to a change in the teaching and learning of mathematics. The practice of applying concepts leads to the acquisition of skills such as manipulating algebraic expressions.

Mathematics in schools has traditionally concentrated on the skills of applying and learning set algorithms for the solution of algebraic problems rather than the teaching of underlying concepts required to achieve a solution, as described by Burton and Brown [6]. This has lead to pupils being unable to absorb the fundamentals of algebra but being able to show a degree of fluency with algebraic manipulations. This has a direct parallel with the teaching of long arithmetic calculations. The calculator has changed the teaching of this area of the curriculum but it remains to be seen if CAS will do likewise for the teaching of algebra.

Hence the Algebra System discussed in this thesis has tried to use the mathematical knowledge embedded in a first generation system (muMath) to produce a second generation system in which concepts are taught through example and where the manipulations are completely hidden in the background. Nevertheless each concept to be taught should have a relevant example showing the manipulations involved for pupils to view
both as support and as an aid. The teaching of concepts does not eliminate the need to practise the manipulation skills but it may eventually pave the way for more pupils to gain a better understanding of algebra.

4 History of Computer Algebra

The newcomer to the study of computer algebra will be struck by the large number of synonyms for computer algebra currently in use. Calmet [12] has remarked that any combination of the words algebraic, symbolic, computation and manipulation describe computer algebra. It is therefore not surprising to find that the history and origins of computer algebra are difficult to trace.

However, modern computer algebra seems to have its origins back in the early 1950's when Kahrimanian and independently Nolan began work on symbolic differentiation. What obstructed computer algebra development was the lack of an underlying language able to handle lists of symbolic expressions. For this reason the production of LISP, by J.McCarthy at M.I.T. in 1965, was a significant event in the development of the computer algebra discipline. J.Sammet [73,75] produced an annotated bibliography of 300 computer algebra systems just one year later in 1966 and updated the list to 380 systems in 1968. The latest bibliography by Wyman [105,106,107] between 1968 and 1970 lists 550 items, however since 1970 no further compilation exists! The latest
historical survey of computer algebra was carried out by J.A. van Hulzen and J. Calmet [94] in 1982, nearly ten years ago.

The task of searching and collating the existing database of published research and then commenting upon significant contributions is vast. The proliferation of systems and their uses have led to research being reported in an ever increasing number of journals. The educational uses of CAS, however, are relatively small in number if current research publications are used as an indicator. This observation can be substantiated by examining the following table, which shows the numbers of research papers from January 1976 listed in INSPEC Computer & Control Abstracts.

<table>
<thead>
<tr>
<th>Year</th>
<th>Entries Listed</th>
<th>Systems Usage</th>
<th>Uses in Education</th>
</tr>
</thead>
<tbody>
<tr>
<td>1976</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1977</td>
<td>25</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1978</td>
<td>32</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>1979</td>
<td>19</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1980</td>
<td>16</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1981</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1982</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1983</td>
<td>23</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>1984</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1985</td>
<td>36</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>1986</td>
<td>64</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>1987</td>
<td>159</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>1988</td>
<td>222</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>1989</td>
<td>150</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>1990</td>
<td>252</td>
<td>29</td>
<td>7</td>
</tr>
<tr>
<td>1991</td>
<td>351</td>
<td>33</td>
<td>3</td>
</tr>
</tbody>
</table>

The table shows the spectacular growth in interest and research since 1982. Much effort has gone into algorithm exploration but limited work exists on the uses of resultant
systems and the application of CAS to education. However it is realised that INSPEC only surveys a limited number of research journals and that papers do exist in specialised educational publications; witness the work of D.Tall in Mathematics for Schools [L5].

Educational uses of CAS are reported in an uncoordinated manner at present, as there is no single authoritative and generally acceptable forum. Hence, this research points to the fact that such a forum should be established. It is acknowledged, however, that newsletters concerning CAS and its uses do exist both in this country and others; e.g. Maths & Stats -- University of Birmingham, U.K.

CASE newsletter (Computer Algebra Systems in Education)
-- Colby College, Waterville, USA.

Much of the current work reported relates to specific systems and concerns fairly well established algebra systems. Only a few major new systems have been developed in recent years, most work concentrating on the enhancement of existing algebra systems. Hence, by 1991, Wynam's original list of 550 packages has not expanded dramatically.

5 Computer Algebra Systems Compared

General Computer Algebra Systems cover a wide variety of mathematics of which only a small subset is required in the
5 Computer Algebra Systems Compared

classroom. The facilities offered by complete implementations of computer algebra systems extend beyond the normal mathematical knowledge used in the classroom. A demonstration of the facilities offered by seven well known systems, with features considered desirable for a classroom system, can be best viewed in the form of comparison tables. The five systems that were considered for the research are compared together with MACSYMA and REDUCE, the latter two acting as standards by which the other five can be viewed. A larger and more detailed comparison of both the mathematical and general facilities of these systems can be found in 'A guide to Computer Algebra Systems' [32].

The tables which follow compare:

(a) Mathematics capabilities;
(b) The user interface;
(c) Features for modifying the computer algebra system.

Key:

RED. = REDUCE Version 3.3
MAC. = MACYSMA Version 309.5
MU. = muMath Version 4.13 + Enhancements 1987
MAP. = Maple Version 3.3
DER. = DERIVE Version 1.63
SYM. = Symbolator (c) 1989
MAT. = Mathematica Version 1.0
Y = Yes -- the feature exists
N = No -- the feature does not exist
### TABLE A -- FEATURES FOR MODIFYING THE COMPUTER ALGEBRA SYSTEM

<table>
<thead>
<tr>
<th>GENERAL FEATURES Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Line Help</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Interactive Lesson</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Access to Op. System</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Source code Avail.</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Recall Depth.</td>
<td>All</td>
<td>All</td>
<td>1</td>
<td>3</td>
<td>All</td>
<td>All</td>
<td>All</td>
</tr>
<tr>
<td>Program Language</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>2-D output</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Available on P.C.</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

1. Implementation environments as of 1988.

<table>
<thead>
<tr>
<th>FILE HANDLING Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read commands from a file.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Write results to a file.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>
## TABLE B -- MATHEMATICS CAPABILITIES

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EQUATION SOLVING</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upto degree 4</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N²</td>
</tr>
<tr>
<td>Simultaneous linear equations</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Simultaneous non-linear equations</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Transcendental functions</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

$2f = x^4 - x^3 - x^2 - x - 1 = 0$ gives a runtime error 202

---

### BASIC ALGEBRA (Equation Solving continued)

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find common factors</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Complete polynomial factorisations</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Substitution</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Numerical Eval.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Arbitrary Precise arithmetic</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N³</td>
<td>Y</td>
</tr>
</tbody>
</table>

$3f = 2^{1001}$ yields a runtime error 205 and SYMBOLATOR quits to the operating system.
### TABLE C — CALCULUS

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Differentiation</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Partial Diff.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>INTEGRATION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polynomials</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Transcendental Functions</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Inverse Trig. Functions</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definite Int.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Numerical Integration</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>DIFFERENTIAL EQUATIONS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Order</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Second Order</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>
# TABLE D -- MATRIX OPERATIONS

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition/Subtraction</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Multiplication</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Inversion</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Determinant</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Eigenvalues</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Eigenvectors</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

## VECTOR ALGEBRA

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Dot Product</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Cross Product</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

## SPECIAL FUNCTIONS

<table>
<thead>
<tr>
<th>Feature</th>
<th>RED.</th>
<th>MAC.</th>
<th>MU.</th>
<th>MAP.</th>
<th>DER.</th>
<th>SYM.</th>
<th>MAT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Arith.</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Graphical Output</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>
From table A it can be clearly seen that all five packages considered for use in the classroom were available on a P.C. (the technology available in schools or currently being made accessible). However, it can be considered that muMath provides the better opportunity for adaptation and enhancement. The ability to enhance the system by providing extra functionality through the embedded programming language and the restriction of only being able to recall the last result makes it ideal for providing a simple classroom introduction to a Computer Algebra System.

From table B it can be seen that all systems offer approximately the same amount of algebraic knowledge. Table C indicates that muMath does not offer a Numerical Integration package and as schools normally teach the rudiments of the area this was considered as a domain where extra functionality could be added to muMath and simultaneously incorporated into the proposed classroom model.

Table D is included to give a degree of completeness and to establish muMath as a mathematically sound and reasonably complete system upon which any proposed work could take place.

The tables presented indicate that each system contains broadly the same functionality when compared against common criteria for performing mathematics. The situation is more involved when applying their appropriateness to the school
mathematics classroom. The next section explores the pros and cons of Maple, Mathematica, muMath and Derive for use in the classroom.

6 Teaching Systems

This section looks at the merits of current computer algebra systems that can be realistically considered for use in the classroom. It then looks at the reasons for the slow integration of computer algebra systems into school mathematics.

Computer algebra systems that can be used for teaching purposes can be seen to fall into two groups:-

(a) those specifically designed for education

(b) those that can be adapted for use in education.

Every computer algebra system can be used in education given powerful enough hardware and an abundance of creativity by the teacher. It is also fair to say that some systems were never targeted at the educational market even though financial considerations have begun to dictate that the market is worth exploring. Users of the systems that can be adapted for education would make convincing arguments that their system is really a member of the former group. This statement can be substantiated by the large amount of
published research where systems such as REDUCE and MACSYMA have been used in undergraduate university or college calculus classrooms.

Within the first group exist two sub-categories; those designed cover a broad band of algebraic concepts and those intended to teach or diagnose errors in a single algebraic concept.

During the early 1980's when microcomputers of limited power and number existed in school large computer algebra systems were not able to be used in school. However, the computing fraternity have realised the usefulness of the hardware for teaching algebraic concepts and have perceptively designed a number of single concept algebraic programs. (See for example Symbolic Calculus (1987) for BBC microcomputers from the Maths Workshop [P12]. Or SOLVE (1932) from the Association of Teachers of Mathematics [P13]). While these were acceptable at the time their features are now embodied in the larger, more powerful general group. By 1987 a major computer manufacturer, Hewlett Packard, had realised the power of computer algebra and had produced a hand-held symbolic calculator (HP-28C). At present this ought to be classified as an hybrid system, however its capabilities for 128K ROM based machine are impressive. The HP-28C can symbolically deal with complex numbers, matrices, vectors, algebraic expressions and some calculus problems.
Of the major purposely designed educational computer algebra systems four are worthy of further investigation. These are MathPert, Mathematica, Maple and muMath. Each system has something special to offer education.

**muMath**

The muMath user interface is limited to a linear line of text input, i.e. expressions such as

\[
\frac{x^2 - x^3 + \frac{1}{x}}{x}
\]

which has to be keyed in as \(x^2 - x^3 + (1/x)\)

However, the output from muMath can be directed through a 'pretty print' routine that essentially gives the impression of a two-dimensional format to the expressions. muMath's size, approximately 1 megabyte of source code and its relatively cheap cost at around 100 pounds, offers a user a large range of mathematical operations and the potential to solve many problems from differing mathematical areas. It should certainly not, because of its size or cost, be considered a 'toy' CAS package. In fact muMath has been used in situations varying from the school classroom to post
graduate research, and consequently it has gained a large following of dedicated users.

The strategy adopted by muMath's designers of being specifically for implementation on microprocessor based computers imposed limitations on the mathematical content. Additionally awkward, unnatural syntax is a direct implication of the original hardware restrictions, such as limited memory and restricted character sets. However syntax such as #E for the exponential function, #PI for π and the fact that all user input having to be in uppercase could easily have been avoided. A single line of code is all that is required to internally change all input to uppercase and this would alleviate hours of frustration and endless retyping.

muMath has to read in text source files containing mathematical routines/operations in order to compile the required mathematical operations. These can be saved in a compact internal format for execution at future sessions. muMath limits the amount of code that can be read in to be compiled, hence it is not possible to have muMath's total mathematical knowledge concurrently available. It is therefore essential to plan carefully the content of systems files. This limitation could have been overcome by giving muMath the facility of automatically loading the correct routines to deal with the problem presented by the user.
muMath was designed when the original IBM PC, 8086 microprocessor based machine was about to be released. Therefore some of the constraints imposed on the design of muMath by this early technology no longer applies when viewed against current IBM PC machines. However no upgrade of the system has been released to take advantage of the technology advances.

In addition help files were sparse if nonexistent in early versions of muMath and finally in 1987 the enhanced muMath contained a limited on-line help facility. All development of muMath by Soft Warehouse ceased in 1987 in favour of its successor Derive.

Derive

Derive, muMath's successor, has a more friendly human interface than muMath. It is based on the idea of a large mathematical work area at the top of screen and a menu structure on the bottom three lines of the screen. Input is still a linear line of input, but this time it is in the bottom part of the screen. The input is checked for syntax errors before being passed to the work area. When the input appears in the work area it does so in a two dimensional format akin to normal mathematical notation. As can be seen from the Derive examples below (Example 11). Derive also tries to use as much natural mathematical notation (√ as against sqrt, or 1/2) as the underlying IBM PC character set
allows. These seemingly minor points all combine to produce a more friendly feel to the package.

The Derive menu structure also negates the necessity of the user having to recall operation names or operator syntax as is the case with muMath. This facility is especially important when the package is being used by inexperienced users, as will be the case with school pupils. So for example:

\[ \int_{4}^{7} x^2 \, dx \]

in muMath is

\texttt{DEFINT (X^2,X,7,4)};

which the user has to either remember the syntax for or look it up in the manual, whereas in Derive the user types in \( x^2 \) and then selects the following options from the menus presented:

- \textit{Calculus}
- \textit{Integrate},
- \textit{Variable} = \textit{x}
- \textit{Lower limit} = 4
- \textit{Upper limit} = 7
- \textit{Simplify}.
The mathematical problem to be solved is built from the mathematical operations contained as entities on the menu structure and by the user supplying relevant variant data. This has the advantage for a young mathematician of reinforcing the underlying mathematical structures inherent in the problem.

As previously stated Derive is a CA system with its origins in muMath but its weakness is the lack of an underlying user accessible programming language. This absence forces what is an otherwise excellent CAS to be static in its mathematical functionality. Also the unwieldiness of using an intermediate language (GWBASIC) as muMath has to for plotting functions, has been superseded by internal code in Derive. Derive is a self contained package which only recently has come supplied with function libraries.

Up until 1990 Derive had not really received the attention it deserves as a CAS package, probably because two (Maple and Mathematica) much more mathematically powerful systems were released for IBM PC machines at approximately the same time as Derive. Maple also had a large user base from its previous implementations on mini and mainframe computers. This combined with the low cost of Maple for an IBM PC served to ensure that Derive has not established a large user base. Now, in 1993, Derive has over 20,000 registered users.
Maple

Maple's user interface is not as friendly as that of Derive, its friendliness is half way between those of muMath and Derive. Expressions are entered as linear lines of text and then displayed in the more normal two dimensional mathematical notation. However nice enhancements such as √ are not included.

Maple needs the power of at least a 80386 microprocessor based computer with a numeric maths coprocessor and a large amount of memory. It can clearly be seen from the status line on Example 6 below that a standard integration problem can take a significant amount of memory (1183K).

The microprocessor version of Maple is influenced by former mini/mainframe implementations. However, the mathematics contained within Maple is formidable when compared to Derive and muMath. The total Maple system requires 7 megabytes but contains many hundreds of mathematical operations, for example there are 77 operations beginning with 'A'. An annoying syntax quirk is the number of functions where two or more variants exist, differentiated by only the first letter, e.g. Exponential and exponential.

For sheer mathematical power, versatility and wide acceptability Maple has to be the choice system. However, in
the last two or three years Mathematica has begun to challenge Maple's superiority.

**Mathematica**

Mathematica occupies approximately the same amount of backing storage as Maple and also requires the power of an 80386 microprocessor based microcomputer. For sheer mathematical power Mathematica compares favourably with Maple, but their mathematical knowledge domains differ each having its own strong area.

Mathematica's alluring graphical functions exemplifies the power that computer algebra systems can offer. The graph plotting routines exploit the considerable video technology capabilities of current microcomputers to the limit. Graphing functions include two and three dimensional cartesian and parametric plotting plus contour and polyhedra together with sophisticated shading and surface illuminations routines.

A particularly welcomed feature is the ability easily to define usual mathematical transformation rules such as:-

\[ \ln(e^x) = x \quad \text{or} \quad \ln(x \cdot y) = \ln(x) + \ln(y) \]

These transformations can be achieved in almost exactly natural mathematical notation, thus the above would be defined as:-

48
log[exp[x_]] := x and log[x_ y_] := log[x] + log[y]

This feature allows more advanced school pupils to experiment with rules in order to reinforce mathematical concepts.

**MATHPERT**

MATHPERT is the most dynamic and the most exciting newcomer to education offering great potential for mathematics teachers. It was designed in 1988 by M. Beeson [4] primarily for education. Beside all the normal algebraic knowledge MATHPERT incorporates a user-model and uses this model to produce step-by-step solutions at a level of detail tailored to the knowledge of the user. These two factors, a user model and step-by-step solutions, make it of considerable importance to the educational world.

As stated MATHPERT can give a multiline explanation of a solution to a users problem. For example the common denominator problem:

\[
\frac{1}{x} + \frac{1}{y}
\]

results in six lines of reasoned output from MATHPERT. It is this capability that sets it apart from all current computer algebra systems.
As a direct consequence of this facility MATHPERT solves problems in much the same way as we teach our pupils. This is referred to as "cognitive faithfulness" by M. Beeson. Fast or complex algorithms, which do not mirror the methods employed by our pupils, are not the basis of MATHPERT as is the case in most other computer algebra systems. Thus, high power algorithms, such as the Risch Norman Integration algorithm, are not a feature of MATHPERT as these types of algorithms do not replicate the methods taught to pupils. As a result the range of mathematical problems that MATHPERT is able to solve is very much limited when compared to most other systems.

The ability to allow a user to interactively apply a series of mathematical operations to a problem also sets MATHPERT apart from its rivals. Given the problem

\[ x^2 + 2x^2 + 7x^4 \]

the user is allowed to apply rules to lead to a solution. So applying the step "collect powers" simplifies the problem down to

\[ 4x^4 + 7x^4 \]

then the step "combine terms" gives

\[ 8x^4 \]
Given a freedom of choice the user is allowed to apply rules which may not perform a useful step but nevertheless it will produce a mathematically correct answer. Thus MATHPERT allows spurious steps to be performed.

The user model consists of a simple coding system for all the operations, approximately four hundred, contained within MATHPERT. An individual view of whether the user is "learning", "knows", "well-known" or "unknown" for the current operation is kept so as to direct a user's course of study.

The educational world will need more published research before approval and adoption is given to MATHPERT.

Having now outlined the capabilities and opportunities offered by each of the four major systems one is left with feeling that none of them adequately addresses the needs of the average school pupil learning algebra for the first time. In fact expectations such as ease of use, a clear teaching methodology and user manageability all seem to be ignored. Each system has its own merits but none of the four was specifically designed with the sole or foremost intention of being a teaching tool. Each system would lay claim that it is primarily designed as a teaching tool. It must stated that to some degree each system can be successfully used in the classroom as experiments outlined in research papers such as [3, 14, 36, 101] testify. It does however seem that a disparity
exists between what the average mathematics teacher and discerning pupil distinguish as their needs from a useful tool for teaching and learning algebra and what is currently offered by computer algebra systems.

A teaching system with firm foundations, based on a computer algebra system, with the power of Maple/Mathematica, the user friendly interface of Derive and the adaptability of muMath is necessary. Evidently this is the ideal and any final system would in the first instance be a compromise consisting of elements of each of the required considerations.

To serve as a demonstration of the mathematical output from the four systems each was presented with two algebraic problems taken from an 'A' level examination paper. It should be made clear that the default settings for each system were used and no attempt was made to force the systems to give a solution in any particular format. The problems were

Problem 1: Solve \( x^2 + 1 = 0 \)

Problem 2: Differentiate \( e^{-x} \sin \sqrt{3} x \).

The output from the final question was fed back into the integration routine within each system to demonstrate the merits of the package when used in a teaching context. This was done in order to demonstrate the Fundamental Theorem of Calculus.
In the second problem, the output from the differentiation was integrated with the expectation that the result would look similar (apart, possibly, from an arbitrary constant) to the original function. It turned out that Derive and muMath met this expectation whereas the more sophisticated systems did not. In the case of Mathematica, the integration could not be performed, and with Maple an additional simplification step was used but this did not provide exactly the correspondence expected (although it is easy to see that a simply trigonometrical identity would yield the result).

The input and output from each system is now given:

---

**Mathematica**

Mathematica 2.0 for MS-DOS 386  
Copyright 1988-91 Wolfram Research, Inc.

```plaintext
In[1] := Solve[x^2 + 1 == 0, x]  
Out[1] = \{ \{x -> 1\}, \{x -> -1\} \}
```

**Example 1**

```plaintext
In[2] := D[E^(-x) * Sin[(3)^(1/2) x], x]
Out[2] = \frac{\sqrt{3} \cos(\sqrt{3} x) \sin(\sqrt{3} x)}{x E^x} - \frac{\sqrt{3} \cos(\sqrt{3} x) \sin(\sqrt{3} x)}{x E^x}
```

**Example 2**

```plaintext
In[3] := Integrate[%, x]
Out[3] = Integrate[............................., x]
```

**Example 3**

---

53
Maple

solve(x^2+1,x);

\[ 1, -1 \]

**Example 4**

F1-Help F2-Find F3-Quit F4-DOS F5-Review F10-Menu ( ) 159k 43sec

diff(e^(-x) * sin((3)^{1/2}x),x);

\[ (-x)^{1/2} (-x)^{1/2} 1/2 -e \ln(e) \sin(3x) + e \cos(3x) 3 \]

**Example 5**

F1-Help F2-Find F3-Quit F4-DOS F5-Review F10-Menu ( ) 1180k 189sec

int(" , x);

\[ (-x)^{1/2} e \tan(1/2 3 x) \]

**Example 6**

2 ---------------------------

1/2 2

1 + tan(1/2 3 x)

F1-Help F2-Find F3-Quit F4-DOS F5-Review F10-Menu ( ) 1180k 189sec

simplify(");

\[ 2 e \sin(1/2 3 x) \cos(1/2 3 x) \]

54
MuMath

muSimp-83 4.12 (03/30/85)
MS-DOS Version
Copyright (C) 1982 The SOFTWARE HOUSE
Licensed by MICROSOFT Corp.

? SOLVE(X^2+1==0,X);                   Example 7
@: { X == - #I,
     X ==  #I }

muSimp-83 4.12 (03/30/85)
MS-DOS Version
Copyright (C) 1982 The SOFTWARE HOUSE
Licensed by MICROSOFT Corp.

DIFF (( #E^(-X)) * SIN ((3)^(1/2)*X), X);         Example 8
@: 3^*(1/2)*COS (3^(1/2)*X) / #E^X - SIN (3^(1/2)*X) / #E^X

? INT(@,X);                           Example 9
@: SIN (3^(1/2)*X) / #E^X

Derive

2
1: x + 1 = 0                        Example 10
2: x = - i
3: x = i
   -x 1/2
4:  ê SIN (3 x)
   d  -x 1/2
5:  - ( ê SIN (3 x))
     dx

-x
6:  ê (√3 COS (√3 x) - SIN (√3 x))    Example 11

-x
7:  ê (√3 COS (√3 x) - SIN (√3 x)

-x
8:  ê SIN (√3 x)                     Example 12

55
As expected all four packages were able to solve the two problems and with exception of Mathematica (see Example 3) each was able to integrate the differentiated expression. One of the systems, Maple, did not give the result expected from the integration and even after applying another operation (simplify) the outcome was not as expected, even though it is of course an equivalent solution (see Example 6). This seeming inconsistency causes mathematical insecurities to occur for young mathematicians and if we are trying to promote the use of CAS as an aid to teaching then packages should be cognitively faithful as expressed by M. Beeson [4].

Further exploration of the outputs from the systems shows Derive to have the closest to handwritten mathematics. The input formats of Derive and muMath are closely analogous, in that an expression is input in a one-dimensional format. The results above are after Derive has processed the input and produced a two-dimensional output for display purposes. muMath can clearly be seen to have the most awkward user interface both for input and for ease of interpreting the output.

From the outputs given above Derive and muMath, in general, gave better results if compared against the assumption that for use in school systems should look and behave like real mathematicians do when they are solving mathematical problems. As muMath is the forerunner of Derive, and considered by us to be reasonably cognitively faithful,
with the ability to have additions programmed then it makes it highly suitable for school usage.

Of course there are many other systems that would all make claim to unique features. However it is our opinion that the four described will be the main influences in secondary education over the next few years.

Since many computer algebra systems have been created by university researchers, it is not surprising that most published reports on the educational use of these systems are from university teachers. A few small groups of university teachers have reported uses of computer algebra systems as an aid in mathematics instruction (Hosack, Small, Jane [39] and Fresse, Lounesto, Stegenga [28]). Individual American Colleges and British Universities have encouraged their students to use computer algebra. Only relatively few mathematical courses exist where computer algebra is an essential element of the course. The impression gained from reading the published reports on the use of computer algebra systems in education is that such uses are isolated experiments, and are not yet part of a rapidly-spreading trend.

Until 1988 only three dissertations had been completed concerning the use of computer algebra systems in education (Heid, Palmier [63] and Hawker [33]). All concerned the use of computer algebra and calculus. Each was able to show that
the use of computer algebra produced only positive or neutral educational results.

Ralston [67], in 1985, claimed that symbolic mathematical systems would have a similar impact upon the curriculum as did hand calculators. The first full year the hand calculator was marketed was 1971 and after 7 years nine dissertations were complete on the uses of hand calculators in education. muMath's first full year of marketing was 1979 and after 7 years no dissertations were recorded on its educational uses. After a further two years the situation was 16 and 2 respectively. This contrast merely serves to reflect the slow pace of dissemination of computer algebra systems into the classroom. It is fair to say that computer algebra systems have attracted only a small amount of attention. This slow dissemination is due in part to the limited market of scientists and analytical engineers to which computer algebra systems are of practical use. Furthermore, the use of computer algebra systems by mathematics teachers still depends on the occasional rebel teacher or a local computer buff willing to experiment. This dissertation is one of only a small number to report upon the classroom work of pupils using muMath to learn algebra.

This research also aims to provide evidence that CAS can be successfully used in the classroom even though other educators (W. Page [61]) have expressed misgivings such as:-
'Because of their power and increasing availability, abuses and misuse of CAS can also cause considerable harm to the vitality and integrity of mathematics education'.

'Since much remains unknown about the role and value that routine processes and computations play in learning mathematics, it is not obvious what intellectual deadwood should be culled from or traded off from a course's syllabus'.

Hopefully the findings of this research will directly contribute to discussion of the use of CAS in the mathematics curriculum. It also seeks to address the above misgivings that educators have expressed about letting young algebra learners use sophisticated computer algebra systems.
1 Introduction

This chapter investigates the functionality of a system which could meet the aims outlined. It analyses the views and expectations of software from both a pupil and teachers' viewpoint. Obviously the two are not mutually exclusive and hence any software developed needs to incorporate both audiences' expectations. This therefore forms the analysis section of the work. While performing the software analysis and taking account of the software requirements we need to be mindful of:

(1) the current software situation, as of 1988
(2) of the views and expectations of pupils and teachers.

None of the existing computer algebra packages which are currently (1992) marketed satisfy all the objectives outlined in chapter one, section three. Notably the embedding of a teaching strategy within a computer algebra package is lacking in all systems. However it is acknowledged that some systems such as; Mathematica, MAPLE and DERIVE have a mechanism whereby lessons can be taught.

In 1988, when this work was initially proposed, muMath was readily available, additionally the source code of the mathematical algorithms which constitute muMath were also
1 Introduction

included with the package. The implementation language (muSimp) was also an integral part of the muMath package. But most importantly in 1988 muMath was the only system which would run on relatively cheap and minimal hardware configurations.

Due to the perceived ease of adaptation, the facilities offered by muMath and its relatively low cost and hence within the budgets of most schools' it was eventually selected as the basis for the proposed system.

The ideal adaptation of muMath into a system suitable for classroom use consists of many related strands. None of these strands are new, all already exist in current research, however no one piece of research unifies all the various strands. The main strands being:-

1. The use of computer technology and computer algebra systems for the teaching of mathematics.

2. The teaching of algebraic manipulative skills through the underlying mathematical concepts.

3. Tailoring of software to meet the needs of each individual pupil.

4. Providing sufficient records for staff to make sound professional judgements.
1 Introduction

At the outset of the work the objective of enhancing a current system for classroom use necessitated that both pupils and teachers were consulted as to what their perceptions were of what constitutes a good software teaching system.

Some requirements of the system are outside the control of the intended audience, for example the hardware prerequisites of the software. As muMath is available on both 8 bit and 16 bit machines research was carried out into the type and quantity of hardware currently in schools and future purchasing plans. The results were fully predictable in that schools have, in 1991, already started to phase out 8 bit machines in favour of the newer 16 bit technology. So the minimum hardware specification for the present implementation is an IBM or IBM compatible microcomputer with a colour monitor, one floppy drive and a hard drive of at least 10 megabytes, (a numeric co-processor is also advantage).

The hardware used to develop the system was an AMSTRAD 1512 microcomputer with a 32 megabyte hard drive. This machine was also used to carry out the experimental use of the resulting system. Because of the demands made upon the hardware, the software ran relatively slowly, with obvious implications on the ability to capture the attention of users over a long period. The software takes 64 seconds to complete "one cycle" on this hardware configuration, but experiments with a Mitac microcomputer based on the 80386DX - 25 Mhz chip
shorten the software cycle time to just 6 seconds. "One cycle" is defined to be the time taken for the software to display a question, muMath to solve it, the user to select an appropriate answer from the concept menu and finally the software to return ready to display the next question. The cost differential between the two machines is approximately 750 pounds (1992 prices) but the effectiveness upon pupil concentration times was easy to detect mainly through the adverse and complementary remarks for each machine (Amstrad 1512 and Mitac) respectively.

The pedagogical advantages eventually shown to be gained from the more powerful 80386 technology reflect upon the resourcing consequences needed to use computer algebra software in schools. The machine power critical to such algebraic teaching packages will demand considerable use of advanced technology which at present is not freely available in most schools. The conclusion can only be that the full use of algebraic teaching software will need considerable investment in hardware and not that algebraic teaching systems should be disregarded as a teaching resource. Also computer algebra systems seem likely not to fully establish themselves, in the immediate future, in the classroom on a widespread scale, financial restrictions rather than unsound educational principles seem to be the main limiting factor.
Discussions with pupils about what they considered a good software package contains lead to the following key issues:—

Interesting - stimulating
Screen Layouts
Software - reliable
Help and Manuals - supportive

The remainder of the section looks at each of the above in more detail.

Interesting

The most frequent expression uttered by the average pupil today, both in and out of class, is 'its boring'. Lessons are all to frequently considered boring by some recipients. Government ministers and educationalists alike have expressed the need for teachers to always present their subject content, to pupils, in an interesting manner. Unfortunately the majority of routine work is repetitive and uninteresting.

If however, this repetitive practice of skills can be delivered by a computer rather than by more traditional methods, pupils might be less likely to view the subject content as boring. Variety of delivery methods can be used as
a teaching methodology and teaching practice and as a means to stimulate pupil interest. We would suggest that when pupils criticise lessons they are often criticising the mode of delivery as opposed to the subject content. This it should be made clear is an observation based on a number of years of experience rather than solid research evidence. However, even though the proposed system would aim not to be repetitive we did find that exercises of 30 or 40 similar questions were still viewed by pupils as excessive, yet teachers considered quantity essential for skill development. The use of a computer as a means of subject content delivery did have an influence upon pupils learning, favourable informal comments after work sessions with pupils confirmed this view.

Screen Presentation

Pupils are bombarded by visually dynamic material almost from the day they are born. A vast amount of an average pupil's time is spent viewing television, video or interacting with visually compelling video games. All this visual material only serves to amplify a pupil's expectations of the quality of educational materials used by teachers. Research into the effects of television viewing habits of children and their subsequent expectations is well documented in the literature. The research is also complemented by a similar body of work into psychology of colour schemes for computer screen layouts. In general this research points to the fact that colours such as bright white and yellows should
be avoided in favour of more pastel colours. Research tends to indicate more favourable user reactions to screen layouts in pastel colour schemes and this has been taken into account when designing ExPress.

Thus, screen layouts in any software system need to be attractive, colourful and display only the necessary information. Long textual passages serve very little purpose as often they are ignored as are screens containing vivid colours.

Software

A considerable number of software packages communicate with the user by a series of beeps or audible warning sounds. This communication system is fine for video games where noise and striking visual images seem to be the prerequisites for an exciting game, however pupils tend to dislike noise / beeps when using educational software, and in fact noise can often deter some pupils from using a software package. This observation is founded on a pupil's fear of exposure of their failure to their peers as very often audible warnings are used to indicate a user error. Hence, we conclude that audible warnings in educational computer software, no matter how well intentioned, should be avoided.

Evidence and good teaching practice tends to support the view that it is far better to reward success and hence
reinforce the positive learning achievements while trying to minimize failures.

Hence the proposed system will only use an audible communication to inform the user of a positive learning achievement in the form of a small tune and even this can be optionally turned on or off by each individual user. As a reward the pupil will be presented with a certificate (see appendix 3) at the end of each working session. This is flexible enough to link into the school's own individual reward system.

Manuals or Help

Most pupils, and to a lesser degree adults, tend not to want to spend time reading manuals describing how to use an item of software. We are sure the sentiments expressed in the phrase

'If in doubt read the manual'

is well practised by most computer users. If this is the case then manuals should reflect this situation and contain only minimal information. To a large extent the minimal content, and appropriateness of content, can be achieved through careful program design. The vast majority of what is contained in expensive to produce manuals could be provided in the form of on-line help files. With the consistent use of
a single key stroke (commonly F1) help should be provided at any point within a system, thus by-passing the frustration of having to leave the computer to search a manual for the required words of help. ExPress does not have a manual, the system has copious on-line help files.

Summary - Pupils' View

Most pupils are now fairly frequently exposed to computer packages either specifically designed as a classroom teaching resource or those designed for commercial use. The use of a menu to access the functionality of packages is a common approach; and hence pupils perceive this method as an easy channel or route into a software package. The type face clarity and layout of text / graphics on a VGA (Video Graphics Adapter) monitor also holds pupils' attention and motivated individuals over longer periods than the lower resolution CGA (Colour Graphics Adapter) monitor. Favourable comments from users on the VGA monitor included:-

' ... Easier to read the screen .... ' 

' ... The colours were brighter and more varied which helped to highlight the important points ... ' 

We believe that a pupil's perception of the worth of a software package is viewed through the quality of the monitor, as suggested by the above two users comments. The
consequence for educational hardware purchasing and software designers is clear. Pupils are now more discerning due to the proliferation of high quality video games and home microcomputing hardware. A user interface which can be driven, explored and easily used without constant reference to a manual would also gain pupil confidence.

Pupils seemed to be saying that a high visual quality, menu driven package with as few commands as possible are the basic requirements of any system. We initially decided to investigate GEM [P13] and Windows [P14] as the front-end interface for our computer algebra teaching system. Both packages were eventually rejected because of the need to write specific code to interface either system to muMath. We considered the design and coding required no more difficult than writing our own new package. Also writing our own package would allow us to reflect and specifically incorporate our objectives. Both GEM and Windows require the user to remember large numbers of key sequences and this was seen as a major disadvantage especially when they were only the driving software for the underlying algebra system. With a mouse the interface is somewhat easier. Our menu package requires the user to memorise:

ESC - to exit back a level having made a error
ENTER - to confirm input
TAB - to view the previous screen
SPACE - to move to the next screen
F1 - for help
2 The Pupils' View

and 4 arrow keys for movement around the menus and lists.

These keys, to a large extent, should be prompted for by the system. Also, no audible warning of an incorrect keypress is given as with a number of other packages. An audible warning given to a pupil by a computer is very often comprehended by pupils and their peers as failure; and fear of failure is all too often associated with mathematics and algebra in particular.

3 The Teacher's View

Theoretically, from a teacher's viewpoint, a fully worked algebraic solution, with comments, at a level of intricacy suitable to the current user would be most desirable. Also, a system which could suggest problems and alter the level of question difficulty automatically when the user demonstrates a familiarity with the concepts contained within questions. In contrast, there were also requests for a system capable of tutoring pupils on algebraic topics with the manipulations hidden. As will be seen some of these aims were achieved by the ExPress system. At the very least a clear sound teaching strategy should be visible in the resultant system.

The design of the proposed teaching strategy eventually became:-
1. The user either inputs a problem at the keyboard or the system reads a problem from a prepared file.

2. Then the background computer algebra system solves the problem.

3. The problem and the solution are presented to the user in the normal 2-dimensional notation.

4. The user is then invited to make a selection from a list of options which describe the actions that could have taken place in the transformation of the problem into the solution. Only problems with a limited number of steps in the transformation are considered suitable for this teaching strategy.

5. If the selection is correct the user is rewarded with a 'success output' and goes onto the next problem.

6. If the user chooses quit from the list of actions they progress to the next problem.

7. If the user chooses help from the list, a single screen gives help either on the current topic, the individual problem or the current option selected from the list of actions.
Additionally, in the background, the system would store information about how the user responds to the problem. This student profile could then be used by the suite of teachers' utilities to produce an analysis of the user's performance. From this analysis the teacher (the professional) would then be able to design an individualised learning scheme by providing the user with a tailored set of problems.

An additional, if minor aim, was for the system to alleviate some of the burdensome administrative tasks associated with teaching. Such administrative tasks as mark keeping, progress reports and rewards to pupils for good work are relatively easy to achieve and would allow staff the time for more important issues such as designing individualised study programmes.

After discussions with other colleagues, a number of issues were highlighted as being important. The issues included:

- System Manageability
- Ease of Installation
- Error Resistant
- Tight Control
- Utilities

The remainder of this section looks at each of the above in more detail.
Teachers' time is always at a premium. Hence, if software requires a large amount of their time, especially with a single pupil, the teacher will perceive this as a waste of a valuable resource. They obviously will not resent the time spent with the individual but would rather spend the time with all pupils perhaps achieving the same objectives by employing a different mode of delivery. Most teachers recognise that complex software has a learning curve which has to be negotiated, but this curve should be as easy to conquer as possible. Hence, rapid learning will give a sense of real advancement and maximize the incentive for staff to use the software with their pupils. Badly designed software quickly becomes unmanageable with a class of pupils, and usually never gets used again even though the learning objectives are commendable. So, once the initial introduction to a software package has been given, pupils should, in the main, feel confident enough to use the software unsupervised. Under this scenario the computer plus the software effectively becomes the pupil's private tutor, and thereafter frees the teacher to deal with other issues such as diagnosing problems and giving remedial help.

The point must be that software should be designed with specific teaching objective(s) and implemented in a clear and as easy to use style as is reasonably possible.
Ease of Installation

With any large complex software system a single disk inserted into a disk drive and 'away you go' is not a viable proposition. Hence complex systems, such as the anticipated system, need to ensure that a painless installation procedure is provided for installing files onto a hard disk and into the appropriate directories. Software is all too frequently provided with either minimal documentation or at the other extreme mounds of manuals, both of which conveniently hide the installation procedure in some obscure section of a chapter. Additionally documentation tends to be written by a computer expert for another computer expert whereas the average classroom teacher does not normally possess the technical jargon, expertise or confidence to install large software packages. Major systems should therefore be provided with a simple installation program which clearly prompts the user for the disks required. A single command, clearly printed on the first disk, should be all that is required to start the whole sequence of events.

The system aimed to achieve these objectives and when eventually used by semi-computer literate colleagues few adverse comments were forthcoming. In fact most colleagues explicitly stated how simple they found the system was to work with, this included non-mathematics specialists.
There is an unwritten but well-known adage:

'Give a new program to a class of children and inevitably one of them will crash the program or cause the system to hang'.

Moreover that pupil will be totally unaware of how they have achieved this unremarkable feat but the child will almost certainly expect the 'all-knowing' teacher to fix their error immediately. This pupil confidence in their teacher often causes the teacher embarrassment because the pupil will very probably have had more contact time, with the software, than the teacher. Software should therefore aim to be as error resistant as possible. This is easy to say but almost impossible to achieve. In the event of a system crashing a simple screen giving clear instruction should be provided to enable a reasonably well controlled recovery to take place. The error recovery procedure should also log, on the disk, the cause of the error so that the teacher can view the cause, replicate the error, and hopefully take the appropriate action for future users. ExPress has aimed to incorporate a comprehensive error recovery procedure. As the production of totally error resistant software is extremely difficult to achieve realistically in a finite time an error recovery procedure should be considered a necessity by educational software designers.
When pupils are introduced to a new software package their sheer inquisitiveness dictates that all parts of the package must be explored immediately, even before the teacher has given them instructions. Whilst this is not a bad fault in itself it can cause tremendous problems when dealing with large complex software systems such as ExPress. Such pupil curiosity can destroy the aims and objectives of a lesson. With this in mind a tight control of the learning situation should be available but be as unobtrusive as is possible so as not to give the user the impression that they are being restricted. This is achieved in ExPress by levels of access given to the user. As will be explained in chapter 5, at the minimal level of access to ExPress the user is restricted to the use of a single command. However, flexibility is accomplished by the single command taking its input from a file of questions. This situation allows the teacher to decide which material and mathematical operations are to be studied by each individual pupil. In no sense is this a restriction designed to discourage learning, it is merely a method of ensuring that all pupils time is concentrated on the task of learning a prescribed set of algebraic concepts rather than time being spent exploring! The system has been designed to ensure that a teacher always has the flexibility to enable any pupil, at any time, to explore further simply by changing their access level. This tight control also gives the teacher a sense of security, briefly mentioned early, as
the teacher has peace of mind knowing that pupils cannot stray from the lesson and question them on unfamiliar sections of the software. This need for security is real and we believe that educational software designers should be acutely aware of this when designing their product.

Utilities

It should be remembered that educational software ought to be designed to teach pupils and to aid teachers in their professional and administrative duties. Most software currently produced achieves one or the other of these functions. The results of pupils interactions with a software package designed primarily to teach a topic should record the pupil achievements and then have the facilities to allow teachers to analyse, review and assess individual pupils progress so that further programmes of study can be set. The facilities for analysing pupil results, with current high technology, need no longer be tables of endless raw numerical results. The results can be presented in a more natural and easily understandable format such as graphs. In addition to the assessment and reviewing of pupils accomplishments teachers have to administer the system and this requires, from the software designer, a clear simple procedure which teaching staff can easily recognise as their normal daily routine.
Summary - Teachers' View

There is an abundance of educational software on the market, most of which is written and produced by software engineers with little educational experience. The main objectives of such software is commercial gain as opposed to educational laudability, it would be grossly unfair, if not a little dangerous, to name individual software items. However, this situation tends to lead to the production of software which does not always satisfy pupils and does not meet the professional expectations of teachers. Software engineers often have preconceived ideas, usually founded in the adult world, of how software packages should be presented for the intended user. At the other extreme when teachers start producing software they, in general, lack the computing skills necessary to produce stable computing systems. Unfortunately, there therefore exists a situation which more often than not is filled by an amateur and therefore good quality software is difficult to find.
Chapter 4 -- Description of the Resulting Software

Introduction

The whole of this chapter is concerned with taking the results of the analysis, outlined in the previous chapter, to produce the design for a system that could reasonably be used in the classroom.

Section 1 of this chapter reviews muMath, its advantages and disadvantages for the use in school. Section 2 describes extensions made to muMath to overcome the disadvantages discussed in section 1. Section 3 takes the extended muMath together with the objective of the work, to use CAS in school, and produces ExPress a system for classroom use. The final section details some of the major display features used in ExPress to transform muMath into a friendly school computer algebra system.

1 muMath

muMath has over the last decade undergone a number of changes culminating in a complete rewrite to produce DERIVE. Even though DERIVE has a friendlier feel and better user interface, muMath has the advantage of being able to be adapted and extended in the underlying language. This is not so in DERIVE.

muMath, the first computer algebra system for a microcomputer, is a collection of non-numerical algorithms
muMath coded in a base language called muSimp. muSimp is a LISP like language designed in 1976 specifically for implementing computer algebra systems. The resulting syntax employed by muMath has a similar feel to that of the older and larger MACSYMA system. A larger more detailed description of muMath can be found in the book muMath: A Microcomputer Algebra System by C. Woof and D. Hodkinson [104].

Before proceeding to explore the extensions made to muMath it would helpful to know why extensions were considered necessary. The following commentary on a muMath work session (further on in the text on page 85) serves to highlight the eccentricities of the system which would ultimately detract young inexperienced users from the mathematics.

Commentary:

(N.B. The questions, with some re-phrasing, contained in the commentary are those asked by pupils from Northicote School Wolverhampton while using muMath).

Example 1 This is a standard solution of the linear equation $2X - 1 = 0$, but why does muMath give a 'solution set' for the solution and why a double '=' sign? \[ 80 \]
Example 2 This problem is the simplification of a standard algebraic expression

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

The question asked was, "why is the 'Y' term first, in the answer from muMath, when textbooks teach us that algebraic terms should be in alphabetical order?"

Examples 3 & 13 These two example show what happens when a mathematical operation does not exist or is spelt incorrectly. The user receives no assistance or explanation. muMath has, nevertheless, responded in a seemingly inconsistent manner by parsing and re-ordering the terms in the expression of example 3 but not in example 13.

Example 4 Factorise the expression $x^2 - x$. muMath outputs the solution with a multiplication sign, when we are taught that the sign is implicit, why does muMath automatically insert it when it is not required?

Example 5 This is the standard expansion of two brackets to form a trinomial. muMath deals with this as we would expect!

Example 6 & 7 muMath has been asked to find the solution to:

$$\frac{d}{dx} (AX^2 - BX)$$
and has given the wrong solution! Is this because DIF does not exist, or does muMath not know how to differentiate? It is neither; it is simply because the constants A and B have to be separated from the variable X by a space as example 7 shows.

Examples 8, 9 & 10 These examples show what happens when you try to find a solution to

$$\frac{d}{dx} (\sin Y)$$

Example 8 gives the wrong answer yet example 9 verifies that muMath can differentiate the Sine function. It is not until Example 10 where muMath is told that Y is a dependant function of X that the solution can be achieved.

Example 11 The output from the Taylor expansion of Sin(Ln X) is not given in the normal format of:

$$\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} X^{2n}$$

or optionally:

$$- (X - 1)(X^2 - 4X + 4X^2 + 6X - 19)$$

\[ 12 \]
Example 12 The solution of the integration problem:

\[ \int \frac{X}{aX^2 + b} \, dX \]

gives an output almost exactly as expected except the log expression is reversed, why?

Example 14 The solution of the integration problem

\[ \int X \sin X \, dX \]

The output here is not very clear. Does

\[ -X \sin (2X)/4 + X^2/4 - \cos (2X)/8 \]

mean

\[ \frac{-X \sin (2X)}{4} + \frac{X^2}{4} - \frac{\cos (2X)}{8} \]

or does it mean

\[ \frac{-X \sin (2X)}{4} + \frac{X^2}{4} - \frac{\cos (2X)}{8} \]
For a seasoned mathematician experienced in the use of and the interpretation of output from CAS the clue is in the middle term. A CA system would have cancelled the $2/4$ to $1/2$ if it were the second option, hence the first option must be correct. This is not always so obvious to new or under confident mathematician and a better display procedure would help.

The 14 examples clearly show not only the faults of the system when considered from the point of view of classroom use but also the power that exists in such a small system. The extensions made to muMath will now be outlined to show how the deficiencies exemplified through these examples were overcome.
Example 1
? SOLVE(2X-1==0,X);
@: {X==1/2}

Example 2
? 3X^2 + 2X^2 - X + 5X + Y;
@: Y + 4 X + 5 X^2

Example 3
? FACT(X^2-X);
@: FACT (-X + X^2)

Example 4
? FCTR(X^2-X);
@: X * (-1 + X)

Example 5
? EXPAND( (3X + 2) (5X + 9) );
@: 18 + 37 X + 15 X^2

Example 6
? DIF(A X^2-BX,X);
@: 0

Example 7
? DIF(A X^2 - B X , X);
@: -B + 2 A X

Example 8
? DIF(SIN Y , X);
@: 0

Example 9
? DIF(SIN Y , Y);
@: COS Y

Example 10
? DEPENDS (Y(X))$

Example 11
? TAYLOR (SIN LN X , X, 1, 5);
@: -1 + X - (-1 + X)^2/2 + (-1 + X)^3/6 - (-1 + X)^5/12

Example 12
? INT(X/(A X^2 + B),X);
@: LN (B + A X^2)/( 2 A)

Example 13
? DEFINIT(1+A #E^X/X/X, X, 1, 2);
@: DEFINIT(1 + A #E^X/X/X, X, 1, 2)

Example 14
? INT(X * (SIN X)^2 , X);
@: -X SIN (2 X)/4 + X^2/4 - COS (2 X)/ 8
2 Extensions to muMath

Extensions to muMath is a deceptive heading in that in the final system ExPress only uses muMath for its algebra engine. The term 'algebra engine' is used here to refer only to the mathematical operations and algorithms contained within muMath. Thus by only using the mathematical operations ExPress, the end product, did not remotely resemble muMath. Further, the additions made to adapt muMath fall into two main categories:

1. Extensions made to the mathematical domain of muMath to overcome some omissions which are normally used and taught to an average mathematics class.

2. The additions needed to produce ExPress.

In their book, muMath: A microcomputer algebra system, C.Wooff and D.Hodgkinson [104] draw the reader's attention to a number of omissions in muMath. These are:

1. No on-line help facility
2. No output format control
3. No graphical output
4. No number theory functions
5. No numerical methods
6. A factoring capability for expressions such as \(ab + ac + bd + bc\)
Of these the first five were considered to be of prime importance and hence they embody the main extensions made to muMath. These will be described in the next section, the remainder of this section will describe the work required to produce ExPress in order to test the theory that a Computer Algebra System could and should be regularly used in the secondary school mathematics classroom.

Current mathematical research by Small & Horsack [40], Stoutemeyer [87], Wilf [102] and Zorn [111] into the learning of algebra frequently cite the need to distance children from monotonous repetitive manipulations in order to emphasise the essential concepts and conclude that the use of CAS should be used. Research by Sleeman [81], over many years, has shown that the solving of equivalent equations e.g.

\[ x + 2 = 5 \]

is an equivalent equation to

\[ x + 2 - 2 = 5 - 2 \]

is badly understood by a significant number of students. The outcome of a study of 96 students, by Sleeman et al [82], clearly points to the teaching of concepts for solving equivalent equations rather than the manipulative procedures. Recent research by Palmiter [63] into the use of MACSYMA for the teaching of calculus concepts, such as,
Definition of the Integral
Fundamental theorem of calculus
Inverse Functions
Techniques of Integration
Applications of the Integral

has shown that pupils taught by a computer algebra system outscored a control group on both concept and computational understanding.

Attempts by educational psychologists and research workers have been made to categorise the types of learning found in human beings. One simple category focuses on learning facts i.e. 'knowing that ... ' and learning skills i.e. 'knowing how ... '. Raphael [68] identifies four categories:

- rote learning.
- parameter learning.
- method learning.
- concept learning.

In Rote Learning information is simply transferred into long term memory for recall when needed. A good example of this is the situation found in many schools where multiplication tables are memorised. A result of rote learning is the ability to improve performance at, in this
example, solving more difficult problems involving basic multiplication facts.

Parameter Learning is best explained through an example. Take the situation that exists in many mathematics classrooms throughout the country. The teacher / textbook presents a child with the problem:

Solve \[ 3x + 1 = 9 \]

This problem is solved by applying a pattern for the solution, hence once the pattern has been learnt all problems of the type

Solve \[ ax + b = c \]

can be solved. Notice however the solution of minor deviations form the pattern e.g. \[ ax - b = c \] cannot be solved because the new question does not conform to the pattern previously learnt.

Method Learning occurs when a child can apply a method to a problem which contains minor deviations of the same problem. Hence, having been taught how to solve \[ ax + b = c \], the child can apply the method to solve problems of the type

\[ ax - b = c \] or \[ ax - bx + c = d \] etc.
Concept Learning requires the ability of classification and the ability to analyse a problem. Hence, having learnt the method of solving a linear equation in one unknown, concept learning takes place when the child is able to determine which, if any, of a given set of problems to apply the method to in order to obtain a solution.

ExPress was designed to address the later two types of learning. In our opinion these cannot be divorced from one another, if one is shown to take place then the other exists and has taken place. Hence, the work was undertaken to implement an algebra system 'accessible' to the average school pupil and usable by such a pupil to improve method/concept learning of algebraic procedures.

Essentially what was required in order to transform muMath into ExPress was an analysis of the phrases currently used by teachers when teaching algebraic techniques. Once this was completed, a method was needed to display these phrases in a menu window once the pupil had studied a question and muMath's solution. In summary, pupils were supplied with a question, the correct solution and a list of choices (phrases) describing actions that could have taken place. Of the choices only one could be correct. All the pupil needed to do was select the correct phrase.
As an example:-

Find the solution for the equation.
\[ x + 7 = 15 \]
The solution is \[ x = 8 \]

This component of the system was completed by muMath which then handed over to ExPress. ExPress presented a menu of choices for the pupil to select the correct action i.e.

- Add a number to both sides.
- Remove the brackets by multiplying.
- Subtract a number from both sides.
- Multiply both sides by a number.
- Divide by a number.
- Quit.

The quit facility, which was used as an 'I don't know' option, was included in all menus to allow pupils 'a way out'. In the experiment, described in chapter 7, quit was only used twice in over two thousand questions answered.

Each category of problem requires its own menu of choices. The phrases used were the ones currently used by teachers in the school where ExPress was eventually used in the experiment. Each suite of phrases is contained in a text file prepared with the aid of a wordprocessor.
2 Extension to muMath

In assigning questions to categories with broad descriptors for the concept required to produce a solution we became acutely aware of the subtle differences in wording and the wording effects on giving a clear statement. This obviously has enormous implications for the teacher when wording questions in lesson exposition and teaching materials. Further this observation may point to the need for a standardisation of wording in order to maximize pupils' learning and minimize their misconceptions. Subtle wordings often interchanged with apparent ease and little thought by teachers cause pupils tremendous problems. Two English phrases commonly used to describe \(- x\), often in the same sentence, are 'minus x' and 'negative x'. The concern is not for which is mathematically correct but for consistency of approach initially among individuals and then at large.

3 Additions made for Classroom Use.

Of the five major additions made to muMath four will be detailed in this section and the fifth, output format, in the following section. As a reminder the five additions made to muMath were:

A. An on-line help facility
B. Output format control - (dealt with in section 4.)
C. Graphical output
D. Number Theory
E. Numerical methods
A. The On-Line Help facility

The current trend in most commercially produced software is context sensitive help. As most pupils in school use packages which include this facility it was felt that this would be the best approach for the muMath on-line help.

The help facility with muMath-87 is a minimal system which is partially menu driven, forcing the user to find the help required by searching numerous unwanted pages. A context sensitive system is intelligent enough to find the precise help required from the positioning of the on-screen cursor.

Each command on the ExPress Pull Down Menu (PDM), described in Chapter 6, has a help screen describing the mathematical operations. Those commands which point to a sub-menu plus the top level menu bar also have help screens. In addition each mathematical operation has a screen which consists of the syntax (modified from the muMath manual) and a mathematical example of the operation.

At a higher level a further more comprehensive help system is provided. This specialised help facility is connected to the choices a pupil can select from when trying to describe the action ExPress has taken in order to solve the current question. Hence in this respect it is extremely specialised as it only applies to the extra facilities required for the design of the experimental use of ExPress.
Each help page for each descriptor on the concept menu works through an example of the current type of question with the concept the pupil has chosen. A typical example is:

Find the solution for the equation.
1. $x + 7 = 15$

The appropriate menu of choices is:

Add a number to both sides.
Remove the brackets by multiplying.
Subtract a number from both sides.
Multiply both sides by a number.
Divide by a number.
Quit.

If a pupil selects help on the concept of 'adding a number to both sides' the following help page would be given.
Example Question: Solve $a + 15 = 31$

\[
\begin{align*}
a + 15 & = 31 \\
a + 15 + 15 & = 31 + 15 \\
a + 30 & = 46
\end{align*}
\]

The Number Added to Both sides is 15

This now requires a fair amount more work to solve the question.

Whereas, if the pupil selected the correct concept for this problem and asked for help, the following would be given:

Example Question: Solve $a + 12 = 31$

\[
\begin{align*}
a + 12 & = 31 \\
a + 12 - 12 & = 31 - 12 \\
a & = 19
\end{align*}
\]

The Number subtracted from Both sides is 12

This also allows the pupil to view the manipulations taken to achieve a solution. Each new step in the manipulation is highlighted in colour, hence the $-12$ on line 3 would be output using a different foreground and background colour to the remainder of the line thus emphasising the manipulation. Within the text files ExPress interprets some
3 Additions made for Classroom Use

special meta-characters as colour codes. As an example, @Solve the following Equation®, would display the text in yellow on a black background.

C. Graphical Output

Maple and Mathematica contain routines to graph functions. Mathematica in particular has an impressive suite of routines which range from ordinary 2-dimensional function plotting to spherical 3-dimensional surface plotting. ExPress should allow pupils the ability to explore graphical representations of functions. Provided with muMath-87 is a program written in BASIC which will plot, in two dimensions, a function. It is not necessary in school to be able to plot in 3-dimensions surfaces and hence rather than code a suite of graphical routines ExPress makes use of the routines provided with muMath-87. Those areas of the mathematics curriculum which require 3-dimensional plotting are limited and muMath's successor DERIVE contains to a limited degree, when compared with Mathematica, the necessary routines. As ExPress is also only a demonstration of how a CAS can be adapted for school use, 3-dimensional plotting was not considered as a high priority for implementation.

D. Number Theory

Again the two main educational contenders, Maple and Mathematica, both contain number theory operations which
could be useful in school. This is because most current classroom work on pre-algebra is taught through number and number patterns. Indeed this is precisely the approach advocated by all reports leading up to the production of The National Curriculum for Mathematics [L2]. It was therefore felt that we needed to show that:

(a) number theory functions could easily be added to muMath.

(b) ExPress could call and use these functions.

The routines added to muMath for school use were:

MERSENNE -- Calculates Mersenne Numbers
AP -- Finds the Sum of an Arithmetic Progression
GP -- Finds the sum of a Geometric Progression
NCR -- Statistical Combinations
PERM -- Statistical Permutations
BINOM -- Calculates Binomial Coefficients
PASCAL -- Calculates PASCAL triangle Numbers
FERMAT -- Calculates Fermat Numbers
FACTORS -- Calculates the Factors of a number
PRIME -- Tests if a number is Prime

A full listing of the contents of all routines is contained in Appendix 5. All the number theory routines were coded in muMath's base language, muSimp.
3 Additions made for Classroom Use

In order for ExPress to call these routines care was needed to ensure that the inputs to these extra operations conformed to the inputs of the standard muMath operations. Obviously this decision was made so that the new number theory routines would demand no special treatment by ExPress in order for them to be called.

As stated in the first section, muMath's routines contain only 16 different types of input. The routines added for number theory need only use four of these types and therefore no new input or output routines were required in order to add these routines.

E. Numerical Methods

Only four routines were considered of immediate use for school, these were:

<table>
<thead>
<tr>
<th>Routine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECT</td>
<td>Rectangular Approximation</td>
</tr>
<tr>
<td>TRAP</td>
<td>Trapezoidal Approximation</td>
</tr>
<tr>
<td>SIMPSON</td>
<td>Simpsons Approximation</td>
</tr>
<tr>
<td>NEWTON</td>
<td>Newton's Approximation for finding roots of polynomials</td>
</tr>
</tbody>
</table>

All of the routines are limited in their functionality to varying degrees, but the principle of how they could be implemented is demonstrated. The main aim here was to show
that such routines could be readily incorporated into the system.

4 Mathematical Display Features.

As already stated, a major disadvantage with muMath is the awkward syntax used to input questions and the equally obtuse method used by muMath to output its results. For example the solution of a quadratic requires the input to muMath to be

\[
\text{SOLVE}(x^2+3x-7=0,x);
\]

Why must SOLVE be uppercase? Why must there be two = signs? These constraints add complications for young learners. The output is even stranger:-

\[
\theta: \{x = -3/2 + 37^{(1/2)}/2, \\
x = -3/2 - 37^{(1/2)}/2\}
\]

The mathematical validity of either the input or the output expressions is not in question. However, neither are consistent with usual handwritten representations of the expressions, i.e. those normally taught in school.

For muMath to be of use in the classroom these disadvantages need to be overcome. When an expression is keyed in to muMath to be evaluated it would normally also need to be displayed as though it were a result. Hence, a routine to display input and output expressions in the more
natural 2-dimensional mathematically handwritten manner would greatly enhance muMath for classroom use.

Technically the 2-dimensional display routine is more sophisticated than any of the other additions made to muMath. What is simplicity itself for humans to write

\[
\frac{-3}{2} + \frac{\sqrt{37}}{2}
\]

requires many hundreds of Pascal programming lines, based on a complex recursive descent routine to analyse parts of expressions. Eventually the method used in the implementation is based upon the notion that any expression can be viewed as a series of simpler expressions. Using a textual reverse polish representation of the expression, a recursive descent algorithm both checks the algebraic validity of the expression and displays both inputs to ExPress and results from muMath.

Hence what the user inputs as \( \frac{X^2 - 3X + 5}{2} \) is displayed as

\( X - \frac{3}{2} X + 5 \) and what muMath outputs as

\[
\theta: \{ \begin{align*}
X &= -\frac{3}{2} + \frac{\sqrt{37}}{(1/2)}/2, \\
X &= -\frac{3}{2} - \frac{\sqrt{37}}{(1/2)}/2
\end{align*} \}
\]

is displayed as
This 2-dimensional feature is partially available in the enhanced version of muMath (muMath-87); where it is implemented in fewer muSimp lines of code. Nevertheless, to analyse all the many and varied expressions output by ExPress through muMath's 2-dimensional display routine was considered more difficult than writing a new routine. There are two main reasons at the heart of this difficulty:

(1) The way in which muMath routine deals with unusual or malformed expressions is inconsistent and at times unpredictable.

(2) The problem of adapting a routine without access to documentation.

In the early version of muMath [P1] there is complete consistency among all the output formats.

This new enhanced display routine is at the heart of making muMath into ExPress the user friendly classroom teaching aid. Displaying expressions in a more normal format was considered a significant step forward and the name of the software produced to support the research reflects this, ExPress: expressions in 2-dimensions.
4 Mathematical Display Features

Hence muMath was enhanced for classroom use in the following ways:

1. By providing a comprehensive on-line help facility.

2. By providing a two-dimensional output format.

3. By provision of a simple graphing package.

4. By provision of simple number theory functions.

5. By provision of numerical integration methods.
Chapter 5 — How does the Software Work?

1 Introduction

Having considered, in the previous two chapters, the analysis and design requirements of a classroom computer algebra system this chapter discusses the implementation details.

The implementation complexity of ExPress is related to the initial concept of making an interactive Computer Algebra System more presentable and user friendly for classroom use. Also, the number of objectives highlighted in chapters 1 & 3 eventually resulted in a large complex system. To make the use of ExPress as simple as possible the software evolved into a main executable file and several supporting text and muMath files. This simplistic design strategy allowed the resulting software to evolve quickly and efficiently without endless recoding or minor changes of design direction being undertaken due to unforeseen implementation problems.

ExPress put simply is:

"a suite of programs and text files controlled by a number of Operating System batch files in which the programs and functionality of ExPress are accessed through the use of a specially designed pull down menu".

It is also necessary to recall that ExPress was initially implemented on small, slow personal computers. That is,
machines with insufficient room to hold a great deal of data in main memory, in addition we did not want to recode muMath. Inevitably, therefore disk files had to be used as a means of extending the amount of effective memory available.

The programs and sub-systems communicate with each other through a number of intermediate disk files. These disk files are dynamic in that they are read from, and written to, by ExPress, and at the end of the working session they are erased in order that ExPress can start afresh at the next session. However, ExPress does check for inconsistencies, and if any are found, ExPress proceeds to 'clean' the system before continuing. The 'cleaning' operation involves closing all open files and, more importantly, all temporary work files created by the system are deleted to bring the system back to a known state. This concept is a direct analogy to that which happens when the operating system XENIX/UNIX is started.

ExPress can be viewed as 3 sub-systems

(i) The Operating System Batch files.
(ii) muMath - plus the relevant system files.
(iii) The Pascal code for ExPress and its many supporting text files.

All three subsystems are required in order to ensure ExPress functions correctly. But before a full account can be given
we need to describe the various levels of access to ExPress that are available to the user.

Modes of Operation - Levels of Access

ExPress was designed with the intention of being a working classroom system of use to both pupils and teachers alike. As a consequence of these two distinct audiences and their different expectations of the software, a layered mode of access to the software was considered the most appropriate.

As stated, ExPress operates in layered modes which can be viewed in the form of a pyramid:

```
  Teach Mode
    User Mode
      Full User Mode
        Manager Mode
```

At the highest level of access, called Teach Mode, a user is presented with questions and taught the various concepts needed to solve them. This is a very limited access to the system in which the user (the pupil) only has the use of the single mathematical operation needed to solve the question posed.
The next level of access is called User Mode. A user is allowed full access to all the mathematical facilities offered by ExPress. No attempt is made to do any teaching, and ExPress is used as a full, normal interactive computer algebra system.

At the lowest-but-one level of access, called Full User Mode, Express allows the user to pose a question and supply their idea of the solution. ExPress then solves and cross-checks its solution with the solution supplied by the user. If the two are found to be different, by which is meant mathematically unequal, ExPress tries to indicate a reason for the discrepancies. This feature was not fully implemented when the experiment, described in chapter 7, was undertaken, but it has subsequently been used with success while under development by a few individuals who had taken part in the experiment. The pattern matching techniques used in the comparisons of the two solutions are relatively crude, but even so the general feeling from those individuals who used this part of ExPress was that they were able to see the errors they had made.

At the deepest level of access, called manager mode, all the facilities of the above levels are accessible to which are added utilities required by teachers for:

(a) system maintenance,
(b) viewing pupil records,
2 Modes of Operation

(c) providing other teaching materials
(d) extending the system through use of the muMath program development system.

The two top levels of access are worthy of a more detailed clarification before continuing as they form the majority of the work.

In Teach Mode ExPress keeps a data file stating what each individual user's current state of progress is in relation to the set of algebraic exercises currently being studied. A second file for each user contains information relating to the path through the exercises that they should take. The aim of this final facility is to allow each user to progress as fast, or as slow, as they wish.

In User Mode the idea of trying to make a computer algebra package intelligent enough to point out mathematical errors to a user was explored. As most computer algebra packages perform their computations differently to humans, only final answers could be checked. However, with fairly crude pattern matching techniques ExPress has the ability to point out some routine errors. Most of this is achieved by holding expressions in an internal format, required by the 2-dimensional display routines, and then simply comparing the formats. As an example:

\[ 3x + 1 \] would be encoded internally as \text{NIAN}.
whereas

\[ 3x - 1 \] would be encoded as NISN

( \text{N = Number, I = Identifier, A = '+', S = '-'} )

and hence ExPress can interpret this as a difference in the binary mathematical operations.

Hence, in conclusion, in the implementation of the ExPress system we have tried to achieve the following four aims of providing:-

(1) a structured method of teaching algebraic concepts at the 'teach mode' level.

(2) a method of user-friendly access to the power of a computer algebra system.

(3) a method of experimentation and extension of the facilities of a computer algebra system.

(4) a suite of system utilities to aid a classroom teacher when using ExPress for the teaching of algebra.

Each sub-system performs a discrete section of the functionality. The large amount of functionality required from ExPress coupled with the limited memory capacity of the hardware selected, forced the final implementation to consist of a number of sub-systems. The remaining section of this chapter investigates the sub-systems and how they interact and communicate with each other.
As outlined earlier the software consists of three integrated sub-systems namely

(i) The Operating System Batch files.

(ii) muMath - plus the relevant system files.

(iii) The Pascal code for ExPress and its many supporting text files.

Each one of these three parts will be dealt with in detail in the remainder of this section.

(i) The Operating System Batch Files

The need for operating system batch files to achieve the aim of allowing ExPress and muMath to communicate became a requirement as the system crystalised. After a long investigation it was found impossible to combine the PASCAL code for ExPress with the muSimp interpreter and muMath source code into a single executable file. Also, even if this could have been achieved, the final runtime ExPress system file would have been far too large to have run on the small personal computers available at the time. Having the facility of operating system batch files also allowed the system to
achieve the large range of facilities as only one of the ExPress sub-systems is memory resident at any one time.

The functionality of ExPress is therefore best viewed through the batch files as these are consequently at the heart of the software. In total the system needs five batch files, they are:

A. LogIn --- A log on screen for name and password
B. Run ExPress System --- Load and run ExPress
C. Run muMath --- Load and run muMath
D. Graphing --- Load a BASIC program for graphing functions
E. Go --- A file to repeat from batch file B.

Each file performs a single operation and the batch files B. to E. are repeatedly called so that a user can continually use the system until the exit option is selected. The interaction between these batch files can be clearly seen in figure 5.1:
A further refinement of the batch file 'Run ExPress' will give clearer insight into how the various processes have been achieved. This is shown in figure 5.2.
Parameter 1
C:MUMATH this informs the Pascal code where
and which menu to load and display.

Parameter 2
and are either 0 or -1 to indicate whether to
allow INPUT to ExPress or to DISPLAY results
from muMath.

Batch file 5, GO.BAT, is merely a method of re-entry to
ExPress after a user has requested use of the MS-DOS
operating system.
As the ExPress system is heavily reliant upon these five MS-DOS operating system batch files, the content of these files are given for further clarification in Appendix 8.

The command to start the system is LogIn. This command executes the file Login.Bat which contains the commands shown in reference 1 in Appendix 8. In essence, the commands display an opening screen followed by a screen for the user to type in their name and password. The login command then calls a loader program before passing control over the Runthesy batch file.

Having gained access to the system, through the login batch file, the batch Runthesy (Run the system) takes over and executes ExPress by informing the system of the name of the menu files and where to find them. Additionally, ExPress makes use of the user's name and password in order to load specific user information and to restrict their access to the system. On exit from ExPress the Runthesy batch file determines which of the other batch files is to be executed. The content of Runthesy.Bat is shown in reference 2 in Appendix 8.

The muMath batch file links the main ExPress system to muMath and loads the appropriate muMath system files. muMath takes its input from the file called inp, interactively created from the users inputs to the ExPress system. The
3 Communication between Sub-Systems

The objective of requiring a large complex system to execute on a small personal computer was therefore overcome by the design and implementation of a number of operating system batch files. These files repeatedly call each other until such time as the user wishes to exit the system.

The next section looks at the second sub-system of muMath and the relevant files.
reader. It clearly indicates the large number of files that need to be in memory before an operation can be performed. As an example, to get muMath to compute:

\[ \text{SOLVE}(3 \times -9 \Rightarrow 0, x) \]

requires files ARITH.MUS, ALGEBRA.ARI, EQN.ALG and SOLVE.EQN. ExPress 'knows' that the operation SOLVE is contained in the muSimp system file SOLVE.SYS, so that when muMath is called by ExPress, this pre-compiled file contains the mathematical operation required.

\[ \text{muMath - 83 Dependency Diagram} \]

\[ \text{MUSIMP.COM} \]

\[ \text{ARITH.MUS} \]

\[ \text{ALGEBRA.ARI} \]

\[ \text{ARRAY.ARI} \quad \text{EQN.ALG} \quad \text{ABSVAL.ALG} \quad \text{DIF.ALG} \quad \text{LOG.ALG} \quad \text{TRG.ALG} \quad \text{HYPER.ALG} \]

\[ \text{MATRIX.ARR} \quad \text{SOLVE.EQN} \quad \text{ATRG.TRG} \]

\[ \text{LINEQN.MAT} \quad \text{ODE.SOL} \quad \text{INT.DIF} \quad \text{LIM.DIF} \quad \text{SIGMA.DIF} \]

\[ \text{VEC.ARR} \quad \text{ODEMORE.ODE} \quad \text{ODENTH.ODE} \quad \text{INTMORE.INT} \]

\[ \text{VECDIF.VEC} \]

116
As stated above it is possible to pre-compile these files into an internal pseudo code and leave an environment image in a muSimp system file on a disk. This environment image loads from the disk in a much shorter time than that required to compile the source code files, and having such an image by-passes the need to compile algorithms every time muMath is used. As not all the algebraic routines will fit into memory at any one time it was decided to break the code into several more manageable yet coherent parts. These divisions were not arbitrarily made but have a direct relationship to the subject areas taught in school. From the original system a number of algebraic routines were deliberately omitted as they would find little use in the average mathematics classroom. The resulting five muSimp system files used by ExPress contain the following muMath files. Routines are indented to indicate dependency:

3 Communication between Sub-Systems

SOLVE.SYS

Files are:-

ARITH.MUS
ALGEBRA.ARI
EQN.ALG
SOLVE.EQN

CALCUL.SYS

Files are:-

ARITH.MUS
ALGEBRA.ARI
ABSVAL.ALG
DIF.ALG
LOG.ALG
TRG.ALG
ATRG. TRG
HYPER.ALG
INT.DIF
INTMORE.DIF
LIM. DIF
SIGMA. DIF

MATRIX.SYS

Files are:-

ARITH.MUS
ALGEBRA. ARI
ARRAY. ARI
MATRIX. ARR
LINEQN. MAT
VEC. ARR

117
3 Communication between Sub-Systems

G.SYS

Files are:-

PDS.SYS (Program Development System)

Files are:-

ARITH.MUS
    ALGEBRA.ARI
    ABSVAL.ALG
    LOG.ALG
    TRG.ALG
    ATRG.TRG
    HYPER.ALG
    MYROUT.ARI *

DISPLAY.MUS
    EDIT.DIS
    CONSOLE.EDI
    DEBUG.MUS

* This file contains the code written in muSimp to demonstrate that muMath is easily expanded. An example of the code especially written for this work is shown in Appendix 2.

To summarise the situation so far. We have designed a system of batch files to overcome the memory limitations of the intended hardware environment. Also, muMath has been adapted, extended and stored in a state which facilitates easy communication between necessary sub-systems. All that is now required is to look at how these two major sub-systems were made to interact to achieve the required aims and objectives of providing an algebraic teaching system based on a computer algebra system.

(iii) - The Pascal code and the support files.

The previous two sections have described the material required to perform the symbolic mathematics and the files to 118
keep the ExPress system continually running. The files and code required to achieve the functionality for ExPress can be further categorised into four sections:

A1. Pascal code to interface ExPress and the menu system.

A2. Pascal code for the teachers' utilities.

A3. Communication files.

A4. Specialist teaching files containing teaching material.

These four sections should be examined in light of the initial aims and objectives which centred around providing muMath with a friendlier feel, embedding a teaching strategy and providing teachers with a method of tailoring the ExPress system to meet the needs of individual pupils. These four sections will now be discussed in more detail.

(A1) Pascal code to interface ExPress and the menu system.

The decision to write a pull down menu system for ExPress was taken when it was realised that interfacing muMath and a pull down menu would be significantly easier, and allow more flexibility, than using an existing menu system such as Windows or Gem. Hence the code required to
interface Express with the pull down menu system was conceived in three parts:

(a) The PASCAL code required to display and drive the pull down menu system. The definition of the ExPress pull down menu being read from files.

(b) The PASCAL code required to analyse inputs needed for the muMath algorithms together with the code required to call the internal routines to get the user inputs from the keyboard.

(c) The PASCAL code required to read help files and interpret embedded meta- characters into colours for highlighting text.

Because the pull down menu system was implemented from the definition of an abstract data type the discussion of the above point (a) is completely contained in Chapter 6. The other two points, (b) and (c), are now discussed in further detail.

Analysis of muMath input data types.

The analysis of muMath data types can be best described with reference to an example. Suppose a user wishes to solve a standard quadratic equation of the form:

\[ x^2 + 2x - 3 = 0 \]
this would be viewed by ExPress and muMath as

\[ \text{Expression}_1 = \text{Expression}_2 \text{ Variable} \]

or

\[ E_1 = E_2 \text{ V} \quad \text{--->} \quad \text{SOLVE} (E_1 = E_2, \text{ V}) \]

transform

Hence the problems that the implementation needs to address are:

1. How to generate correct muMath expressions from simplified user inputs.

2. How to generate prompts for the appropriate parts of a muMath expression, that is, prompting the user for the appropriate muMath data types.

After considerable investigation of all of muMath's algorithms, plus those algorithms provided by us, it was surprisingly found that they only needed, in total, 16 different input types. One extra dummy input type was included to act as an end of input marker. Additionally it was also found that any of the muMath mathematical algorithms needed at most six inputs.
3 Communication between Sub-Systems

On further examination the 16 input types can be more strictly defined into two divisions:

(a) Data types
(b) Input symbols

The input types were each given a unique symbol. Then each of muMath's mathematical algorithms requiring inputs were coded and placed in an ordinary text file which is read by ExPress for each question requiring a solution.

The file has four fields for each mathematical operation. They are:-

Comment field, Algorithm Name, Symbols, Menu option number.

As an example:

Expand EXPAND E***** 0008

The comment field (Expand) appears in the window of the pull down menu which the user selects. The Algorithm binds the user's selection to the appropriate muMath function. The Symbol (E***** ) is used to prompt the user for input information. The Menu Option Number (0008) is used for internal calls to the 2-dimensional display routines.

122
The 16 input types and their respective codes are:

(a) Data Types

<table>
<thead>
<tr>
<th>Input Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>A mathematical expression</td>
</tr>
<tr>
<td>v</td>
<td>Multiple Variable Names</td>
</tr>
<tr>
<td>V</td>
<td>Variable Name</td>
</tr>
<tr>
<td>I</td>
<td>Integer</td>
</tr>
<tr>
<td>n</td>
<td>Real Number</td>
</tr>
<tr>
<td>L</td>
<td>Upper and Lower Limits</td>
</tr>
<tr>
<td>P</td>
<td>Value of Decimal Places</td>
</tr>
<tr>
<td>S</td>
<td>Scalar</td>
</tr>
<tr>
<td>-</td>
<td>Vector</td>
</tr>
<tr>
<td>R</td>
<td>Row Vector</td>
</tr>
<tr>
<td>e</td>
<td>Vector Expression</td>
</tr>
</tbody>
</table>

(b) Input Symbols

<table>
<thead>
<tr>
<th>Input Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>= Automatically input by ExPress</td>
</tr>
<tr>
<td>)</td>
<td>) Automatically input by ExPress</td>
</tr>
<tr>
<td>@</td>
<td>Infix Operator</td>
</tr>
<tr>
<td>T</td>
<td>True Automatically input by ExPress</td>
</tr>
<tr>
<td>[</td>
<td>[ Automatically input by ExPress</td>
</tr>
<tr>
<td>*</td>
<td>Dummy code - need to pad the field to 6 characters</td>
</tr>
</tbody>
</table>

123
3 Communication between Sub-Systems

This coding system is crucial to the implementation of ExPress and as such a longer, fuller description of this approach may clarify the reasoning behind its selection.

Taking a further example, suppose that the user wishes to solve an algebraic expression of the form:

\[ 2x^2 - 3x = 1 + x \]

The system needs to:

1. Convert the users wish into the appropriate muMath expression

\[ \text{SOLVE} \ (2x^2 - 4x - 1 = 0, x); \]

2. Prompt the user for:

   (i) the two algebraic expressions \(2x^2 - 3x\) and \(1 - x\)
   (ii) the variable \(x\).

This works as follows:

1. The user chooses, from the Operation Window of the Pull Down Menu, the option to Solve an algebraic equation. The software now 'knows' that the muMath algorithm \text{SOLVE}. 

124
2. A look-up table (Appendix 6) shows that SOLVE requires three parameters:

(i) E - the expression on the left-hand side
(ii) E - the right-hand expression
(iii) V - the variable 'to be solved for'.

and certain other symbols as shown by the six character template E=EV**, where

E stands for an expression,
V for the name of a variable
and = represents the muMath == symbol.

(The asterisks are used to complete the six character field and play no further part in the translation process).

In the example, this template translates to the muMath representation

SOLVE (2x^2 - 3x == 1 + x , x);

where the comma, parentheses and semi-colon are automatically inserted by ExPress.

The template, E=EV**, also acts as the input to the user interface by defining what prompts are needed. For example
the symbol E results in a window being displayed with the prompt "Please input an expression".

Once the user has entered a response to the prompt an error checking routine is called. For example, the fact that an expression, E, has been prompted for, the error checking routine would examine the user's input to determine whether or not it corresponds to a correctly formed expression.

As soon as ExPress has read the relevant code and checked the user's inputs, the correctly formed muMath code is written to the disk. Eventually ExPress instructs muMath to interpret this code in order to find the solution. muMath then writes its solution back to the disk and returns control back to ExPress. ExPress then interprets the output from muMath to intelligently display, in 2-dimensional format, the answer(s).

This system is obviously limited because if further algorithms were to be added requiring either a new data type or more than 6 inputs then the whole data file would need recoding. This, however, was not viewed as a major drawback as the intention was not to produce a polished product but more a prototype system for classroom use.
3 Communication between Sub-Systems

Help Files

As one of the major disadvantages of the raw muMath software package is a lack of on-line help, it was decided to supply as much on-line help as possible for the ExPress system. It is acknowledged that to some extent muMath's lack of on-line help was rectified by the package CALC-87 muMath enhancements [P12], but this was not available at the outset of the work in early 1988. The CALC-87 on-line help enhancement to the muMath system, however, was of little benefit to the average school pupil. To exemplify this unhelpfulness take the following example from the enhanced CALC-87 muMath on-line help:

"To expand out the numerator of f(x) type:

"EXPAND ( f(x)); <CR>"

"For example, EXPAND ( (X + 2)^2 ) --> X^2 +4 X + 4."

ExPress gives the following help, highlighted in colour:

```
0.A ALGEBRA Sub Command EXPAND
This command will allow you to EXPAND expressions.
Example :-

EXPAND ((X-1)^2) = X^2 - 2X + 1

EXPAND ((3XY - 2Y + 5)^2 / X) = 9 X Y^2 + 4 Y^2 - 20 Y + 25 - 12 Y^2 + 30 Y
   ___  ___  ___
   X   X   X
```
In order to produce a quick and easily built help system for Express, ordinary text files containing the help were constructed using an ordinary wordprocessor. It was the objective to be able to supply help of the correct nature at any point within Express. To a large degree this was achieved by employing the concept of context-sensitive help. Text files were considered to be the easiest option so that any familiar wordprocessor / text editor could be use by colleagues to alter wordings to suit their own individual styles of teaching and wording.

Within the text files can exist meta-characters which Express interprets as colour codes for highlighting important phrases, these characters are:-

<table>
<thead>
<tr>
<th>Character</th>
<th>Foreground colour</th>
<th>Background colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>'@'</td>
<td>Yellow</td>
<td>Black</td>
</tr>
<tr>
<td>'-'</td>
<td>Red</td>
<td>White</td>
</tr>
<tr>
<td>'#'</td>
<td>Green</td>
<td>Black</td>
</tr>
<tr>
<td>'</td>
<td>'</td>
<td>Cyan</td>
</tr>
<tr>
<td>'%'</td>
<td>Blue</td>
<td>White</td>
</tr>
<tr>
<td>'-'</td>
<td>Yellow</td>
<td>Black</td>
</tr>
</tbody>
</table>

So for example:

@Solve the following Equation@

would be displayed in yellow on a black background.
(A2) Pascal code for the Teachers' Utilities.

Colleagues, as stated in Chapter 3, identified many procedures they considered to be of use from any classroom software. These views were taken into consideration when constructing the teachers' utilities section contained within ExPress. Eventually 19 routines were coded. They range from simple system maintenance to graphically investigating how an individual is performing on a particular mathematical concept.

Some decisions deliberately taken while constructing these utility routines forced the responsibility on teachers to learn the names of the routines provided in muMath. The learning was not considered as too arduous a task. A better reference manual for ExPress would speed this learning process and to this end we would recommend readers to Wooff and Hodgkinson book [104]. However, since all the basic information and help is contained within the ExPress on-line help files, only technical information is needed in any reference manual. This is an important point. It is felt that far too much software is produced which requires considerable amounts of time to get up and running that teachers do not care to spend valuable time investigating complex software even though the use of the software may be potentially educationally very good. Hence a minimum of teacher time and effort should be required to use any classroom based
software. Also, given that one of the objectives of the work was to make mathematics teachers aware of Computer Algebra Systems, this muMath learning process seemed an ideal method of ensuring that they started the process of familiarisation. In any future version of ExPress a question authoring system would be provided in order to reduce preparation time to a minimum and negate the necessity of learning the general syntax of muMath mathematical operations.

In addition to an individual pupil monitoring system, the teacher is provided with a mark book of pupils' progress. This material is contained on disk in a special subdirectory of the system, and is only accessible to the teacher via the teachers' utilities sub-system. This file contains a log of every pupil's work and their interaction with ExPress. The file contains the date, the user's name, the number of questions attempted, the number of correct responses, and the time taken.

It is possible, using these utilities, quickly to search and tabulate or graph, thus allowing a teacher to make professional decisions regarding programmes of study for individual pupils by creating individual learning schemes. Such professional decisions are now, even using sophisticated artificial intelligence techniques, unable to be made reliably by computer systems. Thus, a teacher can force an individual learning scheme to be achieved by making the individual:
(a) study the same material as other users but not necessarily in the same order.

(b) or study a combination of specially prepared material in conjunction with the common material.

Hence, the facility within ExPress exists to allow the teacher to create a scheme of work for any user by simply defining a file containing the names of the question files and the sequence in which they are to be studied. A default order file exists for all users not having their own individual scheme. This mechanism gives the system an individualised feeling to the user and the flexibility for the teacher to set work at the correct level for the individual.

(A3) Communication files.

The final communication link between ExPress and muMath was achieved by always making ExPress write a MuSimp program file called TEST.ALG containing the current question. In this file muMath was instructed to output the result of its evaluation to a file called MUMATH.OUT which ExPress then reads and interprets before displaying the result of the mathematical operation. Below is an example of the file TEST.ALG interactively created by ExPress for muMath to read:-
3 Communication between Sub-Systems

ECHO : FALSE$

FUNCTION BREAK (BREAK,#BRK),
ENDFUN$

WRS('MUMATH,'OUT , C , TRUE);

EXPAND((x-1)^2);

ECHO : FALSE$

SYSTEM(1);

Here the user has asked ExPress to expand the expression

\[ (x - 1)^2 \]

and muMath would output the result

\[ @: MUMATH \]
\[ ? \]
\[ ? \]
\[ EXPAND((x-1)^2); \]
\[ ? \]
\[ @: 1 - 2x + x^2 \]
\[ ? \]

ECHO : FALSE$

which ExPress would decipher to display the result as:-

\[ x^2 - 2x + 1 \]

The PASCAL code relies on the fact that the output format of the muMath evaluation is of a constant format.
Finally, the objective of embedding a teaching strategy was achieved by providing a mechanism for displaying, in a window, a list of mathematical concepts. The concepts appertain to the current question and the user is able select the single concept which they think has been used to solve the current question. The PASCAL code to achieve this concept selection is minimal, but the addition of a mathematical explanation of a concept in the form of an example required considerable expansion to this sub-system. So finally the fourth sub-system is now discussed.

(A4) Specialist teaching files containing teaching material.

The files and routines described up until now only transform muMath into a more user friendly system. The following four sets of specialist files help to transform the adapted and enhanced muMath system into ExPress, the useful classroom computer algebra system. ExPress uses additional files which contain:

1. Questions.
2. Concepts.
3. General help files.
4. Remedial help files.

1. The question file contains questions all of the same type. Any number of files can exist, with any number of questions in each file.
3 Communication between Sub-Systems

2. The concept file contains the likely actions/concepts taken by ExPress in order to solve the problem. Only one the action description is correct.

3. The general help file contains a worked example of the question type applying the appropriate descriptor from the concept file.

4. The remedial file gives a complete worked example of the type the user is trying to solve.

Therefore each type of question has four files. For example to get ExPress to teach pupils how to

\[ \text{SOLVE } 3x - 12 = 9 \]

the four files that need to exist might be called

\[ \text{SOLVE.QUS, SOLVE.OVL, SOLVE.RHL, SOLVE.HLP.} \]

The SOLVE.QUS would contain questions of the type we wish pupils to learn how to solve.

The SOLVE.OVL would contain a selection of 6 or 7 relevant concepts which muMath could have used when it solved the problem.
3 Communication between Sub-Systems

The SOLVE.RHL is directly linked to the SOLVE.OVL file. SOLVE.RHL contains a one screen page example of output for ExPress to display corresponding to the concept current being selected by the user from the concept file.

The SOLVE.HLP is a complete, one screen, worked example showing the correct concept applied to the question and the output the user should expect from ExPress.

All the material used in the teaching experiment is contained in the separate support material document A. This document shows the contents of all these text files and how they are to be designed.
Chapter 6 — The Menu System

1 Introduction

The ExPress software system is a complex collection of components including many hundreds of routines and a large number of supporting files. The whole system is co-ordinated through a pull down menu whose function is to provide a friendly easy to use interface for pupils and to support the teaching strategy. It is clearly important that the structure of the pull down menu should be correct, but it is equally important that the structure should be easily adjusted. Therefore, it was decided to specify formally the pull down menu in order to fully understand the nature of interactions between sub-systems and to fully specify each component. Whilst such an approach is good software practice and should be the basis of all software engineering, there is always the need to prototype software (to get a feel for how the software might behave and to try out ideas before investing too heavily in rigorous development techniques). It turned out that the activity of formally specifying the pull down menu did influence the final design of the software.

In addition a formal specification of a piece of software is of enormous benefit in documenting a system and supports future maintenance of the system.

Therefore, this chapter describes the concept of an abstract data type called Pull Down Menu and how it has been
developed from the underlying model of an n-ary tree. The abstract data types are both formally specified.

The Pull Down Menu (PDM) can be visualised as consisting of a large number of boxes or windows. Each window has its own name. Any window can contain the names of other windows or the name of the empty window. The empty window is the termination or leaf node of a particular path through the pull down menu. Once the empty window is encountered, while traversing the structure, then it is an indication that the user requires a specific operation to be executed.

A window, in addition to its name, has a further set of associated properties:

- Positional data - the screen co-ordinates of the top right-hand corner.
- Window Item Count - the width in characters & the length in lines.
- BorderType - Either single or double.
- Colours - Background, Border, Highlight & Text.
- Names - A list of the names of other windows associated with this window.

From the diagram overleaf it can be clearly seen that the ExPress window holds the names of five windows Edit,
Operation, Special, Files and muMath. Taking the operation window it contains the names of further windows; whereas other windows are empty and terminate (Algebra). A full structure for ExPress is given in Appendix 4.

The structure of the windows is probably better viewed schematically as in the following diagram:
The specification of a pull down menu as an abstract data type has been visually demonstrated to be based upon the underlying model of an n-ary tree. Hence, before formally defining the specification of a PDM, a brief summary of the salient points of this underlying specification of an n-ary tree would be of value. For a fuller description and more formal treatment of the specification of an n-ary tree see [OU M353 Unit 12].

The Abstract Data Type N-ary Tree.

Before proceeding to define the abstract data type Pull Down Menu, the definition of the underlying structure, n-ary tree, is discussed. The n-ary tree definition is well documented in the literature, see P. Thomas et al [87], and rather than reproducing the entire formal definition in full,
a shortened more narrative version is given in order to assist with and serve as a clarification for the Pull Down Menu definitions.

NAME:

N - ary Tree

SETS:

N : The set of all N-ary trees.
I' : The set of all items.
B : The set of Boolean values true or false.
E : The set of all error messages.

OPERATIONS:

(i) CreateNaryTree \(\rightarrow N\)
(ii) Data \(N \rightarrow I \cup E\)
(iii) LeftMost \(N \rightarrow N \cup E\)
(iv) RightMost \(N \rightarrow N \cup E\)
(v) Next \(N \rightarrow N \cup E\)
(vi) AddNode \(N \times I \rightarrow N\)
(vii) Parent \(N \times N \rightarrow N \cup E\)
(viii) IsEmptyTree \(N \rightarrow B\)
(ix) RemoveTree \(N \times N \rightarrow N \cup E\)
(x) FindLeaf \(N \times I \rightarrow N \cup E\)
(xi) Find \(N \times I \rightarrow N \cup E\)
(xii) Isin \(N \times I \rightarrow B\)
(xiii) UpDateData \(N \times I \times I \rightarrow N \cup E\)

140
The semantics of these operations will be described informally rather than formally as they are the underlying model rather than the model we are defining.

SEMANTICS:

(i) **CreateNaryTree**:

  takes no source data and returns an empty n-ary tree.

(ii) **Data**:

  takes an n-ary tree as its source data and returns the item in the root node, or an error if the root is empty.

(iii) **LeftMost**:

  takes an n-ary tree as its source data and returns the left most sub-tree or an error if no such tree exists.

(iv) **RightMost**:

  takes an n-ary tree as its source data and returns the right most sub-tree or an error if no such tree exists.

(v) **Next**:

  takes an n-ary tree as its source data and returns the sub-tree whose root
node is the next child node of the root of the source data sub-tree or an error if no such n-ary tree exists.

(vi) AddNode:

takes an n-ary tree and an item as its source data and returns a new n-ary tree with the source data item added as the rightmost child of the source data n-ary tree.

(vii) Parent:

takes two n-ary trees as source data, the second source tree must be a sub-tree of the first. The resulting n-ary tree has as its root node the parent node of the second source tree, or an error message if the operation is unsuccessful.

(viii) IsEmptyTree:

takes an n-ary tree as the source data and returns true if the n-ary tree is empty otherwise it returns false.
RemoveTree:

takes two n-ary trees as source data and removes the one second specified n-ary tree from the first and returns a new n-ary tree or an error if the operation fails.

FindLeaf:

takes an n-ary tree and an item of data as source and returns the root n-ary tree only as its result or an error. A left to right search of the sub-trees is made to find the leafnode.

Find:

takes an n-ary tree and an item as its source data and recursively searches the n-ary tree for the item. The result is an either an error if the item is not found or an n-ary tree with the root containing the source item.

Isin:

takes an n-ary tree and an item as its source data and recursively searches the n-ary tree for the item. The result is true if the item is found otherwise the result is the false.
UpDateData : 

takes a n-ary tree and two items as  
source data. It returns a new n-ary tree  
where the first data item has been  
replaced by the second data item.

Pull Down Menu Abstract Data Type Definition

Using the method described in P. Thomas et al. [87], the 
Pull Down Menu can be defined as an abstract data type. The 
definition of a Pull Down Menu will be in terms of the 
operations inherited from the generic n-ary tree data type. 
Hence the formal definition of a pull down menu is given in 
terms of four components:

A NAME.

The SETS of objects manipulated by the operations. 
The SYNTAX of the operations. 
The SEMANTICS of the operations.

NAME :

PDM (Pull Down Menu)

SETS :

M : The set of all Pull Down Menus. 
B : The set of Boolean values consisting of TRUE or FALSE. 
E : The set of all Error Messages.
The set of all Windows.

WN : The set of all Window Names.

ON : The set of all Operation Names.

N : The set of all Names ( N = WN UNION ON ).

D : The set of all Display Information.

O : The set of all operations.

A Window consists of a window name n ∈ WN, display information d ∈ D and a list of Names (LIST [ N ]).

An Operation consists of an operation name n ∈ ON and a "code". The underlying data items are therefore an ordered triple (n, d, LIST [N]) for a window and an ordered pair for an Operation. Since two data items form lists, we will need to perform some well known list operations such as IsEmptyList, AddToList and RemoveFromList. These are assumed to be so well known not to require formal definition (details can, however, be found in [87]).

SYNTAX:

Operations on Pull Down Menus

(i) CreateMenu : W --> M
(ii) IsEmptyMenu : M --> B
(iii) AddWindow : M x W --> M Union E
(iv) AddOperation : M x O --> M Union E
(v) DeleteWindow : M x W --> M Union E
(vi) DeleteOperation : M x O --> M Union E
2 Pull Down Menu - The Abstract Data Type N-ary Tree.

(vii) FindWindow : $M \times WN \rightarrow M \cup E$
(viii) MenuData : $M \rightarrow W \cup E$

Operations on Windows

(ix) ChangeDisplayInfo : $W \times D \rightarrow W$
(x) CreateWindow : $WN \times D \rightarrow W$
(xi) GetWindowName : $W \rightarrow WN$
(xii) GetDisplayInfo : $W \rightarrow D$
(xiii) AddNameToList : $W \times N \rightarrow W \cup E$
(xiv) DeleteNameFromList : $W \times N \rightarrow W \cup E$
(xv) GetList : $W \rightarrow LIST[N]$
(xvi) GetOperationName : $O \rightarrow ON$
(xvii) Execute : $N \rightarrow$

SEMANTICS:

With the constructive method of defining abstract data types both a pre- and post-condition specification is needed. The pre-condition specifies, for the given operation, what must hold before the operation can be carried out. The post-condition states what will be true once the operation has been carried out and relates the result of the operation to the input data. The variables used in the abstract data type are defined in terms of the set from which they come, hence $m \in M$; $o \in O$; $n \in N$; $d \in D$ and $r$ is the result of the operation. The operations then become:-
Operations on Menus

CreateMenu : W —> M

Pre>CreateMenu(w) ::= True
Post>CreateMenu(w;r) ::= 
  r = AddNode(CreateNaryTree,w)

IsEmptyMenu : M —> B

Pre>IsEmptyMenu(m) ::= True
Post>IsEmptyMenu(m;r) ::= 
  r = ( m = CreateNaryTree)

AddOperation : M x O —> M Union E

Pre>AddOperation(m,o) ::= True
Post>AddOperation(m,o;r) ::= 
  IF IsEmptyMenu(m) THEN
  r = Error Empty Menu
  ELSE
  r = AddToList(GetList(MenuData(m)),
                GetOperationName(w))
AddWindow : M x W  --> M Union E

Pre-AddWindow(m,w) ::= True
Post-AddWindow(m,w;r) ::=  
IF IsEmptyMenu(m) THEN  
r = AddNode(m,w)
ELSE
IF IsInList(GetWindowName(w),GetList(MenuData(m))) THEN
  r = Error - Name already exists
ELSE
  r = (AddNode(m,w) and
       AddToList(GetList(MenuData(m)),GetWindowName(w)))

DeleteWindow : M x W  --> M Union E

Pre-DeleteWindow(m,w) ::= True
Post-DeleteWindow(m,w;r) ::=  
IF IsEmptyMenu(m) THEN
  r = Error No Window to Delete
ELSE
IF IsEmptyList(GetList(MenuData(m))) THEN
  r = Error No Windows to Delete
ELSE
  r = RemoveNaryTree(m,createmenu(w)) and
       RemoveFromList(GetListData(MenuData(m),
                                   GetWindowName(w)))
DeleteOperation : \( M \times O \rightarrow M \cup E \)

\[
\begin{align*}
\text{Pre-DeleteOperation}(m,o) & : = \text{True} \\
\text{Post-DeleteOperation}(m,o;r) & : = \\
& \quad \text{IF IsEmptyMenu THEN} \\
& \quad \quad r = \text{Error Empty Menu} \\
& \quad \text{ELSE} \\
& \quad \quad \text{IF IsEmptyList(GetList(MenuData(m))) THEN} \\
& \quad \quad \quad r = \text{Error No Operations} \\
& \quad \quad \text{ELSE} \\
& \quad \quad \quad r = \text{RemoveFromList(GetList(MenuData(m)), GetOperationName(o))}
\end{align*}
\]

FindWindow : \( M \times WN \rightarrow M \cup E \)

\[
\begin{align*}
\text{Pre-FindWindow}(m,n) & : = \text{True} \\
\text{Post-FindWindow}(m,n;r) & : = \\
& \quad \text{IF IsEmptyMenu THEN} \\
& \quad \quad r = \text{Error Empty Menu} \\
& \quad \text{ELSE} \\
& \quad \quad r = \text{Find}(m, \text{CreateWindow}(n,d))
\end{align*}
\]
MenuData : M ---> W Union E

Pre-MenuData(m) ::= True
Post-MenuData(m;r) ::= 
    IF IsEmptyMenu(m) THEN 
        r = Error Empty Menu 
    ELSE 
        r = DataNaryTree(m)

Operations on Windows

ChangeDisplayInfo : W x D ---> W

Pre-ChangeDisplayInfo(w,d) ::= True
Post-ChangeDisplayInfo(w,d;r) ::= 
    r = CreateWindow(GetWindowName(w),d)

CreateWindow : WN x D ---> W

Pre-CreateWindow(n,d) ::= True
Post-CreateWindow(n,d;r) ::= 
    r = (n, d, CreateList)
2 Pull Down Menu - The Abstract Data Type N-ary Tree.

GetWindowName : W --> WN

Pre-GetWindowName(w) ::= True
Post-GetWindowName(w;r) ::= 
   r = GetFirst(w)

GetDisplayInfo : W --> D

Pre-GetDisplayInfo(w) ::= True
Post-GetDisplayInfo(w;r) ::= 
   r = GetSecond(w)

AddNameToList : W x N --> W Union E

Pre-AddNameToList(w,n) ::= True
Post-AddNameToList(w,n;r) ::= 
   IF IsInList(GetList(w),n) THEN
      r = Error Name Already in List
   ELSE
      r = AddToList(GetList(w),n)
DeleteNameFromList : \text{W} \times \text{N} \rightarrow \text{W Union E}

\begin{align*}
\text{Pre-DeleteNameFromList}(w,n) & := \text{True} \\
\text{Post-DeleteNameFromList}(w,n;r) & := \\
& \text{IF} \ \text{IsInList}(\text{GetList}(w),n) \ \text{THEN} \\
& \quad r = \text{RemoveFromList}(\text{GetList}(w),n) \\
& \quad \text{ELSE} \\
& \quad r = \text{Error Name Not in List}
\end{align*}

GetList : \text{W} \rightarrow \text{LIST [N]}

\begin{align*}
\text{Pre-GetList}(w) & := \text{True} \\
\text{Post-GetList}(w;r) & := \\
& r = \text{GetThird}(w)
\end{align*}

GetOperationName : \text{O} \rightarrow \text{ON}

\begin{align*}
\text{Pre-GetOperationName}(o) & := \text{True} \\
\text{Post-GetOperationName}(o;r) & := \\
& r = \text{GetFirst}(o)
\end{align*}
Execute : N -->

Pre-Execute(n) ::= True
Post-Execute(n;r) ::= 
   IF IsOperatorName(n) THEN
       r = ExecuteACommand(n)$
   ELSE
       r = DisplayWindow(n)

$ Some of the commands specific to the ExPress application
are as follows:--

<table>
<thead>
<tr>
<th>Name</th>
<th>Action / Routine</th>
</tr>
</thead>
<tbody>
<tr>
<td>History</td>
<td>Show Last 30 user actions</td>
</tr>
<tr>
<td>Screen</td>
<td>Output directed to CRT</td>
</tr>
<tr>
<td>Printer</td>
<td>Output directed to Printer</td>
</tr>
<tr>
<td>Disk</td>
<td>Output directed to Disk</td>
</tr>
<tr>
<td>Utilities</td>
<td>Execute the Teachers Utilities</td>
</tr>
<tr>
<td>DOS</td>
<td>Exit to the Operating System</td>
</tr>
<tr>
<td>Evaluate</td>
<td>Evaluate an Expression</td>
</tr>
</tbody>
</table>

To help the reader a more descriptive definition of each
of the operations is now given.
CreateMenu:

takes no source data and returns an empty menu.

IsEmptyMenu:

takes a menu as the source data and returns true if the menu is empty otherwise it returns false.

AddWindow:

takes a menu as the source data and adds a window and gives it a name and positional information.

AddOperation:

takes a menu and a operation as source data. If the operation does not exist then AddOperation adds the operation as a node to the menu structure and returns a new menu, otherwise it returns an error.

DeleteWindow:

takes a menu and a window as source data. If the window exists then DeleteWindow removes the window from the menu structure and returns a new menu, otherwise it returns an error.

DeleteOperation:

takes a menu and an operation as source data. If the operation exists then DeleteOperation removes the operation from the menu structure and returns a new menu, otherwise it returns an error.
FindWindow:

takes a menu and a window name as source data. The menu is recursively searched for the window name. The result is either an error if the window cannot be found or a new menu with the window at the root of the menu structure.

MenuData:

takes a menu as source data. The result is an error if the menu is empty, otherwise the data at the root of the menu structure is returned.

ChangeDisplayInfo:

takes a window with the associated positional information and some new positional information as source data. It returns a new window.

CreateWindow:

takes a window name and positional information as source data. It returns a new window.

GetWindowName:

takes window as source data and returns the name of the window.

GetDisplayInfo:

takes a window as source data and returns the positional information of the window.
AddNameToList:

takes a window and a window name as source data. If the name is already used by another window then an error is returned, otherwise the name is added to the list of window names.

DeleteNameFromList:

takes a window and a window name as source data. If the name is in the list of window names then it is deleted from the list of names, otherwise an error is returned.

GetList:

takes a window as source data. It returns a list of associated window names.

GetOperationName:

takes an operation as source data and returns the name of the operation.

Execute:

takes an operation name as source data. If the operation is a window name then window is displayed otherwise the operation is executed.

Once the abstract data type has been defined then the coding is a relatively simple task. As the data type
specification suggests the ideal method of coding would be through the use of an object oriented language. Unfortunately, at the time of coding such a language did not readily exist and hence a general block structured language was used. If we were to re-code the pull down menu then almost certainly an object oriented language would be sought.

3 Implementation Details

The Pascal code contained within ExPress to display and drive the pull down menu was conceived initially to drive any pull down menu (PDM). ExPress is just one of many applications taking advantage of the menu system. Hence one ought to consider the philosophy behind the methods employed for implementation of a PDM system, other than those already stated in the aims and objectives of the work, so that the pull down menu system will have the ability to interface with many applications. Most software currently being designed to execute on personal computers uses a PDM of some sort and this was considered the best approach to use in the classroom. One of the main reasons for this decision was the need to reinforce pupil familiarisation of such a widely used interface. The decision to create our own user interface for muMath also had implications on the generality of the PDM implementation.

What was the point of designing and coding a system only applicable to this one application?
Why shouldn't other users writing software for the classroom be able to take advantage of pre-written code to drive their systems?

Hence the code for the PDM was designed to be as invisible as possible, as flexible as possible and as easy to call in other users applications. No ExPress specific routines were to be designed or implemented in this part of the code.

Routines were therefore developed to CREATE, EDIT, PRINT, DISPLAY and ANALYSE pull down menus based upon the abstract data definition. Eventually the routines were used to develop a pull down menu as an interface to themselves so that they were a more professional package for third party users. The resultant pull down menu creation package was used to develop the current ExPress pull down menu system and has subsequently been used by colleagues to create menus for their own software.

The information formed by the menu creation package, while building the pull down menu for ExPress, exists in two files, as it does for all menus developed by the package. One file contains the textual information of all the windows and the other the control information (i.e. screen positioning information). The reasoning behind two files was that the textual information could be changed using an editor whereas the control information was assigned to records which the
user should not need access. When a user requests an action from the PDM a selected node returns a value to the calling routine indicating that either a node has been reached or that a further window should be displayed. With the exception of the root window which has no pointer backwards to a previous window all other windows are owned by a previous parent window, hence backwards and sideways traversal is possible.

The driver routine, having been called by the user's application, only returns when a node is selected. It is therefore possible, by the use of the cursor keys or mouse, to move around from window to window and within a window and it is possible to move back to a window's parent without selecting a node in the present window. The driver routine also has some hotkeys which allow faster, more direct movement around the menu. For example, pressing the F3 Function key will take the user immediately back to the root window, text line one.

Some of the technical aspects of the personal computer also forced decisions to be made on how the pull down menu system would be implemented. As the screen of a personal computer is memory mapped and there was a relatively limited amount of main memory available, the data structure selected for the screen display of each window was a dynamic linked list. It is dynamic in that, as the user progresses up and down the pull down menu, a screen memory dump is copied to
the main memory heap and erased as, and when, appropriate. Hence the previous screen images are resident in main memory. This has the advantage that the image can be quickly block moved back to and from the screen mapped memory and the main memory pointer adjusted accordingly thus giving an instantaneous display effect rather than waiting for the screen to be refreshed. Consideration was given to the implementation of the screen menu structure by virtual memory techniques but this is slow because of the constant amount of disk activity. Unfortunately, this made ExPress unsatisfactory to use and so the former method was selected even though it had main memory implications. The dynamic linking of windows allows maximum flexibility in the handling of both the pull down menu and the screen memory.

It was a requirement, because of the limitation imposed by the operating system to 640K of main memory and the heavy usage of both ExPress and muMath of this memory, that the pull down menu code should be as compact as possible. This was achieved when the coding of the abstract data type took place as no memory optimisation could be undertaken on the already existing muMath system.

The objective of having a common menu for all levels of ExPress users also prompted a slight coding modification but a very significant and worthwhile change to the pull down menu system display. The resulting modification was a direct consequence of the need to have only certain nodes of the
menu active for particular levels of users. This was achieved by a table indicating which nodes are active for the current user. Nodes which are unavailable to the current user are displayed in a different colour and are not able to be selected by the driver routine. This is equivalent to the 'greying' of inactive nodes in other menu systems.

The formal definition of the abstract data type allows many other pull down menu systems to be derived from the single generic model. This is the purpose and power of such methodologies. The implementation permits users to spawn their own menu systems with the minimum of effort.

When viewed against commercially produced menu systems, such as those used in the Borland and Microsoft products, the pull down menu for ExPress performed favourably. It should also be noted that the specification, through the use of the abstract data type definition, has a firm foundation in computer science methodology; and if it were to become necessary, an equally favourable or better implementation could be coded when more appropriate and superior programming languages become available.
Chapter 7 — The Experiment

Design and Implementation

Having developed ExPress, the computer algebraic ExPression teaching system, its effectiveness needed to be tested. A number of possibilities arose as to the best method of evaluating how successful ExPress could be in a mathematics classroom. The evaluation of ExPress could either be envisaged as a full class trial under normal everyday circumstances or as a limited single pupil trial. As one of the original concepts behind the work is to introduce a computer algebra system into the classroom it was decided to trial ExPress under normal everyday conditions with a complete class.

While one of the major objectives of writing ExPress was to make an unfriendly computer algebra system useful in the classroom it seemed only natural to use ExPress with a whole class with as little teacher intervention as possible. Even an unfriendly muMath or ExPress system could be made friendly with a considerable amount of teacher input to a small group gathered around a computer. In this scenario the computer is the observer for most of the session and the teacher is performing his/her normal function of teaching, with the aid of a computer, rather than the computer doing the work and the teacher observing. The intention of the experiment is not to make the computer the observer, rather it is to observe how well ExPress could perform on its own. This would then release the teacher and give flexibility and control to the pupil for
speed, and to some degree the direction, of their learning.

Also, a selective small group experiment might be viewed as an invalid test of the usefulness and worth of ExPress, given the original concept of being able to use ExPress with a full class. A selective experiment could also devalue the usefulness of ExPress if the experiment could be challenged and shown that ExPress depends upon a large amount of teacher time. It is worth noting that a number of other successful research experiments [3,5,63] have only used a few pupils to test their systems. However, it is realised that the purpose of their research was not to test the system itself. These experiments were conducted in a semi-artificial manner in that the researchers directed and controlled the use of their software. Useful classroom software should be, as far as possible, reliable, effective, self reliant and need little or no help from the teacher.

Hence, in conclusion, the practice of testing classroom software with unreasonable amounts of aid from a teacher seems of dubious worth. A system designed for classroom teaching cannot be effectively evaluated when used with a small group of pupils in a tightly controlled environment with constant supervision.

So, to test ExPress a class of 15 year olds were selected from a local comprehensive school. The main aim was to use ExPress as a normal part of the day to day delivery of the
1 Design and Implementation

curriculum in an effort to enhance the teaching of algebra. The system was to be allowed to stand or fall by itself. No special teacher explanation, aid or help was to be given to the users - except for the initial introduction of which keys to press to move around the Pull Down Menu system.

Two relevant areas of the national curriculum document (March 1989) [L2] were chosen for the ExPress system to teach to the pupils. The two topics were selected from attainment target 6 levels 6 and 8, that is, those applicable to national curriculum key stage 4 (14 - 16 year olds). The statements are :-


In the revised national curriculum document (May 1991) [L3] the two selected topics remain at levels 6 and 8 respectively.

2 Resources

ExPress has been developed so that children can be given the opportunity to use a computer algebra system in a structured learning environment. The system is designed to hide the algebraic manipulation so laboriously practiced in many classrooms and concentrate on the underlying mathematical concepts. In order to test the system's teaching effectiveness
it was proposed to conduct an experiment with the clear objective of:

Testing the effectiveness of the ExPress system and isolate as many background factors as possible.

The conduct of the experiment.

In the interests of a balanced design, the class selected was split into two roughly similar groups, called group A and group B as follows. Each group contained equal numbers of pupils, paying attention to the distribution of the sexes as far as possible and with an informal regard to ability, however they were all taken from one appropriate mathematics set. The whole class were told that the next two topics will be one week modules each and that one of the topics would be taught to them by computer.

The computer package used was ExPress running on three Amstrad 1512 machines, with a hard disk and a colour monitor. A printer was available to print a reward in the form of a certificate after each session, (see Appendix 3).

Before beginning the work both groups were pre-tested to ascertain their current level of knowledge of that week's topic. Each group then studied the same topic in the same classroom with the same teacher and the ExPress system. While group A were studying the work on solving linear equations on
the computer, without teacher aid, group B performed the same work by pencil and paper methods with the aid of the teacher. The same questions from the currently used textbooks [L4] were solved by both groups. At the end of the week each group were given the same post-test to ascertain their level of achievement.

In the second week of the experiment group A studied **Factorisation** by pencil and paper methods, and group B undertook the same work at the computer with the ExPress system. Again both groups answered the same questions and took the identical pre-test and post-test.

The content of the two topics was completely determined by the textbook hence the knowledge taught contained questions of the following types:

<table>
<thead>
<tr>
<th>Solving Equations</th>
<th>Factorisation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Factor is</strong></td>
<td></td>
</tr>
<tr>
<td>$x + 1 = 9$</td>
<td>$2x ± 6$</td>
</tr>
<tr>
<td>$x - 2 = 10$</td>
<td>$3xy ± 4px$</td>
</tr>
<tr>
<td>$2x = 8$</td>
<td>$3xy ± 6xz$</td>
</tr>
<tr>
<td>$x - 5 = 0$</td>
<td>$3xy ± 3x$</td>
</tr>
<tr>
<td>$2x ± 9 = 12$ and</td>
<td></td>
</tr>
<tr>
<td>$4x ± 10 = 3x ± 9$</td>
<td></td>
</tr>
</tbody>
</table>
2 Resources

At the end of the experiment each pupil was informally questioned and asked to comment on their experience of being taught by ExPress. The questions all related to their feelings about the ExPress system and their use of the system.

The times taken to solve the sets of exercises by pencil and paper method were noted for comparison with the times taken by those working with ExPress. Time comparisons to some extent will not be truly reflective of any fundamental features as ExPress requires only the selection of the correct concept from a list in a menu box whereas the pencil and paper methods usually culminate in a complete worked solution after many intermediary steps. However, it was hoped that trends may be discernible when the timings are analysed.

The pre-tests and post-tests were marked by the teacher as were the pencil and paper workings of the two groups. While the pupil is working with the ExPress system the marking and assessing of the users performance takes place automatically and the results recorded in a file on the hard disk. The individual pupil files give a complete profile of the pupils' interactions with ExPress as both numbers of correct and incorrect responses are recorded.

3 Experiment Results

The primary aim of the experiment was to evaluate the use of ExPress in the classroom. A secondary aim of the experiment was to evaluate the effectiveness or ineffectiveness of the
contribution made by ExPress to the learning of algebra. The learning or memory retention of the skills taught were tested over a short time period, less than one week.

The statistics obtained from the experiment are fully detailed in Appendix 1. Tables 1-3 are the raw results for group A solving equations using ExPress in week 1. Tables 4-6 are the raw results for group B solving equations by paper and pencil methods in week 1. Tables 7-9 are the raw results for group B solving factorisations using ExPress in week 2. Tables 10-12 are the raw results for group A using the paper and pencil methods in week 2 to factorisations. All these results are combined and analysed in tables 13-17. This can be best explained with reference to the following grid:

<table>
<thead>
<tr>
<th></th>
<th>Week 1</th>
<th>Week 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A</td>
<td>Solving equations using ExPress</td>
<td>Factorising using pencil &amp; paper</td>
</tr>
<tr>
<td>Group B</td>
<td>Solving equations using pencil &amp; paper</td>
<td>Factorising using ExPress</td>
</tr>
</tbody>
</table>

Looking at tables 13 and 14, the results synopsis for each group respectively, the solving of equations took approximately the same overall time to be completed whether performed by paper and pencil or by using ExPress. This contrasts markedly with the comparable situation for factorisations. There are two main reasons for this apparent time contradiction:

(1) 162 questions fewer were attempted by the group using paper and pencil methods. At an average of 1:06 minutes per
Experiment Results

question using ExPress, (1:06 minutes comes from calculations made while performing the experiment), gives a value of 162 * 1.1 minutes = 2:58:12 additional hours needed.

(2) The method taught to the paper and pencil group required only a single line of working. This contrasts noticeably to the number of lines of working required for solution of equations. For example:-

Factorise Solve

\[ 2x - 4xy \]
\[ = 2x (1-2y) \]

\[ 6x - 1 = 4x + 5 \]
\[ 6x - 1 + 1 = 4x + 5 + 1 \]
\[ 6x = 4x + 6 \]
\[ 6x - 4x = 4x - 4x + 6 \]
\[ 2x = 6 \]
\[ x = 6 / 2 = 3 \]

If we consider the time taken to solve an equation or a factorisation by ExPress to be roughly equivalent then this explains the apparent anomaly in the timings.

The topic equations had previously been encountered by pupils and as such the test results failed to show any significant improvement regardless as to the teaching method. Both paper and pencil methods and ExPress achieved over 80% success. However there was a noticeable difference, in favour of ExPress, when pupils experienced a new topic, factorisations. Both the exercise average and the post test
averages were higher for ExPress than for pencil and paper given that both groups started from approximately the same state as shown by the pre-test scores.

These illustrations show what we expected from the experiment, in that where the curriculum had been taught in previous years (e.g. solution of equations) ExPress made little or no difference. However in the case where a new topic, factorisation, was being taught for the first time ExPress performed equally as well as the teacher. The computer taught pupils were at no disadvantage having not performed the laborious manipulations.

Pupils taught algebra by a computer algebra system performed as well and certainly no worse than those taught by more traditional methods.

We believe this to be a most significant result and wish to emphasise the point that topics taught through acquisition of an understanding of concepts only and where the delivery mode is a computer algebra package, appear from this work, do not place the pupil at a disadvantage when compared with more traditional methods. This is a surprising result but we claim that this supports and adds to the work of J.Palmiter [63].

Further, by studying individual pupil performances, albeit on two different topics, it can be established that when ExPress taught the pupils a new topic then the difference in
3 Experiment Results

their pre-test and post-test scores were better than when taught by a teacher. An examination of the differences in test scores given in the table below supports this conclusion. The table also shows that ExPress did not damage the learning experience of pupils as none of the differences were negative irrespective of the method of delivery employed. It does, however, appear that ExPress made a better attempt at teaching the new topic when compared with the teacher. This is a dangerous assumption in that we have no knowledge of relative pupil abilities or their personal preferences toward aspects of algebra. All that we can say with absolute certainty is that ExPress did not perform worse than the teacher. A simple comparison of the differences in pre-test and post-test scores of students will help to confirm this assumption.

| Pupil | Computer Topic | Pencil Paper Topic | |
|-------|----------------|--------------------|
| A     | +2  | Not a | +4 | New Topic |
| B     | +2  | new topic | +3 | |
| C     | +1  |         | +5 | |
| D     | +2  | Solution | +4 | Factorising |
| E     | +1  | of     | +1 | of |
| F     | 0   | Equations | +2 | Expression |
| G     | 0   |         | +3 | |
| H     | +8  |         | +1 | |
It is interesting to note that pupils mainly completed computer exercises within a lesson and were enthusiastic enough to spend a large amount of their own time at lunchtime and after school completing exercises. This can also be attributed in part to the novelty value of being taught by a computer and the additional enthusiasm of the teacher. It did take longer to solve questions using ExPress but, as stated earlier, this was due to the relatively poor performance of the hardware given the huge demands made by ExPress on it.

Having tested the achievements of the pupils and analysed the outcomes we need to be certain that the tests were fair and the populations of equivalent ability. The statistics in the second half of Appendix 1 seek to show that the experiment was soundly based.
The first part of the statistical analysis establishes that the variances and means of the pre-test results are equal and hence the two groups can be considered to have approximately the same ability thus forming an homogeneous population. The second part of the statistical analysis evaluates the post-test results and establishes a significant difference in the results for factorising when taught by computer. This increase in result can only be attributed to ExPress as other factors such as ability have been shown to be statistically insignificant across the two groups.

4 Follow up Work

It would be all to easy to conclude that ExPress was a total success and that no further work needs to be undertaken. This would be a naive view. ExPress is not the end of the line, whereas in my opinion muMath could be regarded as so for the present; if we take the production of Derive as an indication then muMath is also regarded by Soft Warehouse as having a limited life. The process of learning from such experiments while searching for better, clearer and more natural methods of teaching algebra by computer is only just evolving. So the follow up to this work should be at least twofold.

Systems such as MACSYMA, Maple and Reduce have had extended use for teaching in undergraduate education. As already stated most university departments now use computer algebra systems with their students and the practice is evolving at an ever
increasing pace. So, a longer more detailed use of ExPress, or a similar system, should be undertaken to ascertain the effectiveness of a computer algebra teaching system in secondary education. The research should aim to see if more pupils of average or below average ability would benefit from using such a system.

As a secondary aim it would be worthwhile to gauge the effectiveness of teaching algebra by computer on the whole ability range. This would probably entail the production of a system geared specifically to the requirements of current mathematical syllabuses. Additionally a longer term experiment of the practicalities of teaching all algebra by computer through the teaching concepts only and ignoring repetitive tiresome manipulations.

ExPress is not the end of the line in the search for a system based on the powerful algebraic systems in existence. Secondary education does not need larger, more powerful, quicker or the latest algebraic algorithms. Hence for the immediate future systems based on muMath are more than adequate. What should be investigated is ways of presenting the mathematical intelligence or knowledge contained within a computer algebra system in a manner acceptable to a larger audience in as clear and as simple way as possible.

A more detailed exposition of future work can be found in Chapter 8, the conclusion.
1 Resulting Implications

In general research into the use of CAS in education is heavily orientated towards university undergraduate and postgraduate level. In Britain what little research that has been undertaken at school level seems to be patchy and uncoordinated. The work at school level has mainly been left to keen individuals rather than co-ordinated, well funded projects. An example of the keen individual approach take the work of D.Tall [L5] in a series of articles in Mathematics in School from the Mathematical Association. In complete contrast, a considerable body of research exists in to how pupils learn algebra and the diagnosing of their common algebraic misconceptions as reported by Brown and Burton [6]. What is needed now is the co-ordination and unification of both research into computer algebra systems and the learning of mathematics. The fundamental complexities of both research areas suggest this combined approach is not likely to take place for a considerable time.

General computer algebra systems such as Derive, Maple and Mathematica all have advantages over the early specific systems such as SCHOONSHIP, CAMAL and ASHMEDEAI. However, none of these systems were designed with the school population in mind. Computer algebra systems designed primarily for teaching have limited functionally, (see Symbolic Calculus - 1988 Maths Workshop). The very existence of these limited
single operation algebra packages together with the acknowledgement by software designers of their potential usage in school is an indication that the larger more general systems would be used in school.

This research has highlighted three major reasons for the lack of use of CAS in schools are:-

(a) Lack of staff awareness of CAS.

(b) Limited hardware in schools.

(c) Suitability of existing computer algebra systems.

Each of these points will take time, funding and awareness raising if they are to be successfully overcome. Since the work originally began in 1988 there has been widespread awareness raising with press articles in the teaching professions press as recently as January 1993.

The phrase "knowledge is power" is certainly true when applied to computer algebra systems as their capabilities can help shape the pupil's knowledge by influencing and directing what is done and learnt. Surely the inventive use of CAS can help enrich an area of mathematics which is normally taught in cold uninspiring manner either from the textbook or the blackboard?
Referring back to the initial aims and objectives set out in chapter one we are able to subjectively evaluate the success of the project. The work has shown that:-

(a) it is possible to use CAS in the secondary classroom.

(b) it is improve, adapt and extend an existing CAS for the secondary school classroom.

(c) it is the use of CAS for the teaching of basic elementary algebraic concepts is as good as a human being.

(d) the omission of the routine repetitive manipulations associated with learning and understanding of algebraic concepts has no serious short term effects on the pupils' level of understanding.

(e) the time required to use CAS as a method of delivery is not significantly longer than normal pencil and paper methods as demonstrated by the experiment and summarised in the table below:-

<table>
<thead>
<tr>
<th></th>
<th>By Pencil &amp; Paper</th>
<th>By Computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factorise</td>
<td>21:45:36 hrs.</td>
<td>33:51:45 hrs.</td>
</tr>
<tr>
<td>Solving Equations</td>
<td>34:51:45 hrs.</td>
<td>36:45:12 hrs.</td>
</tr>
</tbody>
</table>
A more detailed report of the experiment is contained in [89], 'An experiment in the use of computer algebra in the classroom' by P. Thomas and M. G. Rickhuss [Appendix 10].

ExPress does not give a reasoned solution as MATHPERT does but it does have the advantage of being built upon a well known computer algebra system. Additionally ExPress addresses the aim of algebraic teaching but from a different viewpoint.

The two systems, MATHPERT and ExPress, share a common goal of being primarily designed for use in the classroom. It is this point which particularly sets these two systems apart from all other CAS. They aim to point the way forward for future researchers.

However, not all researchers subscribe to the view that computer algebra systems have a part to play in the mathematics education of pupils. Waits and Demana [96] report against the use of CAS in schools in 'The Case Against Using Computer Algebra Systems'. Their argument is based upon a pupil trying to solve the following example:

'A box with no top and a volume of 30 cubic cms. is made from an 8.5cm - by - 11cm piece of cardboard by removing squares of equal side length from each corner and turning up the sides. Determine the side length of the removed squares.'
Whilst one can agree that this is a reasonable type of question for a student to solve the values selected in the example do result in the three horrendous, almost unintelligent symbolic solutions. One of the roots of the resulting cubic,

\[ x (11 - 2x)(8.5 - 2x) - 30 = 0 \]

is:

\[ x = \frac{-13}{4} - \frac{\sqrt{399 \sin \left( \frac{18 \sqrt{7054995}}{2351665} \right)}}{6} \]

the other two are equally as daunting, (output is from Derive). However, the same question with a single minor alteration to one of the lengths, i.e. 8 cm - by - 11 cm and keeping the volume of 30 cubic centimetres results in the cubic equation

\[ x (11 - 2x)(8 - 2x) - 30 = 0 \]

and solutions of:

\[ x = 3; \quad x = \frac{13}{4} - \frac{\sqrt{129}}{4}; \quad x = \frac{13}{4} + \frac{\sqrt{129}}{4}. \]

(results from Derive).
This is a much more realistic problem, even though the first term contains a factor of 2, for pupils and the output from Derive would be more easily interpreted by most pupils working at this level.

A similar problem with the lengths of 7 cm - by - 11 cm and a volume of 30 cubic centimetres results in the cubic

\[ x (7 - 2x)(11 - 2x) - 30 = 0 \]

and more realistic solutions of:

\[
\begin{align*}
x &= \frac{1}{2}; & x &= \frac{5}{2}; & x &= 6.
\end{align*}
\]

The second point made by Waits and Demana is a much more valid and one that has been highlighted by this research. Schools today do not possess enough desktop computers of sufficient power to run large sophisticated computer algebra systems. Moreover mathematics classes probably have a restricted amount of time allocated thus making regular access difficult. The point is therefore made that inexpensive graphics calculators are the best way forward at present. We can not agree with this view because what is proposed appears to be a 'wait and see' attitude rather a positive, enthusiastic, 'how can we best achieve this' conviction.
A far more positive view is taken in 1984 by Kunkle and Burch \cite{Kunkle1984} in 'The Classroom Computer Takes a Quantum Jump' when hardware was severely limited by comparison with today's standards.

Surely we should subscribe to the view that in our pupils lie the future of mankind and as such we should seek to give them the best possible education. We should therefore make the resources available if research can prove the worth of a teaching style / methodology. Hopefully this work has emphasised the merits of computer algebra systems in the classroom and has converted a small section of the teaching mathematics community into campaigning for the required resources for our children.

2 Future Work

If CAS can been seen to have an effect upon the learning of algebra it seems reasonable to ask questions about applications other than learning. As CAS can be shown to be an aid in individualised learning environments then perhaps assessment by CAS is the next stage. The frequency and style of customised CAS examinations may lead to implications for the rates of learning.

A method of making the input to systems consistent with the way in which mathematicians write mathematics would enhance the performance of computer algebra systems. Work is
in progress in Europe to produce a Mathematics Workbench. The workbench will pay particular attention to the human-computer interface. The limitations of hardware may result in a change to the way in which Mathematicians write Mathematics. The significant point is that the layout is important for effective communication.

Mathematics teachers might feel more comfortable with using systems as a means of delivering the syllabus with an improved interface. For example, a method whereby the screen simulates a blank piece of paper (much the same way as a spreadsheet does) and allows the user to position mathematical symbols on the paper may be a more natural interface for a mathematician. As a clarification a procedure whereby the user is presented with a grid (piece of graph paper) on the screen:-

and fills it in as shown to solve an integration problem:-
This is obviously a difficult problem in that minor notation changes, which all mathematicians tend to use, would create extreme interpretation obstacles. The input routines would nevertheless be able to cope with minor deviations such as:

before forming the muMath syntax ' DEFINT(x^2,4,5,x); '. More difficult notation changes would either not be able to be understood, with the inevitable frustration for the user, or would totally confuse the system resulting in the wrong problem being solved.

Another important area for further research concerns the enhancement of the implementation of the teaching strategy.
Improving ExPress to make it intelligent enough to be able to adjust the pace and pathway through question files as a direct result of the users responses to previous questions needs investigation. One question to be answered is 'Should the ability to allow pupils to prematurely complete files based on responses in order to facilitate faster progress and remove unnecessary routine repetitive work' be incorporated. The danger here is that teachers might feel that their professional judgment and expertise are being invaded and eroded.

As a follow up to the experiment it would be interesting to test the same pupils after six, twelve and twenty for months. We would then be able to appraise the long term usefulness of such a package in the mathematics classroom.
References

[1] Algebra made mechanical  

A Tutoring and Diagnostic System for Arithmetic  
University Exeter Paper No.166 (1987)

Experiments with muMath in Austrian high schools  

Logic and computation in MATHPERT  
An expert system for learning mathematics  
editors E.Kaltofen and S.M.Watt

Wherefore "plug-and-chug"?: computer algebra versus  
A-level mathematics  
The Mathematical gazette  
Vol75 No. 471 March 1991 pp40-48

Diagnostic Models for Procedural Bugs in Basic  
Mathematical Skills  
in Intelligent Tutoring Systems Academic Press 1982  
ed. D.Sleeman

Symbolic Algebraic Computation  

[8] B.Buchberger  
CAMP: A teaching project in symbolic computation at the  
University of Linz  

[9] A.Bundy  
The Computer Modelling of Mathematical Reasoning  
Academic Press

Diagnosing bugs in a simple procedural skill  
in Intelligent Tutoring Systems Academic Press 1982  
ed. D.Sleeman

Introducing computer algebra to users and to students  

Introduction in proceedings of EUROCAM '82  
European Computer Algebra Conference. Marseilles  
Lecture Notes in Computer Science 144 Springer Verlag.

186
References

Computer algebra: past and future
in Proceedings of EUROCAL '85

Computer algebra in the undergraduate mathematics classroom
in Proceedings of SYMSAC 1986 Pages 135-140

The design of MAPLE: A compact portable and powerful computer algebra system
in Proceedings of EUROCAL '83 Pages 101-115

[16] C.Cole and S.Wolfram
SMP - a symbolic manipulation program
in Proceedings of SYMSAC '81 Pages 20-23

[17] G.Collins
The SAC-1 System: An Introduction and Survey
Second Symposium on Symbolic & Algebraic Manipulation
(March 1971) Pages 144-152

[18] T.L. Cromer
Some applications of computer algebra
Collegiate Microcomputer (May 1987) Pages 182-188

An Alternative Approach to Learning in Undergraduate Mathematics
In Selected Readings in Computer Based Learning Pages 195-201

The Parallel Risch Algorithm
EUORCAM '82 Pages 145-156

PowerMath - a system for the MacIntosh
in Proceedings of SYMSAC '86 Pages 13-15


[23] J.H.Davenport
Mathematics of Computer Algebra Systems
The Influence of Computers and Informatics on Mathematics and its Teaching 1985 Pages 76-84

[24] E.Dubinsky
Computer Experiences as an aid in learning Mathematics concepts
The Influence of Computers and Informatics on Mathematics and its Teaching 1985 Pages 61-70
Symbolic manipulation programs for the personal computer
Science (3 February 1989) Pages 679-684

[26] J.Fitch
Solving Algebraic Problems with REDUCE
Journal Symbolic Computation (1985 - 1 ) Pages 211-227

[27] C.Fraser
Algebraic manipulation by computer-a sign of things to come?

[28] R.Freese, P.Lounesto and D.Stegenga
The use of muMath in the calculus classroom
Journal of computers in Mathematics & Science Teaching
(1986 Fall) Pages 52-55

[29] M.R.Genesereth
The role of plans in intelligent teaching systems
in Intelligent Tutoring Systems Academic Press 1982
ed. D.Sleeman

Examples of Maple applied to problems from the American Mathematical Monthly

[31] J.Griesmer and R.Jenks
The SCRATCHPAD System
Proceedings of SYMSAC 1971 Pages 266-267

A guide to Computer Algebra Systems Version 1.0
Liverpool University (28 June 1988)

[33] C.M.Hawker

[34] A.C.Hearn
The personal algebra machine
in Information processing 80
Noth-Holland, Amsterdam 1980

[35] A.C.Hearn
Reduce-A case study in algebra system development
EUROCAM '82 Pages 263-272

A Guide to Computer Algebra Systems
Computer Lab. Liverpool University (1988)
References

Introductory Calculus in 1990
ICIM Symposium on Computers and Mathematics 1984

[38] J.M.Hosack
A guide to computer algebra systems
College Mathematics Journal 17-5 (1986) Pages 434-441

Report on the use of a symbolic mathematics system in undergraduate instruction

[40] J.M.Hosack and D.Small
Computer Algebra Systems : Tools for reforming calculus instruction
In Towards a lean and lively calculus: Report of the conference/ workshop to develop curriculum and teaching methods for calculus at the college level
Editor R.Douglas (1986)
Mathematical Association of America

[41] N.Jurkovic
Edusym - educational symbolic manipulations on a microcomputer
Proceedings of SYMSAC '86 Pages 154-156

[42] Z.A.Karian
Getting Started with a Computer Algebra System

[43] J.Kennelly
Calculus classroom experiences with the HP-28C

[44] A.Kent & J.Williams (Eds.)
Symbolic Manipulation Packages in Encyclopedia of Computer Science and Technology
(Supplement 4) pages 364-390 (1988)

[45] D.E.Knuth
The Art of Computer Programming Vol 2, Semi-numerical Algorithms
Addison-Wesley (1981)

Symbolic Computer Algebra : The Classroom Computer Takes a Quantum Jump
Mathematics Teacher March 1984 pp 209-214

[47] K.D.Lane, A.Ollongren and D.R.Stoutemyer

189
[48] F.Lichtenberger
Self-explanatory symbolic computation for math. education

[49] J.D.Lipson
Algebra and Algebraic Computing
Addison-Wesley (1981)

[50] M.MacCullum
Computer algebra-tomorrow's calculator
New Scientist 112 (23 October 1986) Pages 53-55

Machine Learning An Artificial Intelligence Approach
Volume2
Morgan Kaufmann (1986)

[52] A.Miola and I.V. Pottosin
A bibliography of Soviet works in algebraic manipulations

[53] A.Miola
Complexity Issues in algebraic Algorithms: Relevant topics for mathematical education

[54] R.D.Mills
Using a Small Algebraic Manipulation System to Solve Differential and Integral Equations by Variational and Approximation Techniques
Journal Symbolic Computing (1987 - 3) Pages 291-301

[55] J.Moses
Symbolic integration, Ph.D. Thesis
MIT (1967) also Project MAC Technical Report TR-47

[56] J.Moses
Current Capabilities of the MACSYMA System
Proceedings of SYMSAC 1971 pages 263-265

[57] J.Moses
Algebraic Simulation- a guide for the perplexed
Proceedings of SYMSAC 1971 pages 282-304

[58] H.Murakami and N.Hata
Mathematical Education in the Computer Age
The Influence of Computers and Informatics on Mathematics and its Teaching Proceeding of the ICMA 1985 Pages 85-94

[59] E.Ng Editor EUROSAM 1979
Proceedings of the Interantional Collequium on the Methods of Symbolic and Algebraic Computation Marseille 1979

190
[60] A. Ollongren
Classroom experiences with interactive formula manipulation

[61] W. Page
Computer Algebra Systems Issues and Inquiries
Computers Mathematics Applications Vol. 19 No. 6 (1990)
Pages 51 - 69

[62] S. Papert
Mindstorms: Children, Computers and powerful ideas
Harvest Press 1980

[63] J. Palmiter
Effects of computer algebra systems on concept and skill acquisition in calculus
Journal for Research in Mathematics Education Volume 22 Number 2 March 1991 pp151-156

[64] R. Pavelle, M. Rothstein and J. Fitch
Computer algebra
Scientific America (December 1981) Pages 136-146, 151-152

[65] R. Pavelle, P. Wang
MACSYMA from F to G
Journal Symbolic Computation (1985 - 1) Pages 69-100

[66] P. Piddock
Extended muSimp/muMath for teaching and learning mathematics
Computing and Education 10 - 1 (1986) Pages 155-158

[67] A. Ralston
The Really New College Mathematics and its Impact of the High School Curriculum

[68] B. Raphael
The Thinking Computer: Mind Inside Matter 1976

[69] M. Rodino and A. Scarafiotti Abete
Computer Experiments of Mathematical Analysis Teaching at the Politecnic of Torino
The Influence of Computers and Informatics on Mathematics and its teaching (March 1985) Pages 265-267

Derive A Mathematical Assistant (1988)
[71] J.E. Sammet
An Annotated Descriptor Based
Bibliography on the Uses of Computers for
Non-Numerical Mathematics.

[72] J.E. Sammet
Revised Annotated Descriptor Based
Bibliography on the Uses of Computers for Non-Numerical
Mathematics.
In Proceedings of the IFIP Working Conference on
Symbolic Languages
Ed. G. Bobrow (1968)

[73] J.E. Sammet
Survey of Formula Manipulation
In Comm. of ACM 9 pp 555-569 (1966)

[74] J.E. Sammet
Formula Manipulation by Computer
In Advances in Computers Vol 8 pp 47-102
Eds. F. Alt & M. Rubinoff Academic Press 1971

[75] J.E. Sammet
Software for Nonnumerical Mathematics
In Mathematical Software

[76] J. Slagle
A Heuristic Program that Solves Symbolic Integration
Problems in Freshman Calculus

[77] L. Sterling, E. Shapiro
The Art of Prolog - Chapter 9 Pages 135-145
MIT Press

Solving Symbolic Equations with Press
EUROCAM '82 Pages 109-116

[79] W. Squire
Computer Algebra V: The Function Modules
ACCESS May/June 1985 Pages 18-19

[80] W. Squire
Computer Algebra VI: The DIF, LIM and SIGMA Modules
ACCESS September/October 1985 Pages 39-40

[81] D. Sleeman
Assessing aspects of competence in basic algebra
in Intelligent Tutoring Systems Academic Press 1982
ed. D. Sleeman
[82] D.H. Sleeman, R. Steinberg and D. Ktorza
Algebra students' knowledge of equivalence of equations
Journal for Research in Mathematics Education
Volume 22 Number 2 March 1991 pp112-121

[83] D.R. Stoutemyer
PICOMATH-80, an even Smaller Computer Algebra Package
SIGSAM Bulletin 14 -3 (1980) Pages 5-7

[84] D.R. Stoutemyer
A Preview of the Next IBM-PC version of muMath
EUROCAL 85 Volume 1 Pages 33-44

[85] D.R. Stoutemyer
A radical proposal for computer algebra in education

[86] D. Stoutemyer
Using computer symbolicmath for learning by discovery.
The Influence of Computers and Informatics on
Mathematics and its teaching (March 1985) Pages 185-190

[87] P.G. Thomas, Hugh Robinson and Judy Emms
Abstract Data Types (Their Specification, Representation, and Use)

[88] P.G. Thomas and P.A. Leadbetter
Trends in Mathematics Teaching
Sixth International Conference on Technology and Education
(21-23 March 1989)

[89] P.G. Thomas and M.G. Rickhuss
An experiment in the use of computer algebra in the classroom.
Education and Computing

[90] R. Tilidetzke
A Comparison of CAI and Traditional Instruction in a
Computer Algebra Course.
Journal of Computers in Mathematics and Science Teaching
Vol 11 No.1 pp53-62 1992

[91] R.G. Tobey
Algorithms for Antidifferentiation of Rational Functions
Harvard University (1967) Ph.D. Thesis

[92] R.G. Tobey
Symbolic mathematical computation-introduction and overview
Proceedings of SYMSAC 1971

193
[93] Z. Usiskin
Why Elementary Algebra Can, Should and Must be an
Eight-Grade Course for Average Students
Mathematics Teacher September 1987 Pages 428-438

[94] J.A. van Hulzen and J. Calmet
Computer Algebra Systems
in Symbolic and Algebraic Computation Computing
Supplement 4
Springer Verlag (1982) Pages 221-244

[95] J.A. van Hulzen and J. Calmet
Computer Algebra Applications
in Symbolic and Algebraic Computation Computing
Supplement 4
Springer Verlag (1982) Pages 245-258

[96] B. Waits and F. Demana
A Case Against Computer Symbolic Manipulation in School
Mathematics Today.
Mathematics Teacher
Vol 85 No. 3 pp 180-183 1992

[97] J.A. van Hulzen
Computer Algebra Systems viewed by a notorious user
in Proceedings of EUROCAM '82 Pages 166-180

[98] P. Wayner
Symbolic math on a Mac.
BYTE (January 1989) Pages 239-244

[99] J. Weber
Mathematica
Personal Computer World (February 1989) Pages 140-143

[100] J.E. White
Learning by teaching: A novel use for symbolic algebra
in the classroom
CASE Newsletter 4 (June 1988) Pages 3-9

[101] H.S. Wilf
The disk with a College Education
American Mathematical Monthly 89
pp 4-8 (1982)

[102] H.S. Wilf
Symbolic manipulation and algorithms in elementary
calculus
The future of College Mathematics (1983) pp.27-42

[103] S. Wolfram
Mathematica: A system for Doing Mathematics by Computer
Addison-Wesley
1988
[104] C. Wooff and D. Hodgkinson
muMath: A Microcomputer Algebra System

[105] C. Wooff and D. Hodgkinson
Computer algebra systems in physics teaching

[106] J. Wyman
Addition 3 to above reference [73]
SIGSAM bull 10 Oct 1968

[107] J. Wyman
Addition 4 to above reference [73]
SIGSAM bull 12 July 1969

[108] J. Wyman
Addition 5 to above reference [73]
SIGSAM bull 15 July 1970

[109] D.Y.Y. Yun and D.R. Stoutemeyer
Symbolic mathematical computations
in Encyclopedia of Computer Science and Technology 15
Pages 235-310

[110] M.N. Zloof
Query by Example
Proceedings of N.C.C. 44 (May 1975)

[111] P. Zorn
Computer symbolic manipulations in elementary calculus
Mathematical Association of America Notes 6 pp237-249 (1986)
References - Software Products

[P1] MS-DOS (Microsoft Disk Operating System)
[P2] muMath (Computer Algebra Package)
[P3] muSimp (Programming Language)
[P4] Turbo Pascal V 5.5 (Borland Products)
[P5] Derive (Computer Algebra Package)
[P6] MACSYMA (Computer Algebra Package)
[P7] REDUCE (Computer Algebra Package)
[P8] Maple (Computer Algebra Package)
[P10] Mathematica (Computer Algebra Package)
References - Applications

[AP1] J.Calmet and J.A.van Hulzen
Computer algebra applications
in Symbolic and Algebraic Computation Computing
Supplement 4
Springer Verlag (1982) Pages 245-258

[AP2] J.P.Fitch
The application of symbolic algebra to physics
Proceedings of EUROSAM 1979 pages 30-41

[AP3] A.C.Hearn
Applications of symbolic manipulation in theoretical physics
Proceedings of SYMSAC 1971 pages 17-21

Applications of Computer Algebra
Kluwer, Dordecht 1985

[AP5] R.Pavelle
MACSYMA: Capabilities and applications to problems in engineering and sciences
in Applications of Computer Algebra
Kluwer, Dordecht 1985

[L1] B.Raphael
The Thinking Computer: Mind Inside Matter
1976

[L2] Mathematics in the National Curriculum
HMSO Publications
1989

HMSO Publications
1991

[L4] L.Bostock, S.Chandler, A.Shepherd and E.Smith
ST(P) Mathematics 1
Stanley Thornes (Publishers) Ltd

[L5] D.Tall
Series of articles in
Mathematics for Schools Vol 103-112
COMPUTER ALGEBRA SYSTEMS

AND

SECONDARY EDUCATION

APPENDICES

Michael Rickhuss B.A. (Hons).
Department of Computer Science,
The Open University,
Walton Hall,
Milton Keynes MK7 6AA.

A thesis submitted for the Master of Philosophy Degree,
Open University, in June 1993.
Appendices

1 Statistics from the Experiment.......................... 1

2 Functions Added to muMath............................ 33

3 Teaching Materials.................................... 36

4 ExPress Menu Structure................................ 37

5 Names of mathematical operations added to muMath.... 40

6 Table of routines and their syntax..................... 41

7 Algebra content of the 1990 and 1991 GCSE exams..... 44

8 Batch Files............................................. 48

9 Example of teaching materials for the experiment ... 52

10 Published Paper

   An Experiment in the use of Computer Algebra in the Classroom. (1992)
Appendix 1 -- Statistics from the Experiment

This appendix contains the raw results from the experimental use of ExPress in the classroom. They should be viewed in conjunction with the notes contained in Chapter 7 of the thesis. Additionally this appendix contains the statistical analysis of the results.

GROUP A WEEK 1 RESULTS

<table>
<thead>
<tr>
<th>Exercise/Identifier:</th>
<th>Solve 1</th>
<th>Solve 7</th>
<th>Solve 2</th>
<th>Solve 3a</th>
<th>Solve 3b</th>
<th>Solve 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(31)</td>
<td>(31)</td>
<td>(24)</td>
<td>(14)</td>
<td>(41)</td>
<td>(41)</td>
</tr>
<tr>
<td>Pupil A.</td>
<td>30</td>
<td>30</td>
<td>16</td>
<td>9</td>
<td>36</td>
<td>32</td>
</tr>
<tr>
<td>Pupil B.</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>35</td>
<td>37</td>
</tr>
<tr>
<td>Pupil C.</td>
<td>31</td>
<td>31</td>
<td>21</td>
<td>11</td>
<td>40</td>
<td>39</td>
</tr>
<tr>
<td>Pupil D.</td>
<td>24</td>
<td>30</td>
<td>12</td>
<td>9</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>Pupil E.</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>32</td>
<td>38</td>
</tr>
<tr>
<td>Pupil F.</td>
<td>27</td>
<td>30</td>
<td>13</td>
<td>12</td>
<td>30</td>
<td>27</td>
</tr>
<tr>
<td>Pupil G.</td>
<td>29</td>
<td>29</td>
<td>15</td>
<td>11</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>Pupil H.</td>
<td>30</td>
<td>29</td>
<td>16</td>
<td>13</td>
<td>36</td>
<td>37</td>
</tr>
<tr>
<td>Average Score</td>
<td>93.9%</td>
<td>97.1%</td>
<td>66.1%</td>
<td>77.7%</td>
<td>84.1%</td>
<td>84.7%</td>
</tr>
</tbody>
</table>
# TABLE 2

This table gives the total time required by each pupil to answer the exercises in table 1.

<table>
<thead>
<tr>
<th>Exercise:</th>
<th>Solve 1</th>
<th>Solve 2</th>
<th>Solve 3a</th>
<th>Solve 3b</th>
<th>Solve 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil A.</td>
<td>45:12</td>
<td>31:27</td>
<td>20:42</td>
<td>57:18</td>
<td>58:27</td>
</tr>
<tr>
<td>Pupil C.</td>
<td>41:27</td>
<td>38:12</td>
<td>19:42</td>
<td>51:37</td>
<td>49:17</td>
</tr>
</tbody>
</table>
### TABLE 3

This table summarizes the information in tables 1 and 2. It also shows the results of the pre and post tests.

<table>
<thead>
<tr>
<th>Pupil</th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (182)</th>
<th>Total-Time hr:min:sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>10  67%</td>
<td>12  80%</td>
<td>153  84%</td>
<td>4:20:23</td>
</tr>
<tr>
<td>B.</td>
<td>11  73%</td>
<td>13  87%</td>
<td>162  89%</td>
<td>4:20:21</td>
</tr>
<tr>
<td>C.</td>
<td>11  73%</td>
<td>12  80%</td>
<td>174  96%</td>
<td>4:23:46</td>
</tr>
<tr>
<td>D.</td>
<td>9  60%</td>
<td>11  73%</td>
<td>136  75%</td>
<td>4:36:55</td>
</tr>
<tr>
<td>E.</td>
<td>13  87%</td>
<td>14  93%</td>
<td>160  88%</td>
<td>3:46:57</td>
</tr>
<tr>
<td>F.</td>
<td>9  60%</td>
<td>12  80%</td>
<td>139  76%</td>
<td>6:34:39</td>
</tr>
<tr>
<td>G.</td>
<td>10  76%</td>
<td>10  67%</td>
<td>158  87%</td>
<td>4:39:48</td>
</tr>
<tr>
<td>H.</td>
<td>11  73%</td>
<td>11  73%</td>
<td>161  88%</td>
<td>4:02:23</td>
</tr>
</tbody>
</table>

**Averages**

<table>
<thead>
<tr>
<th></th>
<th>71%</th>
<th>79%</th>
<th>85%</th>
<th>4:05:06</th>
</tr>
</thead>
</table>

Appendix 1
GROUP B WEEK 1 RESULTS

**TABLE 4**

*Group B - Week 1 - Equations using pencil and paper.*

<table>
<thead>
<tr>
<th>Exercise:</th>
<th>Solve 1</th>
<th>Solve 7</th>
<th>Solve 2</th>
<th>Solve 3a</th>
<th>Solve 3b</th>
<th>Solve 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>31</td>
<td>31</td>
<td>24</td>
<td>14</td>
<td>41</td>
<td>41</td>
</tr>
<tr>
<td>Pupil I.</td>
<td>27</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>37</td>
<td>36</td>
</tr>
<tr>
<td>Pupil J.</td>
<td>26</td>
<td>24</td>
<td>17</td>
<td>9</td>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>Pupil K.</td>
<td>30</td>
<td>21</td>
<td>17</td>
<td>9</td>
<td>33</td>
<td>37</td>
</tr>
<tr>
<td>Pupil L.</td>
<td>30</td>
<td>22</td>
<td>21</td>
<td>10</td>
<td>27</td>
<td>38</td>
</tr>
<tr>
<td>Pupil M.</td>
<td>21</td>
<td>21</td>
<td>20</td>
<td>9</td>
<td>31</td>
<td>37</td>
</tr>
<tr>
<td>Pupil N.</td>
<td>22</td>
<td>27</td>
<td>18</td>
<td>10</td>
<td>36</td>
<td>35</td>
</tr>
<tr>
<td>Pupil O.</td>
<td>31</td>
<td>31</td>
<td>20</td>
<td>12</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>Pupil P.</td>
<td>21</td>
<td>30</td>
<td>19</td>
<td>11</td>
<td>34</td>
<td>35</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>27</td>
<td>26</td>
<td>17</td>
<td>12</td>
<td>35</td>
<td>36</td>
</tr>
</tbody>
</table>

| 84% | 81% | 76% | 74% | 83% | 90% |
This table gives the total time required by each pupil to answer the exercises in table 4.

<table>
<thead>
<tr>
<th>Exercises:</th>
<th>Solve 1</th>
<th>Solve 7</th>
<th>Solve 2</th>
<th>Solve 3a</th>
<th>Solve 3b</th>
<th>Solve 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil K.</td>
<td>20:12</td>
<td>22:05</td>
<td>17:18</td>
<td>15:33</td>
<td>26:57</td>
<td>26:00</td>
</tr>
<tr>
<td>Pupil N.</td>
<td>20:14</td>
<td>22:00</td>
<td>16:35</td>
<td>16:12</td>
<td>25:45</td>
<td>26:00</td>
</tr>
<tr>
<td>Pupil O.</td>
<td>19:01</td>
<td>21:45</td>
<td>14:59</td>
<td>14:55</td>
<td>24:00</td>
<td>23:45</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>19:55</td>
<td>20:55</td>
<td>18:45</td>
<td>17:40</td>
<td>26:11</td>
<td>26:45</td>
</tr>
</tbody>
</table>
TABLE 6

This table summarizes the information in tables 4 and 5 and gives the pre and post test results.

<table>
<thead>
<tr>
<th></th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (182)</th>
<th>Total-Time hr:min:sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil I.</td>
<td>11 73%</td>
<td>12 80%</td>
<td>152 94%</td>
<td>2:08:13</td>
</tr>
<tr>
<td>Pupil J.</td>
<td>7 46%</td>
<td>12 80%</td>
<td>147 81%</td>
<td>2:11:45</td>
</tr>
<tr>
<td>Pupil K.</td>
<td>11 73%</td>
<td>11 73%</td>
<td>147 81%</td>
<td>2:07:35</td>
</tr>
<tr>
<td>Pupil L.</td>
<td>9 60%</td>
<td>13 87%</td>
<td>138 76%</td>
<td>2:13:34</td>
</tr>
<tr>
<td>Pupil M.</td>
<td>9 60%</td>
<td>13 87%</td>
<td>139 76%</td>
<td>2:18:16</td>
</tr>
<tr>
<td>Pupil N.</td>
<td>13 87%</td>
<td>13 87%</td>
<td>148 81%</td>
<td>2:06:46</td>
</tr>
<tr>
<td>Pupil O.</td>
<td>15 100%</td>
<td>15 100%</td>
<td>175 96%</td>
<td>1:58:25</td>
</tr>
<tr>
<td>Pupil P.</td>
<td>12 80%</td>
<td>13 87%</td>
<td>150 82%</td>
<td>2:09:03</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>9 80%</td>
<td>12 80%</td>
<td>153 84%</td>
<td>2:10:11</td>
</tr>
</tbody>
</table>

Averages 71%  84%  83%  2:09:19
GROUP B WEEK 2 RESULTS

**TABLE 7**

Group B - Week 2 - Factorising using ExPress.

<table>
<thead>
<tr>
<th>Exercises:</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Factor 4</th>
<th>Factor 5</th>
<th>Factor 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(30)</td>
<td>(12)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
</tr>
<tr>
<td>Pupil I.</td>
<td>26</td>
<td>9</td>
<td>21</td>
<td>22</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>Pupil J.</td>
<td>20</td>
<td>8</td>
<td>21</td>
<td>21</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>Pupil K.</td>
<td>27</td>
<td>10</td>
<td>27</td>
<td>29</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>Pupil L.</td>
<td>21</td>
<td>9</td>
<td>27</td>
<td>28</td>
<td>30</td>
<td>27</td>
</tr>
<tr>
<td>Pupil M.</td>
<td>26</td>
<td>12</td>
<td>30</td>
<td>27</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>Pupil N.</td>
<td>28</td>
<td>11</td>
<td>29</td>
<td>29</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>Pupil O.</td>
<td>28</td>
<td>11</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>Pupil P.</td>
<td>21</td>
<td>8</td>
<td>27</td>
<td>26</td>
<td>25</td>
<td>24</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>19</td>
<td>7</td>
<td>24</td>
<td>21</td>
<td>24</td>
<td>25</td>
</tr>
</tbody>
</table>

| Percentage | 80.0% | 78.7% | 87.4% | 86.3% | 88.1% | 87.7% |
TABLE 8

This table gives the total time required by each pupil to answer the exercises in table 7.

<table>
<thead>
<tr>
<th>Exercises:</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Factor 4</th>
<th>Factor 5</th>
<th>Factor 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil N.</td>
<td>39:58</td>
<td>19:07</td>
<td>38:03</td>
<td>38:11</td>
<td>41:00</td>
<td>34:14</td>
</tr>
<tr>
<td>Pupil P.</td>
<td>40:00</td>
<td>22:06</td>
<td>41:50</td>
<td>42:51</td>
<td>47:15</td>
<td>39:12</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>44:02</td>
<td>24:51</td>
<td>42:37</td>
<td>44:37</td>
<td>48:43</td>
<td>41:11</td>
</tr>
</tbody>
</table>
This table summarizes the information in tables 7 and 8 and also shows the pre and post test results.

<table>
<thead>
<tr>
<th>Pupil</th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercise (162)</th>
<th>Total-Time hr:min:sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>5 33%</td>
<td>13 87%</td>
<td>122 75%</td>
<td>3:57:45</td>
</tr>
<tr>
<td>J</td>
<td>8 53%</td>
<td>11 73%</td>
<td>110 68%</td>
<td>3:57:54</td>
</tr>
<tr>
<td>K</td>
<td>4 27%</td>
<td>9 60%</td>
<td>153 94%</td>
<td>3:23:58</td>
</tr>
<tr>
<td>L</td>
<td>5 33%</td>
<td>13 87%</td>
<td>142 88%</td>
<td>3:56:42</td>
</tr>
<tr>
<td>M</td>
<td>11 73%</td>
<td>11 73%</td>
<td>153 94%</td>
<td>3:34:46</td>
</tr>
<tr>
<td>N</td>
<td>8 53%</td>
<td>12 80%</td>
<td>156 96%</td>
<td>3:30:33</td>
</tr>
<tr>
<td>O</td>
<td>6 40%</td>
<td>11 73%</td>
<td>158 98%</td>
<td>3:31:52</td>
</tr>
<tr>
<td>P</td>
<td>6 40%</td>
<td>10 67%</td>
<td>131 81%</td>
<td>3:52:14</td>
</tr>
<tr>
<td>Q</td>
<td>5 33%</td>
<td>10 67%</td>
<td>120 74%</td>
<td>4:06:01</td>
</tr>
</tbody>
</table>

Averages 43% 74% 85% 3:52:25
GROUP A WEEK 2 RESULTS

**TABLE 10**

<table>
<thead>
<tr>
<th>Exercises:</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Factor 4</th>
<th>Factor 5</th>
<th>Factor 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(30)</td>
<td>(12)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
</tr>
<tr>
<td>Pupil A.</td>
<td>21</td>
<td>7</td>
<td>21</td>
<td>23</td>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>Pupil B.</td>
<td>24</td>
<td>9</td>
<td>26</td>
<td>30</td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>Pupil C.</td>
<td>22</td>
<td>9</td>
<td>24</td>
<td>27</td>
<td>29</td>
<td>24</td>
</tr>
<tr>
<td>Pupil D.</td>
<td>19</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>Pupil E.</td>
<td>23</td>
<td>6</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>21</td>
</tr>
<tr>
<td>Pupil F.</td>
<td>24</td>
<td>7</td>
<td>19</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>Pupil G.</td>
<td>27</td>
<td>8</td>
<td>26</td>
<td>26</td>
<td>24</td>
<td>23</td>
</tr>
<tr>
<td>Pupil H.</td>
<td>21</td>
<td>8</td>
<td>22</td>
<td>21</td>
<td>22</td>
<td>24</td>
</tr>
</tbody>
</table>

75%  64%  77%  81%  81%  78%
TABLE 11

This table gives the total time required by each pupil to answer the questions in table 10.

<table>
<thead>
<tr>
<th>Exercises</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Factor 4</th>
<th>Factor 5</th>
<th>Factor 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil E.</td>
<td>28:15</td>
<td>17:01</td>
<td>27:45</td>
<td>28:33</td>
<td>31:12</td>
<td>33:23</td>
</tr>
</tbody>
</table>
This table summarizes the information in tables 10 and 11.

<table>
<thead>
<tr>
<th></th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercise (162)</th>
<th>Total-Time hr:min:sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil A.</td>
<td>5 33%</td>
<td>9 60%</td>
<td>117 72%</td>
<td>2:44:22</td>
</tr>
<tr>
<td>Pupil B.</td>
<td>8 53%</td>
<td>11 73%</td>
<td>144 89%</td>
<td>2:41:49</td>
</tr>
<tr>
<td>Pupil C.</td>
<td>5 33%</td>
<td>10 67%</td>
<td>135 83%</td>
<td>2:37:27</td>
</tr>
<tr>
<td>Pupil D.</td>
<td>7 46%</td>
<td>11 73%</td>
<td>121 75%</td>
<td>2:45:51</td>
</tr>
<tr>
<td>Pupil E.</td>
<td>11 73%</td>
<td>12 80%</td>
<td>122 75%</td>
<td>2:46:09</td>
</tr>
<tr>
<td>Pupil F.</td>
<td>6 40%</td>
<td>8 53%</td>
<td>110 68%</td>
<td>2:45:58</td>
</tr>
<tr>
<td>Pupil G.</td>
<td>7 46%</td>
<td>9 60%</td>
<td>134 83%</td>
<td>2:39:50</td>
</tr>
<tr>
<td>Pupil H.</td>
<td>6 40%</td>
<td>9 60%</td>
<td>118 73%</td>
<td>2:44:06</td>
</tr>
</tbody>
</table>

| Averages      | 46%           | 66%           | 77%            | 2:43:12               |
COMBINED RESULTS

TABLE 13

A Summary of the results for Group A.

<table>
<thead>
<tr>
<th></th>
<th>Week 1 using ExPress</th>
<th>Week 2 using pen &amp; paper.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EQUATIONS</strong></td>
<td><strong>FACTORISING</strong></td>
<td></td>
</tr>
<tr>
<td>Total Time</td>
<td>36:45:12</td>
<td>21:45:36</td>
</tr>
<tr>
<td>Number of questions</td>
<td>182</td>
<td>162</td>
</tr>
<tr>
<td>Number of pupils</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Questions Attempted</td>
<td>1456</td>
<td>1296</td>
</tr>
<tr>
<td>Correct Responses</td>
<td>1243</td>
<td>1001</td>
</tr>
<tr>
<td>Pre-test average %</td>
<td>71%</td>
<td>46%</td>
</tr>
<tr>
<td>Post-test average %</td>
<td>79%</td>
<td>66%</td>
</tr>
<tr>
<td>Exercises average %</td>
<td>85%</td>
<td>77%</td>
</tr>
</tbody>
</table>
TABLE 14

A Summary of the results for Group B.

<table>
<thead>
<tr>
<th></th>
<th>Week 1 using Pen &amp; paper.</th>
<th>Week 2 using ExPress</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQUATIONS</td>
<td>34:51:45</td>
<td>33:51:45</td>
</tr>
<tr>
<td>Number of questions</td>
<td>182</td>
<td>162</td>
</tr>
<tr>
<td>Number of pupils</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Questions Attempted</td>
<td>1638</td>
<td>1458</td>
</tr>
<tr>
<td>Correct Responses</td>
<td>1349</td>
<td>1245</td>
</tr>
<tr>
<td>Pre-test average %</td>
<td>71%</td>
<td>43%</td>
</tr>
<tr>
<td>Post-test average %</td>
<td>84%</td>
<td>74%</td>
</tr>
<tr>
<td>Exercises average %</td>
<td>83%</td>
<td>85%</td>
</tr>
</tbody>
</table>
TABLE 15
This table gives scores for individual pupils.
( BOLD typeface indicates computer interactions )

<table>
<thead>
<tr>
<th>Maximum Score</th>
<th>T</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>T</th>
<th>T</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil A.</td>
<td>10</td>
<td>30</td>
<td>30</td>
<td>16</td>
<td>9</td>
<td>36</td>
<td>32</td>
<td>12</td>
<td>5</td>
<td>21</td>
<td>7</td>
<td>21</td>
<td>23</td>
<td>21</td>
<td>24</td>
<td>9</td>
</tr>
<tr>
<td>Pupil B.</td>
<td>11</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>35</td>
<td>37</td>
<td>13</td>
<td>8</td>
<td>24</td>
<td>9</td>
<td>26</td>
<td>30</td>
<td>30</td>
<td>25</td>
<td>11</td>
</tr>
<tr>
<td>Pupil C.</td>
<td>11</td>
<td>31</td>
<td>31</td>
<td>21</td>
<td>11</td>
<td>40</td>
<td>39</td>
<td>12</td>
<td>5</td>
<td>22</td>
<td>9</td>
<td>24</td>
<td>27</td>
<td>29</td>
<td>24</td>
<td>10</td>
</tr>
<tr>
<td>Pupil D.</td>
<td>9</td>
<td>24</td>
<td>30</td>
<td>12</td>
<td>9</td>
<td>30</td>
<td>31</td>
<td>11</td>
<td>7</td>
<td>19</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>23</td>
<td>25</td>
<td>11</td>
</tr>
<tr>
<td>Pupil E.</td>
<td>13</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>32</td>
<td>38</td>
<td>14</td>
<td>11</td>
<td>23</td>
<td>6</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>21</td>
<td>12</td>
</tr>
<tr>
<td>Pupil F.</td>
<td>9</td>
<td>27</td>
<td>30</td>
<td>13</td>
<td>12</td>
<td>30</td>
<td>27</td>
<td>12</td>
<td>6</td>
<td>24</td>
<td>7</td>
<td>19</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>8</td>
</tr>
<tr>
<td>Pupil G.</td>
<td>10</td>
<td>29</td>
<td>29</td>
<td>15</td>
<td>11</td>
<td>37</td>
<td>37</td>
<td>10</td>
<td>7</td>
<td>27</td>
<td>8</td>
<td>26</td>
<td>26</td>
<td>24</td>
<td>23</td>
<td>9</td>
</tr>
<tr>
<td>Pupil H.</td>
<td>11</td>
<td>30</td>
<td>29</td>
<td>16</td>
<td>13</td>
<td>36</td>
<td>37</td>
<td>11</td>
<td>6</td>
<td>21</td>
<td>8</td>
<td>22</td>
<td>21</td>
<td>22</td>
<td>24</td>
<td>9</td>
</tr>
<tr>
<td>Pupil I.</td>
<td>11</td>
<td>27</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>37</td>
<td>36</td>
<td>12</td>
<td>5</td>
<td>26</td>
<td>9</td>
<td>21</td>
<td>22</td>
<td>21</td>
<td>23</td>
<td>13</td>
</tr>
<tr>
<td>Pupil J.</td>
<td>7</td>
<td>26</td>
<td>24</td>
<td>17</td>
<td>9</td>
<td>35</td>
<td>36</td>
<td>12</td>
<td>8</td>
<td>20</td>
<td>8</td>
<td>12</td>
<td>21</td>
<td>19</td>
<td>21</td>
<td>11</td>
</tr>
<tr>
<td>Pupil K.</td>
<td>11</td>
<td>30</td>
<td>21</td>
<td>17</td>
<td>9</td>
<td>33</td>
<td>37</td>
<td>11</td>
<td>4</td>
<td>27</td>
<td>10</td>
<td>27</td>
<td>29</td>
<td>30</td>
<td>30</td>
<td>9</td>
</tr>
<tr>
<td>Pupil L.</td>
<td>9</td>
<td>30</td>
<td>22</td>
<td>21</td>
<td>10</td>
<td>27</td>
<td>38</td>
<td>13</td>
<td>5</td>
<td>21</td>
<td>9</td>
<td>27</td>
<td>28</td>
<td>30</td>
<td>27</td>
<td>13</td>
</tr>
<tr>
<td>Pupil M.</td>
<td>9</td>
<td>21</td>
<td>21</td>
<td>20</td>
<td>9</td>
<td>31</td>
<td>37</td>
<td>13</td>
<td>11</td>
<td>26</td>
<td>12</td>
<td>30</td>
<td>27</td>
<td>29</td>
<td>29</td>
<td>11</td>
</tr>
<tr>
<td>Pupil N.</td>
<td>13</td>
<td>22</td>
<td>27</td>
<td>18</td>
<td>10</td>
<td>36</td>
<td>35</td>
<td>13</td>
<td>8</td>
<td>28</td>
<td>11</td>
<td>29</td>
<td>29</td>
<td>30</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>Pupil O.</td>
<td>15</td>
<td>31</td>
<td>31</td>
<td>20</td>
<td>12</td>
<td>40</td>
<td>41</td>
<td>15</td>
<td>6</td>
<td>28</td>
<td>11</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>Pupil P.</td>
<td>12</td>
<td>21</td>
<td>30</td>
<td>19</td>
<td>11</td>
<td>34</td>
<td>35</td>
<td>13</td>
<td>6</td>
<td>21</td>
<td>8</td>
<td>27</td>
<td>26</td>
<td>25</td>
<td>24</td>
<td>10</td>
</tr>
<tr>
<td>Pupil Q.</td>
<td>9</td>
<td>27</td>
<td>26</td>
<td>17</td>
<td>12</td>
<td>35</td>
<td>36</td>
<td>12</td>
<td>5</td>
<td>19</td>
<td>7</td>
<td>24</td>
<td>21</td>
<td>24</td>
<td>25</td>
<td>10</td>
</tr>
</tbody>
</table>
Combined results.

TC — Total marks for those using computer.

TP — Total marks for those using pencil and paper.

MC — Mean mark for those using the computer.

MP — Mean mark for those using pencil and paper.

<table>
<thead>
<tr>
<th>FileName</th>
<th>Computer Usage</th>
<th>Pencil &amp; Paper</th>
<th>Diff. of Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total (TC)</td>
<td>Mean (MC)</td>
<td>Total (TP)</td>
</tr>
<tr>
<td>PreTest</td>
<td>84</td>
<td>10.5</td>
<td>96</td>
</tr>
<tr>
<td>Solve 1</td>
<td>233</td>
<td>29.1</td>
<td>235</td>
</tr>
<tr>
<td>Solve 2</td>
<td>241</td>
<td>30.1</td>
<td>227</td>
</tr>
<tr>
<td>Solve 3</td>
<td>127</td>
<td>15.9</td>
<td>165</td>
</tr>
<tr>
<td>Solve 4</td>
<td>87</td>
<td>10.9</td>
<td>93</td>
</tr>
<tr>
<td>Solve 5</td>
<td>276</td>
<td>34.5</td>
<td>308</td>
</tr>
<tr>
<td>Solve 6</td>
<td>278</td>
<td>34.8</td>
<td>331</td>
</tr>
<tr>
<td>Post Test</td>
<td>95</td>
<td>11.9</td>
<td>114</td>
</tr>
<tr>
<td>Pre Test</td>
<td>58</td>
<td>6.5</td>
<td>55</td>
</tr>
<tr>
<td>Factor 1</td>
<td>216</td>
<td>24.0</td>
<td>181</td>
</tr>
<tr>
<td>Factor 2</td>
<td>85</td>
<td>9.4</td>
<td>61</td>
</tr>
<tr>
<td>Factor 3</td>
<td>227</td>
<td>25.2</td>
<td>184</td>
</tr>
<tr>
<td>Factor 4</td>
<td>233</td>
<td>25.6</td>
<td>194</td>
</tr>
<tr>
<td>Factor 5</td>
<td>238</td>
<td>26.4</td>
<td>194</td>
</tr>
<tr>
<td>Factor 6</td>
<td>237</td>
<td>26.3</td>
<td>187</td>
</tr>
<tr>
<td>Post Test</td>
<td>96</td>
<td>10.7</td>
<td>79</td>
</tr>
</tbody>
</table>
TABLE 17

This table summarises how each pupil performed.

<table>
<thead>
<tr>
<th>Pupil</th>
<th>SOLVE</th>
<th>FACTORISE</th>
<th>COMPUTER</th>
<th>PENCIL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PRE-</td>
<td>POST</td>
<td>SOLVE (182)</td>
<td>FACT. (162)</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>12</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>C</td>
<td>11</td>
<td>12</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>9</td>
<td>11</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>E</td>
<td>13</td>
<td>14</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>F</td>
<td>9</td>
<td>12</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>G</td>
<td>10</td>
<td>10</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>H</td>
<td>11</td>
<td>11</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>I</td>
<td>11</td>
<td>12</td>
<td>5</td>
<td>13</td>
</tr>
<tr>
<td>J</td>
<td>7</td>
<td>12</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>K</td>
<td>11</td>
<td>11</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>L</td>
<td>9</td>
<td>13</td>
<td>5</td>
<td>13</td>
</tr>
<tr>
<td>M</td>
<td>9</td>
<td>13</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>N</td>
<td>13</td>
<td>13</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>O</td>
<td>15</td>
<td>15</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>P</td>
<td>12</td>
<td>13</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Q</td>
<td>9</td>
<td>12</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>
Statistical Results from the Experiment

Symbols used (topic by topic) :-

\[ n \] is the sample size for results taught by computer.
\[ N \] is the sample size for results NOT taught by computer.
\[ s^2 \] is the variance of results taught by computer.
\[ S^2 \] is the variance of results NOT taught by computer.
\[ \sigma_1^2 \] is the population variance for results taught by computer.
\[ \sigma_2^2 \] is the population variance for results NOT taught by computer.
\[ m \] is the mean for the results taught by computer.
\[ M \] is the mean for the results NOT taught by computer.
\[ \mu_1 \] is the population mean for the results taught by computer.
\[ \mu_2 \] is the population mean for the results NOT taught by computer.

The statistics represented here are sectioned as follows:-

1. A test to establish the homogeneity of the population.
2. A test to establish any population differences in the pre-test results.
3. A test on the difference of means on the post-test results.
4. A test on the difference of means on the exercise results.
5. A Summary
To test the homogeneity of the background population

To test for homogeneity it was decided to use the $F$ - Test:

If $\sigma_1^2$ and $\sigma_2^2$ are the population variances and $s_1^2$ and $s_2^2$ are the sample variances and $n$ and $N$ are the sample size then:

$$\frac{n\ s_1^2}{\sigma_1^2} \approx F(n-1, N-1)$$

$$\frac{N\ s_2^2}{\sigma_2^2}$$

Under the Null Hypothesis $\sigma_1^2 = \sigma_2^2$ against

Alternate Hypothesis $\sigma_1^2 \neq \sigma_2^2$ and

$$\frac{n\ s_1^2}{\sigma_1^2} = \frac{N\ s_2^2}{\sigma_2^2} \approx F(n-1, N-1)$$
Using data from above for pupils solving equations:

Table 1.3 Results taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>66.6</td>
<td>73.3</td>
<td>73.3</td>
<td>60.0</td>
<td>86.6</td>
<td>60.0</td>
<td>66.6</td>
<td>73.3</td>
</tr>
</tbody>
</table>

Table 1.6 Results not taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>73.3</td>
<td>46.6</td>
<td>73.3</td>
<td>60.0</td>
<td>60.0</td>
<td>86.6</td>
<td>100.0</td>
<td>80.0</td>
<td>60.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>Standard Deviation (n)</th>
<th>Standard Deviation (n-1)</th>
<th>n</th>
<th>$S^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.3</td>
<td>69.96</td>
<td>8.149</td>
<td>8.712</td>
<td>8</td>
<td>75.899</td>
</tr>
<tr>
<td>Table 1.5</td>
<td>71.08</td>
<td>15.399</td>
<td>16.333</td>
<td>9</td>
<td>266.766</td>
</tr>
</tbody>
</table>

Then $F = \frac{266.766}{75.899} = 3.514$

The 2 tail F test with 2.5% confidence limits gives $F(8,7) = 4.90$

hence $4.90 > 3.514$, so we accept the null hypothesis.
Using data from above for pupils factorising:

Table 2.9  Results taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>33.3</td>
<td>53.3</td>
<td>26.6</td>
<td>33.3</td>
<td>73.3</td>
<td>53.3</td>
<td>40.0</td>
<td>40.0</td>
<td>33.3</td>
</tr>
</tbody>
</table>

Table 2.12  Results not taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>33.3</td>
<td>53.3</td>
<td>33.3</td>
<td>46.6</td>
<td>73.3</td>
<td>40.0</td>
<td>46.6</td>
<td>40.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>n</th>
<th>n-1</th>
<th>S^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 2.3</td>
<td>42.93</td>
<td>13.741</td>
<td>14.575</td>
<td>9</td>
<td>212.430</td>
</tr>
<tr>
<td>Table 2.5</td>
<td>45.80</td>
<td>12.215</td>
<td>13.057</td>
<td>8</td>
<td>170.485</td>
</tr>
</tbody>
</table>

Then \( F = \frac{212.430}{170.485} = 1.246 \)

The 2 tail F test with 2.5 % confidence limits gives \( F(7,8) = 4.53 \)
hence \( 4.53 > 1.246 \), so we accept the null hypothesis.

These two tests establish that the pre-test groups can be considered as samples from the same background population and therefore their sample variances can be justifiably used as an unbiased estimate of the population variance.
The following results using the \( t \)-test will establish that the pre-test and post-test samples could have come from the same population and that the difference in their means is not significant.

Using an unbiased estimator \( \hat{s}^2 \) of the population variance

\[
\hat{s}^2 = \frac{n \cdot s^2 + N \cdot S^2}{n + N - 2}
\]

and for small groups

\[
z = \frac{m - M}{\hat{s} \sqrt{\frac{1}{n} + \frac{1}{N}}}
\]

\[z \approx t \left( N + n - 2 \right)\]

Under the Null Hypothesis \( \mu_1 = \mu_2 \),

and the Alternate Hypothesis \( \mu_1 \neq \mu_2 \).
Appendix 1

[2.1] Pre - Test for Solving Equations (Test 3)

\[ S^2 = \frac{8 \times 76.303 + 9 \times 266.766}{8 + 9 - 2} = 200.754 \]

\[ S = 14.169 \]

\[ z = \frac{-1.120}{14.169 \sqrt{\frac{17}{72}}} \]

\[ z = 0.1626 \]

\[ z \approx t (9 + 8 - 2) \]

The critical value for the 2.5% t-test is 2.13 with 15 degrees of freedom. Hence 2.13 > 0.1626 and we accept the Null Hypothesis.
Pre-Test for Factorising Expressions (Test 4)

\[ S^2 = \frac{9 \times 212.430 + 8 \times 170.485}{8 + 9 - 2} = 218.383 \]

\[ S = 14.777 \]

\[ z \approx t \left( \frac{9 + 8 - 2}{14.777 \sqrt{\frac{17}{72}}} \right) \]

\[ z = 0.3997 \]

The critical value for the 2.5% t-test is 2.13 with 15 degrees of freedom.

Hence 2.13 > 0.3997 and we accept the Null Hypothesis.

These 4 tests establish that the pre-test populations are 'identical'.
[3.1] Test the difference of means in the post-test samples.

Using the same symbols set we set up

The Null Hypothesis to be \( \mu_1 = \mu_2 \),

and the Alternate Hypothesis \( \mu_1 \neq \mu_2 \).

[3.1] Post-test (Test 5)

Using the data from above for pupils solving equations:

Table 1.3 Results taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>80.0</td>
<td>86.6</td>
<td>80.0</td>
<td>73.3</td>
<td>93.3</td>
<td>80.0</td>
<td>66.6</td>
<td>73.3</td>
</tr>
</tbody>
</table>

Table 1.6 Results not taught by computer

<table>
<thead>
<tr>
<th>Pupil</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>80.0</td>
<td>80.0</td>
<td>73.3</td>
<td>86.6</td>
<td>86.6</td>
<td>86.6</td>
<td>100.0</td>
<td>86.6</td>
<td>80.0</td>
</tr>
</tbody>
</table>

\[
\begin{array}{|c|c|c|c|c|}
\hline
 & Mean & Standard Deviation & n & S^2 \\
 & & n & n-1 & \\
\hline
Table 1.3 & 79.14 & 7.777 & 8.313 & 8 & 69.106 \\
Table 1.5 & 84.41 & 7.023 & 7.449 & 9 & 56.235 \\
\hline
\end{array}
\]

\[
S^2 = \frac{8 \times 69.106 + 9 \times 56.235}{8 + 9 - 2} = 70.597
\]

25
\[ S = 8.402 \]

\[ z = \frac{-5.270}{8.402 \sqrt{\frac{17}{72}}} \]

\[ z = 1.2908 \]

\[ z \approx t(9 + 8 - 2) \]

The critical value for the 2.5% t-test is 2.13 with 15 degrees of freedom. Hence 2.13 > 1.2908 and we accept the Null Hypothesis.

[3.2] Post-test (Test 6)

Using data from above for pupils factorising expressions:-

**Table 2.9 Results taught by computer**

<table>
<thead>
<tr>
<th>Pupil</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>86.6</td>
<td>73.3</td>
<td>60.0</td>
<td>86.6</td>
<td>73.3</td>
<td>80.0</td>
<td>73.3</td>
<td>66.6</td>
<td>66.6</td>
</tr>
</tbody>
</table>

**Table 2.12 Results not taught by computer**

<table>
<thead>
<tr>
<th>Pupil</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>60.0</td>
<td>73.3</td>
<td>66.6</td>
<td>73.3</td>
<td>80.0</td>
<td>53.3</td>
<td>60.0</td>
<td>60.0</td>
</tr>
</tbody>
</table>
Appendix 1

<table>
<thead>
<tr>
<th>Table</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>n</th>
<th>$s^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3</td>
<td>74.03</td>
<td>8.566</td>
<td>9</td>
<td>82.555</td>
</tr>
<tr>
<td>2.5</td>
<td>65.81</td>
<td>8.455</td>
<td>8</td>
<td>81.703</td>
</tr>
</tbody>
</table>

$$s^2 = \frac{9 \times 82.555 + 8 \times 81.703}{8 + 9 - 2} = 93.107$$

$$S = 9.649$$

$$z = \frac{8.220}{9.649 \times \frac{17}{\sqrt{72}}}$$

$$z = 1.7531$$

$$z \approx t(9 + 8 - 2)$$

The critical value for the 2.5% t-test is 2.13 with 15 degrees of freedom. Hence 2.13 > 1.7531 and we accept the Null Hypothesis.

Tests 5 and 6 show that the difference in the means of the post-test results of the computer taught and the none computer taught results is not significant at the 5% level.
Test the differences in the means of the Exercise results.

Using the same symbol set we setup

The Null Hypothesis to be $\mu_1 = \mu_2$,
and the Alternate Hypothesis $\mu_1 \neq \mu_2$.

Equation Solving Exercises (Test 7)

Table 1.1 Results taught by computer

<table>
<thead>
<tr>
<th>Exercise (Solve)</th>
<th>1</th>
<th>7</th>
<th>2</th>
<th>3a</th>
<th>3b</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>93.9</td>
<td>97.1</td>
<td>66.1</td>
<td>77.7</td>
<td>84.1</td>
<td>84.7</td>
</tr>
</tbody>
</table>

Table 1.4 Results not taught by computer

<table>
<thead>
<tr>
<th>Exercise (Solve)</th>
<th>1</th>
<th>7</th>
<th>2</th>
<th>3a</th>
<th>3b</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>84.2</td>
<td>81.4</td>
<td>76.4</td>
<td>73.8</td>
<td>83.4</td>
<td>89.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>Standard Deviation n</th>
<th>Standard Deviation n-1</th>
<th>n</th>
<th>$s^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.1</td>
<td>83.93</td>
<td>10.275</td>
<td>11.226</td>
<td>6</td>
<td>126.023</td>
</tr>
<tr>
<td>Table 1.4</td>
<td>81.48</td>
<td>5.271</td>
<td>5.717</td>
<td>6</td>
<td>32.684</td>
</tr>
</tbody>
</table>

\[
\frac{S^2}{\bar{S}^2} = \frac{6 \times 126.023 + 6 \times 32.684}{6 + 6 - 2} = 95.654
\]
\[ S = 9.780 \]

\[ z = \frac{2.450}{\sqrt{\frac{9.780}{6}}} \]

\[ z = 0.4338 \]

\[ z \approx t(6 + 6 - 2) \]

The critical value for the 2.5% t-test is 2.228 with 10 degrees of freedom. Hence 2.228 > 0.4338 and we accept the Null Hypothesis.

[4.2] Factorisation Exercises (Test 8)

Table 2.1 Results taught by computer

<table>
<thead>
<tr>
<th>Exercise (Factor)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>80.0</td>
<td>78.7</td>
<td>87.4</td>
<td>86.3</td>
<td>88.1</td>
<td>87.7</td>
</tr>
</tbody>
</table>

Table 2.4 Results not taught by computer

<table>
<thead>
<tr>
<th>Exercise (Factor)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>75.4</td>
<td>63.5</td>
<td>76.6</td>
<td>80.8</td>
<td>80.8</td>
<td>77.9</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>Standard Deviation</td>
<td>n</td>
<td>$\sigma^2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td>--------------------</td>
<td>----</td>
<td>-----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table 2.1</td>
<td>84.70</td>
<td>3.840</td>
<td>6</td>
<td>17.698</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table 2.4</td>
<td>75.83</td>
<td>5.866</td>
<td>6</td>
<td>41.293</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$S^2 = \frac{6 \times 17.698 + 6 \times 41.293}{6 + 6 - 2} = 35.3946$$

$$S = 5.949$$

$$z = \frac{8.870}{5.949 \sqrt{\frac{2}{6}}}$$

$$z = 2.5824$$

$$z \approx t\left(6 + 6 - 2\right)$$

The critical value for the $2.5\%$ $t$-test is 2.228 with 10 degrees of freedom. Hence $2.228 < 2.5824$ and we reject the Null Hypothesis in favour of the Alternate Hypothesis.
[4.3] Factorisation Exercises

Testing for a significant increase in the mean using a 1 tail test.

Using the same symbols set we setup

The Null Hypothesis to be \( \mu_1 = \mu_2 \),
and the Alternate Hypothesis \( \mu_1 \neq \mu_2 \).

The results of test 8 yield a value of \( z = 2.5824 \) with 10 degrees of freedom, the critical value for \( t \) is 1.81.

As \( 2.5824 > 1.81 \) we reject the Null Hypothesis and conclude that there is a significant increase in the exercises means when factorisation is taught by the computer.

This leads to the premise that:

```
USING A COMPUTER ALGEBRA SYSTEM TO TEACH CHILDREN
IS NO WORSE THAN MORE TRADITIONAL DELIVERY METHODS
```
### 5.0 Summary

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Equation Solving</th>
<th>Factorising</th>
<th>Degrees of Freedom</th>
<th>%</th>
<th>Test Type</th>
<th>Critical Value</th>
<th>Accept NH</th>
<th>AH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRE</td>
<td>3.496</td>
<td></td>
<td>8,7</td>
<td>2.5</td>
<td>F 2t</td>
<td>4.90</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>PRE</td>
<td>1.246</td>
<td></td>
<td>7,8</td>
<td>2.5</td>
<td>F 2t</td>
<td>4.53</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>PRE</td>
<td>0.1626</td>
<td></td>
<td>15</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.13</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>PRE</td>
<td>0.3997</td>
<td></td>
<td>15</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.13</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>POST</td>
<td>1.2908</td>
<td></td>
<td>15</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.13</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>POST</td>
<td>1.7531</td>
<td></td>
<td>15</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.13</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>EX.</td>
<td>0.4338</td>
<td></td>
<td>10</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.228</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>EX.</td>
<td>2.5824</td>
<td></td>
<td>10</td>
<td>2.5</td>
<td>t 2t</td>
<td>2.228</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>EX.</td>
<td>2.5824</td>
<td></td>
<td>10</td>
<td>2.5</td>
<td>t 1t</td>
<td>1.81</td>
<td>✓</td>
<td>×</td>
</tr>
</tbody>
</table>

* t 1t indicates the t - test 1 tail.
* t 2t indicates the t - test 2 tail.
* F 2t indicates the F - test 2 tail.
Appendix 2

A Sample of some of the many extra muMath functions written for ExPress.

FERMAT NUMBERS

% A function to calculate a given Fermat number %
% Example calling procedure is: %
% FERMAT(9) %

FUNCTION FERMAT(N),
     WHEN NOT INTEGER (N) OR N < 0 ,
     MESSAGE(TRUE,"FERMAT function requires INTEGERS > 0 ",2),
     EXIT,
     #TIME : TRUE,
     CHECKTIME(TRUE),
     TEMP : 2^(2^N) + 1,
     CHECKTIME(FALSE),
     #TIME : FALSE,
     FERMAT : TEMP,
ENDFUN$

% A function to calculate a given Euler number %
% Example calling procedure is: %

% EULER (9) %

FUNCTION EULER (N, %Local Variables% T,T1,B,F,P,VAL,C),
     WHEN NOT INTEGER (N),
     EXIT,
     #TIME : TRUE,
     CHECKTIME(TRUE),
     T : 2^(2 N + 2),
     T1 : (2 N)!,
     B : HALFPI*(2 N + 1),
     F : T T1/B,
     P : 'P,
     EUVAL : EXPR( (1/ (P^(2 N + 1))),-1,N,1,3,51,2),
     VAL : F EUVAL,
     PRTMATH (VAL),
     NEWLINE (2) ,
     CHECKTIME(VAL),
     #TIME : FALSE,
     EULER : VAL,
ENDFUN$
BERNOULLI NUMBERS

% A function to calculate a given Bernoulli number %
% Example calling procedure is: %

% BERNO (9) %

FUNCTION BERNO (N,%Local Variables% T,B,B1,F,P,VAL),
WHEN NOT INTEGER (N),
EXIT,
#TIME : TRUE,
CHECKTIME(TRUE),
T: (2 N)!,
B: 2^((2 N - 1)!,
B1: ( 2 * HALFPI)^((2 N)!,
F: T/(B B1),
P : 'P,
BVAL : EXPR( 1/P^((2 N)!,1,N,1,2,20,1),
CHECKTIME(FALSE),
#TIME : FALSE,
BERNO : BVAL * F,
ENDFUN$

FIBONACCI NUMBERS

% A function to calculate a given Fibonacci number %
% Example calling procedure is either: %

% FIB (9) %

% or %

% FIBONACCI (9) %

FUNCTION FIB(N),
PRINTLINE(FIBONACCI(N)),
ENDFUN$

FUNCTION FIBONACCI(N),
WHEN N < 0 OR NOT INTEGER (N),
MESSAGE(TRUE,"Argument NOT acceptable to
FIBONACCI",2),
EXIT,
WHEN N = 0 , 1 EXIT,
WHEN N = 1 , 1 EXIT,
FIBONACCI(N-1) + FIBONACCI(N-2),
ENDFUN$
CATALAN NUMBERS

% A function to calculate a given Catalan number %
% Example calling procedure is either : %

% CATALAN (9) %

FUNCTION CATALAN (N, %Local Variables% T, T1, T2),
WHEN N < 0 OR NOT INTEGER(N),
    MESSAGE (TRUE,"argument NOT acceptable to CATALAN",2),
    EXIT,
#TIME : TRUE,
CHECKTIME(TRUE),
T : NCR(2 N, N),
T1 : 1 / (N+1),
T2 : T * T1,
CHECKTIME(FALSE),
#TIME : FALSE,
ENDFUN$

NCR

% A function to calculate a given NCR number %
% Example calling procedure is either : %

% NCR (9 , 6) %

FUNCTION NCR (N, R, %Local Variables% T1, T2, T3),
WHEN N = FALSE OR R = FALSE,
    MESSAGE(TRUE,"This function is used to",3),
    PRINTLINE("Calculate n"),
    PRINTLINE("C"),
    PRINTLINE("r"),
    PRINTLINE("Hence the minimum number of arguments to"),
    PRINTLINE("NCR is 2"),
    EXIT,
WHEN NOT INTEGER (N) OR NOT INTEGER (R),
    MESSAGE(TRUE,"Error BOTH numbers MUST be integers",2),
    FALSE EXIT,
WHEN N < 0 OR R < 0,
    MESSAGE(TRUE,"ONLY POSITIVE NUMBERS ALLOWED",2),
    FALSE EXIT,
WHEN R > N ,
    MESSAGE(TRUE,"R CANNOT BE GREATER THAN N",2),
    EXIT,
T : N!,
T1: R!,
T2 : (N - R)!,
T3 : T/(T1 T2)
ENDFUN$
Appendix 3

Pupils' Certificate output by Express

*******************************************
*                                      *
*  NORTHICOTE SCHOOL                   *
*  MATHEMATICS                         *
*  DEPARTMENT                          *
*  COMPUTER ALGEBRA COURSE             *
*                                      *
*                                      *
*                                      *
*                                      *
*  Congratulations DAZ you have answered *
*  11 questions during your session with ExPress *
*  today Monday 14th May 1990.          *
*  It took you 13 minutes 48 seconds.    *
*  You got 10 questions correct FIRST time *
*  This certificate counts for 1 point towards a commendation *
*  you need 5 points for 1 commendation *
*                                      *
*  Your Percentage is 90.91 %           *
*                                      *
*******************************************
Appendix 5

Names of Mathematical Operations added to muMath

Additional routines added to muMath. Only the names of the routines and their input syntax is given here. The muSimp code for the routines is available in the supporting material.

FUNCTION P (N, UNK, VALX)
FUNCTION L (N, UNK, VALX)
FUNCTION p (M, N, UNK, VALX)
FUNCTION l (M, N, UNK, VALX)
FUNCTION LEGENDRE (N, UNK, VALX)
FUNCTION LEGENDRE1 (M, N, UNK, VALX)
FUNCTION LAGUERRE (N, UNK, VALX)
FUNCTION LAGUERRE1 (M, N, UNK, VALX)
FUNCTION CHEBYSHEV (N, X, VALX)
FUNCTION CHEBYSHEV2 (N, X, VALX)
FUNCTION CHEBVAL (N, UNK)
FUNCTION CHEBVAL2 (N, UNK)
FUNCTION U (N, UNK, VALX)
FUNCTION T (N, UNK, VALX)
FUNCTION H (N, UNK, VALX)
FUNCTION HERMITE (N, UNK, VALX)
FUNCTION FERMAT (N)
FUNCTION EULER (N)
FUNCTION BERNO (N)
FUNCTION FIBONACCI (N)
FUNCTION CATALAN (N)
FUNCTION FACTORS (N, PRIMEFLAG, TIMEFLAG)
FUNCTION PRIME (EX1)
FUNCTION ISPRIME (N)
FUNCTION AP (A, D, N)
FUNCTION GP (A, R, N)
FUNCTION NCR (N, R, FLAG)
FUNCTION BINOM (X, N, P)
FUNCTION PASCAL (N, S)
FUNCTION MERSENNE (N)
FUNCTION TRAP (F, VAR, UPPER, LOWER, INTER, DP)
FUNCTION RECT (F, VAR, UPPER, LOWER, INTER, DP)
FUNCTION SIMPSON (F, VAR, UPPER, LOWER, INTER, DP)
FUNCTION NEWTON (F, VAR, X, ACC)

UPPER, LOWER, DP, A, R, N, M, D, P, VALX, S and ACC are elements of the set of reals.

PRIMEFLAG, TIMEFLAG, FLAG are Booleans

EX1 is algebraic expression

VAR and UNK are the independent variable.
The following is the menu structure of ExPress, indentation indicating the depth level of each new window.

**Edit**
- Sound On/Off
- Explain
- History
- Output
  - Screen
  - Printer
  - Disk

**Files**
- Save Current Menu Settings
- Change Drive
- Change Directory
- Catalogue Disk
- Exit to DOS
- Change Logon Person

**muMath**
- Execute muMath
- Execute muSimp
- Evaluate
- Set Question Mode
- Set Teach Mode
- Utilities
- Run a DOS Command

**Special**
- Cut and Paste
- Set muMath Variables
  - Numerical
    - Numerator
    - Denominator
    - Base Expansion
    - Power Expansion
    - Log Base
    - Log Expansion
    - Trig Expansion
    - Point
  - Boolean
    - Point
    - Ask User
    - Zero Power
    - PBRCH
    - Zero Base
    - Absolute Logs
    - Degrees
- muMath Notation ON/OFF

37
Appendix 4

Operation
Algebra
Solve
Matrix
Integrate
Ordinary
Functions
Trig.
Hyperbolic
Log
Absolute
Definite
Indefinite
Limits
Taylor
Summations
Products
Differentiate
Functions
Trig.
Hyperbolic
Log
Absolute
Partial
Differential Eqns.
Vectors
Number Theory
Polynomials
Laguerre
Laguerre First
Lagrange
Lagrange First
Hermite
Chebyshev
Numerical Int
Simpsons
Rectangular
Trapeziodal
Probability
N.C.R.
P.N.R.
Mersenne
Primes
Factors
Catalan
Fibonacci
Evaluate a Function
Appendix 4

Progressions
A.P.
G.P.
Harmonic
Binomial
McClauren
Pascal

Number Patterns
Rectangular
Pentagonal
Hexagonal

Graph Function
Appendix 5
Names of Mathematical Operations added to muMath

Additional routines added to muMath. Only the names of the
routines and their input syntax is given here. The muSimp
code for the routines is available in the supporting
material.

FUNCTION P (N,UNK,VALX)
FUNCTION L (N,UNK,VALX)
FUNCTION p (M,N,UNK,VALX)
FUNCTION l (M,N,UNK,VALX)
FUNCTION LEGENDRE (N,UNK,VALX)
FUNCTION LEGENDRE1 (M,N,UNK,VALX)
FUNCTION LAGUERRE (N,UNK,VALX)
FUNCTION LAGUERRE1 (M,N,UNK,VALX)
FUNCTION CHEBYSHEV (N,X,VALX)
FUNCTION CHEBYSHEV2 (N,X,VALX)
FUNCTION CHEBVAL (N,UNK)
FUNCTION CHEBVAL2 (N,UNK)
FUNCTION U (N,UNK,VALX)
FUNCTION T (N,UNK,VALX)
FUNCTION H (N,UNK,VALX)
FUNCTION HERMITE (N,UNK,VALX)
FUNCTION FERMAT(N)
FUNCTION EULER (N)
FUNCTION BERN0 (N)
FUNCTION FIBONACCI (N)
FUNCTION CATALAN (N)
FUNCTION FACTORS (N,PRIMEFLAG,TIMEFLAG)
FUNCTION PRIME (EX1)
FUNCTION ISPRIME (N)
FUNCTION AP (A,D,N)
FUNCTION GP (A,R,N)
FUNCTION NCR (N,R,FLAG)
FUNCTION BINOM (X,N,P)
FUNCTION PASCAL (N,S)
FUNCTION MERSENNE (N)
FUNCTION TRAP (F,VAR,UPPER,LOWER,INTER,DP)
FUNCTION RECT (F,VAR,UPPER,LOWER,INTER,DP)
FUNCTION SIMPSON (F,VAR,UPPER,LOWER,INTER,DP)
FUNCTION NEWTON (F,VAR,X,ACC)

UPPER, LOWER, DP, A, R, N, M, D, P, VALX, S and ACC
are elements of the set of reals.

PRIMEFLAG, TIMEFLAG, FLAG are Booleans

EX1 is algebraic expression

VAR and UNK are the independent variable.
Appendix 6

Table of routines and their syntax

This table contains the names of the mathematical routines which ExPress is able to call. The 6 character code describes the inputs required, and the menu field is the command number issued by the ExPress pull down menu.

<table>
<thead>
<tr>
<th>Comment Field</th>
<th>Algorithm</th>
<th>6 Char. Codes</th>
<th>Menu Number/comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expand</td>
<td>EXPAND</td>
<td>E*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Expand D</td>
<td>EXPD</td>
<td>E*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Factorise</td>
<td>FCTR</td>
<td>E*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Divide Out</td>
<td>DIVOUT</td>
<td>EE*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Partial Fractions</td>
<td>PARFRAC</td>
<td>EE*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Rationalise</td>
<td>RATIONALIZE</td>
<td>E*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Solve</td>
<td>SOLVE</td>
<td>E=EV**0009</td>
<td>*</td>
</tr>
<tr>
<td>Identity Matrix</td>
<td>IDMAT</td>
<td>n*****0008</td>
<td>*</td>
</tr>
<tr>
<td>Transpose</td>
<td></td>
<td>a@*****0010</td>
<td>*</td>
</tr>
<tr>
<td>Dot product</td>
<td></td>
<td>a@a*****0010</td>
<td>*</td>
</tr>
<tr>
<td>Determinant</td>
<td>DET(</td>
<td>a)*****0010</td>
<td>*</td>
</tr>
<tr>
<td>Matrix division</td>
<td>\</td>
<td>a@a*****0010</td>
<td>*</td>
</tr>
<tr>
<td>Power</td>
<td></td>
<td>a@I*****0010</td>
<td>*</td>
</tr>
<tr>
<td>Integrate</td>
<td>INT</td>
<td>EV*****0011</td>
<td>*</td>
</tr>
<tr>
<td>Sine</td>
<td>INT(SIN)</td>
<td>E)V*****0012Sine</td>
<td>*</td>
</tr>
<tr>
<td>Cosine</td>
<td>INT(COS)</td>
<td>E)V*****0012Cosine</td>
<td>*</td>
</tr>
<tr>
<td>Tangent</td>
<td>INT(TAN)</td>
<td>E)V*****0012Tangent</td>
<td>*</td>
</tr>
<tr>
<td>Cosecant</td>
<td>INT(CSC)</td>
<td>E)V*****0012Cosecant</td>
<td>*</td>
</tr>
<tr>
<td>Secant</td>
<td>INT(SEC)</td>
<td>E)V*****0012Secant</td>
<td>*</td>
</tr>
<tr>
<td>Cotangent</td>
<td>INT(COT)</td>
<td>E)V*****0012Cotangent</td>
<td>*</td>
</tr>
<tr>
<td>Arc Sine</td>
<td>INT(ASIN)</td>
<td>E)V*****0012Sine</td>
<td>*</td>
</tr>
<tr>
<td>Arc Cosine</td>
<td>INT(ACOS)</td>
<td>E)V*****0012Cosine</td>
<td>*</td>
</tr>
<tr>
<td>Arc Tangent</td>
<td>INT(ATAN)</td>
<td>E)V*****0012Tangent</td>
<td>*</td>
</tr>
<tr>
<td>Arc Cosecant</td>
<td>INT(ACSC)</td>
<td>E)V*****0012Cosecant</td>
<td>*</td>
</tr>
<tr>
<td>Arc Secant</td>
<td>INT(ASEC)</td>
<td>E)V*****0012Secant</td>
<td>*</td>
</tr>
<tr>
<td>Arc Cotangent</td>
<td>INT(ACOT)</td>
<td>E)V*****0012Cotangent</td>
<td>*</td>
</tr>
<tr>
<td>Sinh</td>
<td>INT(SINH)</td>
<td>E)V*****0013Sinh</td>
<td>*</td>
</tr>
<tr>
<td>Cosh</td>
<td>INT(COSH)</td>
<td>E)V*****0013Cosh</td>
<td>*</td>
</tr>
<tr>
<td>Tanh</td>
<td>INT(TANH)</td>
<td>E)V*****0013Tanh</td>
<td>*</td>
</tr>
<tr>
<td>Cosech</td>
<td>INT(CSCH)</td>
<td>E)V*****0013Cosech</td>
<td>*</td>
</tr>
<tr>
<td>Sech</td>
<td>INT(SECH)</td>
<td>E)V*****0013Sech</td>
<td>*</td>
</tr>
<tr>
<td>Coth</td>
<td>INT(COTH)</td>
<td>E)V*****0013Coth</td>
<td>*</td>
</tr>
<tr>
<td>Arc sinh</td>
<td>INT(ASINH)</td>
<td>E)V*****0013Sinh</td>
<td>*</td>
</tr>
<tr>
<td>Arc cosh</td>
<td>INT(ACOSH)</td>
<td>E)V*****0013Cosh</td>
<td>*</td>
</tr>
<tr>
<td>Arc tanh</td>
<td>INT(ATANH)</td>
<td>E)V*****0013Tanh</td>
<td>*</td>
</tr>
<tr>
<td>Arc cosech</td>
<td>INT(ACSC)</td>
<td>E)V*****0013Cosech</td>
<td>*</td>
</tr>
</tbody>
</table>
Appendix 6

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arc sech</td>
<td>INT(ASECH)</td>
<td>E V**0013 Sech</td>
</tr>
<tr>
<td>Arc coth</td>
<td>INT(ACOTH)</td>
<td>E V**0013 Coth</td>
</tr>
<tr>
<td>Logs to base E</td>
<td>INT(LN)</td>
<td>E V**0014 Ln</td>
</tr>
<tr>
<td>Logs base LOGAS</td>
<td>INT(LOG)</td>
<td>E V**0014 Log</td>
</tr>
<tr>
<td>Logs to base given</td>
<td>INT(LOG)</td>
<td>E Vn**0014 Log</td>
</tr>
<tr>
<td>Expand Logs</td>
<td>LOGEXPD</td>
<td>EV**0014</td>
</tr>
<tr>
<td>Absolute</td>
<td>INT(ABS)</td>
<td>EV**0015{</td>
</tr>
<tr>
<td>Definite Integrate</td>
<td>DEFINIT</td>
<td>EVL**0016</td>
</tr>
<tr>
<td>Integrate</td>
<td>INT</td>
<td>EV**0017</td>
</tr>
<tr>
<td>Limits</td>
<td>LIM</td>
<td>E V**0018</td>
</tr>
<tr>
<td>Limits with variable</td>
<td>LIM</td>
<td>EV**0018</td>
</tr>
<tr>
<td>Limits as x tends to</td>
<td>LIM</td>
<td>E V**0018</td>
</tr>
<tr>
<td>Limits as to below</td>
<td>LIM</td>
<td>EVPT**0018</td>
</tr>
<tr>
<td>Taylor's Series</td>
<td>TAYLOR</td>
<td>EVPn**0019 Taylor</td>
</tr>
<tr>
<td>Summation</td>
<td>SIGMA</td>
<td>EVL**0020</td>
</tr>
<tr>
<td>Products</td>
<td>PROD</td>
<td>EVL**0021</td>
</tr>
<tr>
<td>Differentiate</td>
<td>DIF</td>
<td>Evn**0022</td>
</tr>
<tr>
<td>Sine</td>
<td>DIF(SIN)</td>
<td>Evn**0023 Sine</td>
</tr>
<tr>
<td>Cosine</td>
<td>DIF(COS)</td>
<td>Evn**0023 Cosine</td>
</tr>
<tr>
<td>Tangent</td>
<td>DIF(TAN)</td>
<td>Evn**0023 Tangent</td>
</tr>
<tr>
<td>Cosecant</td>
<td>DIF(CSC)</td>
<td>Evn**0023 Cosecant</td>
</tr>
<tr>
<td>Secant</td>
<td>DIF(SEC)</td>
<td>Evn**0023 Secant</td>
</tr>
<tr>
<td>Cotangent</td>
<td>DIF(COT)</td>
<td>Evn**0023 Cotangent</td>
</tr>
<tr>
<td>Arc Sine</td>
<td>DIF(ASIN)</td>
<td>Evn**0023 Sine</td>
</tr>
<tr>
<td>Arc Cosine</td>
<td>DIF(ACOS)</td>
<td>Evn**0023 Cosine</td>
</tr>
<tr>
<td>Arc Tangent</td>
<td>DIF(ATAN)</td>
<td>Evn**0023 Tangent</td>
</tr>
<tr>
<td>Arc Cosecant</td>
<td>DIF(ACSC)</td>
<td>Evn**0023 Cosecant</td>
</tr>
<tr>
<td>Arc Secant</td>
<td>DIF(ASEC)</td>
<td>Evn**0023 Secant</td>
</tr>
<tr>
<td>Arc Cotangent</td>
<td>DIF(ACOT)</td>
<td>Evn**0023 Cotangent</td>
</tr>
<tr>
<td>Sinh</td>
<td>DIF(SINH)</td>
<td>Evn**0024 Sinh</td>
</tr>
<tr>
<td>Cosinh</td>
<td>DIF(COSH)</td>
<td>Evn**0024 Cosinh</td>
</tr>
<tr>
<td>Tanh</td>
<td>DIF(TANH)</td>
<td>Evn**0024 Tanh</td>
</tr>
<tr>
<td>Cosech</td>
<td>DIF(CSCH)</td>
<td>Evn**0024 Cosech</td>
</tr>
<tr>
<td>Sech</td>
<td>DIF(SECH)</td>
<td>Evn**0024 Sech</td>
</tr>
<tr>
<td>Coth</td>
<td>DIF(COTH)</td>
<td>Evn**0024 Coth</td>
</tr>
<tr>
<td>Arc sinh</td>
<td>DIF(ASINH)</td>
<td>Evn**0024 Sinh</td>
</tr>
<tr>
<td>Arc cosh</td>
<td>DIF(ACOSH)</td>
<td>Evn**0024 Cosh</td>
</tr>
<tr>
<td>Arc tanh</td>
<td>DIF(ATANH)</td>
<td>Evn**0024 Tanh</td>
</tr>
<tr>
<td>Arc cosech</td>
<td>DIF(ACSC)</td>
<td>Evn**0024 Cosech</td>
</tr>
<tr>
<td>Arc sech</td>
<td>DIF(ASECH)</td>
<td>Evn**0024 Sech</td>
</tr>
<tr>
<td>Arc coth</td>
<td>DIF(ACOTH)</td>
<td>Evn**0024 Coth</td>
</tr>
<tr>
<td>Logs to base E</td>
<td>DIF(LN)</td>
<td>Evn**0025 Ln</td>
</tr>
<tr>
<td>Logs base LOGAS</td>
<td>DIF(LOG)</td>
<td>Evn**0025 Log</td>
</tr>
<tr>
<td>Logs to base given</td>
<td>DIF(LOG)</td>
<td>Evn**0025 Log</td>
</tr>
<tr>
<td>Expand Logs</td>
<td>LOGEXPD</td>
<td>Evn**0025 Log</td>
</tr>
<tr>
<td>Absolute</td>
<td>DIF(ABS)</td>
<td>Evn**0026{</td>
</tr>
<tr>
<td>Dependant Variable</td>
<td>DEPENDS</td>
<td>[V(v)]0027</td>
</tr>
<tr>
<td>Diff Variable</td>
<td>DIFVAR</td>
<td>V****0027</td>
</tr>
<tr>
<td>Solve diff. Equ.</td>
<td>SOLVE</td>
<td>DV****0028</td>
</tr>
<tr>
<td>Dependant Variable</td>
<td>DEPENDS</td>
<td>[V(v)]0028</td>
</tr>
<tr>
<td>Diff. Variable is</td>
<td>DIFVAR</td>
<td>V****0028</td>
</tr>
</tbody>
</table>
None Scalar  NONSCALAR  v*****0029  *
Dot product  DOT  e@e***0029  *
Cross  CROSS  e@e***0029  *
Vector expand  VECEXP  en****0029  *
Components  COMPONENTS  e********029  *
Grad  GRAD  S*****0029  *
Laplacian  LAPLACIAN  S*****0029  *
DIV  DIV  ********029  *
CURL  CURL  ********029  *
SCPLOT  SCPLOT  R******029  *
VECPOT  VECPOT  R******029  *
NEWCORDS  NEWCORDS  Rv******029  *
Graph a Function  GRAPH  Ew******065  *
Laguerre Poly.  LAGUERRE  nvv*****066Laguerre  *
Laguerre First Kind  LAGUERRE  nnnv*****067Laguerre First  *
Lagrange Poly.  LAGRANGE  nnnv*****068Lagrange  *
Lagrange First Kind  LAGRANGE  nnnv*****069Lagrange First  *
Hermite Polynomial  HERMITE  nvv*****070Hermite  *
Chebyshev  CHEBY  nvvv*****071Chebyshev  *
Simpsons Rule  SIMPSON  EVLP*****072Simpsons  *
Rectangular Rule  RECT  EVLP*****073Rectangular  *
Trapezoidal Rule  TRAPEZOID  EVLP*****074Trapezoidal  *
Combinations  NCR  nn*****075NCR  *
Permutations  PNR  nn*****076PNR  *
Mersenne Check  MERSENNE  n*****077Mersenne  *
Prime Number Check  PRIME  n*****079Factors  *
Factors of Number  FACTORS  n*****080Catalan  *
Catalan Number  CATALAN  n*****081Catalan  *
Fibonacci Number  FIB  nnn*****083AP  *
Geometric Prog.  GP  nnn*****084GP  *
Harmonic Prog.  HARMONIC  ********085Harmonic  *
Binomial Expansion  BINOMIAL  ********086Binomial  *
Maclurins Theorem  MACLUR  ********087Maclurins  *
Pascal triangle  PASCAL  n*****0117Pascal Nos.  *
Rectangular number  RECTNUMBER  n*****0118Rectangular Nos.  *
Hexagonal number  HEXA  n*****0120Hexagonal Nos.  *
Appendix 7

Algebraic content of the M.E.G. G.C.S.E 1990 and 1991 Examinations

1990 EXAMINATION

The following is a short analysis of the 6 papers for the 1990 G.C.S.E. Mathematics examination.

Paper 1 (Foundation)

Qu.19 required a knowledge of substitution
Qu.24 required the use of algebra and solution of linear equations.

6 marks out of 100 i.e. 6% of the paper required a knowledge of algebra.

Paper 2 (Intermediate)

Qu.3 as Qu.19 from paper 1.
Qu.7 as Qu.24 from paper 1.
Qu.10 forming a linear equation with brackets and being able to solve it.
Qu.19 (a) factorising - with common number and variable.
Qu.22 Solution of three linear equations.

18 marks out of 100 i.e. 18% of the paper required a knowledge of algebra.

Paper 3 (Higher)

Qu.3 as Qu.22 paper 2.
Qu.8 factorising \( ax - 6x -2a +12 \) and \( 6t^2 - t - 2 \)
Qu.10 (b) forming a linear equation.
Qu.13 Substitution a value into a quadratic expression.
Qu.17 Simplifying algebraic fractions.
Qu.18 Use of algebra.
Qu.21 Solution of an inequality.

27 marks out of 100 i.e. 27% of the paper required a knowledge of algebra.
Paper 4 (Foundation 2)

Qu. 2 required the use of algebra.

5 marks out 100 i.e. 5% of the paper required a knowledge of algebra.

Paper 5 (Intermediate 2)

Qu. 2 Use of algebra in parts (a) to (c)
Qu. 7 Graph of a quadratic expression - evaluate plotting points.

16 marks out of 100 i.e. 16% of the paper required a knowledge of algebra.

Paper 6 (Higher 2)

Qu. 2 as Qu. (a), (b) in Qu. 2 paper 5 plus solving a simultaneous equation.
Qu. 3 use of algebra.
Qu. 8 Graphic of a cubic expression - evaluate plotting points.
Qu. 9 use of algebra plus an expansion in part (d).
Qu. 10 use of algebra in a vectors question.

40 marks out of 100 i.e. 40% of the paper required a knowledge of algebra.
Appendix 7

Summary 1990

✓ Indicates grades available in the 1990 Examination

Percentage indicates required algebra knowledge as a fraction of the total paper.

<table>
<thead>
<tr>
<th>Grades Applicable</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>U</th>
<th>Approx % of Algebra req</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOUNDATION</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>5 %</td>
</tr>
<tr>
<td>INTERMEDIATE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
<td></td>
<td>17 %</td>
</tr>
<tr>
<td>HIGHER</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>34 %</td>
</tr>
</tbody>
</table>

To summarise the percentage of algebra on each of the 1990 examination papers was:-

<table>
<thead>
<tr>
<th>Year/Paper</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1990</td>
<td>6%</td>
<td>18%</td>
<td>27%</td>
<td>5%</td>
<td>16%</td>
<td>40%</td>
</tr>
</tbody>
</table>

A similar analysis exercise was performed on the 1991 and 1992 summer exams. The following results were obtained:-

<table>
<thead>
<tr>
<th>Year/Paper</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991</td>
<td>14%</td>
<td>12%</td>
<td>31%</td>
<td>0%</td>
<td>19%</td>
<td>27%</td>
</tr>
<tr>
<td>1992</td>
<td>2%</td>
<td>14%</td>
<td>28%</td>
<td>6%</td>
<td>8%</td>
<td>20%</td>
</tr>
</tbody>
</table>
Hence the average algebra content over the last three years at each of the levels was:

<table>
<thead>
<tr>
<th>Level</th>
<th>1990</th>
<th>1991</th>
<th>1992</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foundation Level</td>
<td>5%</td>
<td>7%</td>
<td>4%</td>
</tr>
<tr>
<td>Intermediate Level</td>
<td>17%</td>
<td>15%</td>
<td>11%</td>
</tr>
<tr>
<td>Higher Level</td>
<td>34%</td>
<td>29%</td>
<td>24%</td>
</tr>
</tbody>
</table>

which tends to support the premise that algebra is playing a less significant role in Mathematics public examinations.
This appendix lists the contents of all the operating system batch files used by ExPress. They are referenced in chapter 5.

Reference 1

To start the system the user types Login, the file Login.Bat is then executed by the operating system. It contains the following commands:

```
REM --- LOGIN.BAT v1.01 ---

cls
e:
cd \nREM - Clean the system of previous users temporary files --
If exist temp.dat erase temp.dat
Loader
runthesy e:mumath
REM -- End of file LOGIN.BAT v1.01 --
```
The main operating system batch file is a direct coding of the diagram 5.2 in chapter 5. The file contains the following commands:

REM --- RUNTHESY.BAT v1.01 ---

:Beg

EXPRESS %1,%2 %3

if errorlevel 11 goto end
if errorlevel 10 goto pend
if errorlevel 8 goto END5
if errorlevel 7 goto END4
if errorlevel 6 goto END3
if errorlevel 5 goto END2
if errorlevel 4 goto END1
if errorlevel 3 second %1
if errorlevel 2 goto Beg

:pend

if exist temp.dat erase temp.dat
if NOT errorlevel 2 termin
if exist end.dat erase end.dat
goto end

:END1

graphing %1
goto end

:END2

cls
musimp SOLVE
EXPRESS %1 0 -1
goto beg

:END3

cls
Musimp PDS
EXPRESS %1 0 -1
goto beg
Appendix 8 — Reference 2 (cont.)

:END5
  cls
  Utils
  EXPRESS %1 0 -1
  goto beg

:END4
  cls
  echo ------ Type 'GO' to return to the system -----
  :end
  e:
  cd \n
REM -- End of file RUNTHESY.BAT v1.01 --

Reference 3

The operating system batch file which calls muSimp and the appropriate muMath system (SOLVE) file is called muMath.bat. The file contains the following commands:

REM --- muMath.BAT v1.01 ---
EVSCREEN
MUSIMP E:SOLVE <inp
E:
EXPRESS %1 0 0
runthesy %1 0 -1
REM --- End of file muMath.BAT v1.01 ---

50
Reference 4

The operating system batch file which calls the graphing plotting program is called Graphing.bat. The file contains the following commands:

```
REM --- GRAPHING.BAT v1.01 ----
MUSIMP E:G <inp >pp
GWBASIC 2D-IBM
E:
EXPRESS %1 0 0
runthesy %1 0 -1
REM --- End of File GRAPHING.BAT v1.01 ----
```

Reference 5

The operating system batch file which restarts the system after a user has requested access to the operating system is called Go.bat. The file contains the following commands:

```
REM --- GO.BAT v1.0 ---
cls
runthesy %1 0 -1
REM --- End of File GO.BAT v1.0 ---
```
Appendix 9

A sample of the teaching materials used.

The experiment generated a large amount of material in the form of text files. The following appendix contains the complete range of material produce for the first factorisation exercise.

Contents:

[9.1] Description of the layout of a Question File
[9.2] Contents of Question File Factor1.Qus
[9.3] Description of the layout of an Overlay File
[9.5] Worksheet Factor1
[9.8] Pre - Test Factorise version 1
[9.9] Post - Test Factorise version 1
[9.1] Description of the layout of a Question File

The layout of each of the question files should be in five sections:-

1. The name of the mumath *.sys file
2. The number of questions in the file
3. The instruction to be displayed
4. The questions
5. The End of File indication.

E.G.

SOLVE ..........The name of the system file used by MuMath
30 .............Number of questions in this file

We are trying to Factorise the following:-- .... Instruction to user

0001*2 x + 4**FCTR(2 x + 4);

--- END OF FILE --- ........ To denote the end of file.
Appendix 9

[9.2] Contents of Question File Factor1.Qus

SOLVE
30
We are trying to Factorise the following :-
0001*2 x + 4**FCTR(2 x + 4);
0002*2 y + 6**FCTR(2 y + 6);
0003*2 w + 8**FCTR(2 w + 8);
0004*3 x + 6**FCTR(3 x + 6);
0005*5 x + 10**FCTR(5 x + 10);
0006*7 x - 21**FCTR(7 x - 21);
0007*6 x + 12**FCTR(6 x + 12);
0008*8 x - 16**FCTR(8 x - 16);
0009*10 x + 20**FCTR(10 x + 20);
0010*11 x - 22**FCTR(11 x - 22);
0011*12 x + 36**FCTR(12 x + 36);
0012*5 x + 25**FCTR(5 x + 25);
0013*5 x + 50**FCTR(5 x + 50);
0014*3 x - 30**FCTR(3 x - 30);
0015*4 x - 8**FCTR(4 x - 8);
0016*10 x + 30**FCTR(10 x + 30);
0017*21 x + 42**FCTR(21 x + 42);
0018*2 x - 22**FCTR(2 x - 22);
0019*5 x + 55**FCTR(5 x + 55);
0020*12 x - 60**FCTR(12 x - 60);
0021*12 + 6 x**FCTR(12 + 6 x);
0022*39 - 13 x**FCTR(39 - 13 x);
0023*12 - 4 y**FCTR(12 - 4 y);
0024*4 y - 16**FCTR(4 y - 16);
0025*5 x + 45**FCTR(5 x + 45);
0026*18 x + 36**FCTR(18 x + 36);
0027*9 x + 36**FCTR(9 x + 36);
0028*9 x - 81**FCTR(9 x - 81);
0029*7 x - 49**FCTR(7 x - 49);
0030*12 x - 24**FCTR(12 x - 24);
--- END OF FILE ---
Appendix 9

Contents of Question File Factor1.Qus

The layout of each of the overlay files should be in four sections:-

1. The name of the overlay file (*.ovl)
2. The number of overlay concepts
3. The length (in characters) of the longest concept
4. The concepts strings followed by the associated range of questions form the question file to which the concept applies.

E.G. (Part of Solve1.ovl file)

SOLVE1.OVL --- Name of overlay file
6 --- Six concept strings
50 --- Longest string is 50 characters

Add a number to both sides. --- Concept string
0 --- Applies to no questions
0

Subtract a number from both sides. --- Concept string
1 --- Applies to questions 1-31
31
[9.3] Description of the layout of an Overlay File

FACTOR1.OVL
5
50
Find a number common in all the terms.
1
30
Put the whole expression into a bracket.
0
0
Add all the terms together.
0
0
Nothing to do in this, it is factorised.
0
0
Quit
0
0
FACT0R1.RHL

@ ----------- Header Information ----------- @

@ ^--- % Find a number common in all the terms. ^--- @

#THIS IS THE CORRECT ANSWER#

To find the number each term has in common we must break each number down into its "PRIME FACTORS."

Example " Qu. Factorise 2 a + 6"

2 is 2 * 1
6 is 2 * 3

therefore "2 a + 6 => 2*1a + 2 * 3"

\( ^\text{\small (Common Number)} \)

"2 a + 6 => 2 * ( 1 a + 3 )"

@ ^--- % Put the whole expression into a bracket. ^--- @

#THIS IS AN INCORRECT ANSWER#

This is an incorrect answer in that it is an unwanted step. It DOES NOT make the solution incorrect it just makes the working longer.

Example " Qu. Factorise 2 a + 6"

"(| 2 a +6 |)"

The expression "2 a + 6" has not been altered by putting into a bracket.

@ ^--- % Add all the expressions together. ^--- @

#THIS IS AN INCORRECT ANSWER#

This is an incorrect answer in that it is an impossible step.

Example " Qu. Factorise 2 a + 6"

2 a + 6

The expression "2 a + 6" cannot be reduced by adding. It is "NOT POSSIBLE" to add #2 lots of a to the #ordinary number 6."
This is an incorrect answer in that it is possible to factorise.

Example  

Qu. Factorise $2a + 6$

2 is $2 \times 1$  
6 is $2 \times 3$

therefore $2a + 6 \Rightarrow 2 \times 1a + 2 \times 3$

(Common Number)

$2a + 6 \Rightarrow 2 \times (1a + 3)$

Also we would not have set a trick question!

Only for those who want to guess !!!!!!
FACT0R1.HLP

@ ----------- Header Information ----------- @
@ ^------- % FACT0R1 FILE HELP. ^------- @

#THIS GIVES AN ILLUSTRATION OF WHAT YOU ARE TRYING TO DO|

Example  " Qu. Factorise 2 a + 6 |

2 is 2 * 1  
6 is 2 * 3

therefore  "2 a + 6 => 2 * 1 a + 2 * 3 | |
          \ /  
          (Common Number)

"2 a + 6 => 2 * ( 1 a + 3 ) | |

@ --------------------- End of File ------------------------ @
Appendix 9

[9.8] Pre - Test Factorise version 1

P R E - T E S T

NAME: __________________________ DATE: _____________

Answer all the Questions. You have 20 minutes

Factorise into brackets each of the following :-

1. $2a + 8$
2. $6x + 4$
3. $7x - 21$
4. $16 - 4x$
5. $24x + 12y + 36$
6. $7xw + 14yw$
7. $12xy + 9xz - 5ax$
8. $5x - 3x$
9. $3x^2 + 5xz$
10. $2xyz + 4xy$
11. $21x - 7xy + 28xyz$
12. $3xy^2 + 6xyz - 9xyz$
13. $8x + 8$
14. $4x^2 - 4xy - 4xy$
[9.9] Post - Test Factorise version 1

POST - TEST

NAME: ___________________________ DATE: ____________

Answer all the Questions. You have 20 minutes

Factorise into brackets each of the following :-

1. \(2a + 8\)
2. \(6x + 4\)
3. \(7x - 21\)
4. \(16 - 4x\)
5. \(24x + 12y + 36\)
6. \(7xw + 14yw\)
7. \(12xy + 9xz - 5ax\)
8. \(5x - 3x\)
9. \(3xt + 5xz\)
10. \(2xyz + 4xy\)
11. \(21x - 7xy + 28xyz\)
12. \(3x^2yz + 6xyz - 9x^2yz\)
13. \(8x + 8\)
14. \(4x^2 - 4xy - 4xy\)
An experiment in the use of computer algebra in the classroom

P.G. Thomas and M.G. Rickhuss
Computing Department, The Open University, Milton Keynes, UK

Abstract


This paper describes an experiment to determine the effect of the use of a computer algebra system in the school classroom. A standard computer algebra package, muMath, was enhanced for use in the classroom and a comparison between it and conventional teaching methods was performed. The conclusion from the experiment indicates that such a system is a useful teaching aid in the classroom.

Keywords: computer algebra; mathematics teaching.

The objective of the experiment

Advances in computer hardware and software have made the teaching of school algebra possible with the aid of computer packages. Most of the packages currently available, however, have been developed for research and professional use and tend to be rather user unfriendly in a school context as identified in the comprehensive review of the impact of computer algebra in the teaching of mathematics [1]. One objective of the work reported here was to make one specific computer algebra (CA) system [2] usable in a classroom situation. The amendments and enhancements to the chosen CA system are described elsewhere [3]. This paper reports on the results of testing the new system with a class of 15 year olds to determine whether or not such a system could enhance the teaching of algebra.

Our computer system was developed around the well-known CA system muMath and was designed to provide the opportunity for children to use a computer algebra system in a structured learning environment. The system is designed to hide the algebraic manipulation and concentrate on the mathematical concepts involved. The experiment had the primary objective of testing the effectiveness of the CA system by focussing on those factors related solely to the CA system itself.

The conduct of the experiment

The algebraic topics covered by the experiment are defined in the UK's National Curriculum document applicable to Key Stage 4 (which pertains to attainment targets in mathematics for 14 to 16 year olds). The average age of the children who took part in the experiment was 15 years 2 months.

The class selected for the experiment consisted of 17 children who were randomly split into two groups of approximately equal abilities. The groups contained approximately equal numbers of pupils and attention was paid to ensuring an even distribution of the sexes.

At the beginning of the experiment, the whole class was told that two topics were to be taught over the following two weeks. The pupils were also told that one of the topics would be taught by computer. During the first week the groups worked on the solution of linear equations. One group was taught using the computer, the other
using more conventional "pencil and paper" methods. The same examples and exercises, taken from [4], were used by both groups.

Prior to the actual teaching, both groups were pre-tested to ascertain their level of knowledge. Each group then spent one week studying each topic. At the end of each week both groups were given the same post-test to determine their level of achievement. The pre- and post-test contained some problems of such a degree of difficulty that no pupil was expected to find the correct solutions.

The second week's topic was factorisation and once again pre- and post-tests were given. This time, however, the roles of the two groups were reversed. The group who had used the computer in the first week were taught using other methods in the second week, and vice-versa. In this way it was hoped that any bias between the two groups, in terms of their ability and use of the computer, would be avoided.

The topics covered were based on the following knowledge.

(1) Solving equations of the following kind:

\[
\begin{align*}
\text{x} + 1 &= 9, \\
\text{x} - 2 &= 10, \\
2\text{x} &= 8, \\
\frac{x}{4} &= 5, \\
2\text{x} + 9 &= 12, \\
4\text{x} + 10 &= 3\text{x} + 9.
\end{align*}
\]

(2) Factorising expressions such as:

\[
\begin{align*}
2\text{x} + 6 & \quad \text{(a common numerical factor)}, \\
3\text{xy} + 4px & \quad \text{(a common variable)}, \\
3\text{xy} + 6xz & \quad \text{(a common numerical and variable factor)}, \\
3\text{xy} + 3x & \quad \text{(a common term)}.
\end{align*}
\]

Following the two-week experiment, each pupil was asked to complete a questionnaire which related to their subjective views about the computer system and their usage of it.

Results obtained by the pupils

In this section we have presented the raw scores obtained by the individual pupils as a collection of tables. We have also shown the time taken to perform the tasks. We shall present an analysis of these results in a later section. In the experiment we need to distinguish between two sets of pupils, which we shall refer to as Group I and Group II, two bodies of knowledge, which we shall refer to as solving equations and factorisation, and two methods of teaching: computer teaching and noncomputer teaching. The tables below show the results obtained for the various combinations of these factors. We have separated the results into those obtained for solving equations and those obtained for factorisation.

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>E1</th>
<th>E7</th>
<th>E2</th>
<th>E3a</th>
<th>E3b</th>
<th>E4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(31)</td>
<td>(31)</td>
<td>(24)</td>
<td>(14)</td>
<td>(41)</td>
<td>(41)</td>
</tr>
<tr>
<td>Pupil A</td>
<td>30</td>
<td>30</td>
<td>16</td>
<td>9</td>
<td>36</td>
<td>32</td>
</tr>
<tr>
<td>Pupil B</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>35</td>
<td>37</td>
</tr>
<tr>
<td>Pupil C</td>
<td>31</td>
<td>31</td>
<td>21</td>
<td>11</td>
<td>40</td>
<td>39</td>
</tr>
<tr>
<td>Pupil D</td>
<td>24</td>
<td>30</td>
<td>12</td>
<td>9</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>Pupil E</td>
<td>31</td>
<td>31</td>
<td>17</td>
<td>11</td>
<td>32</td>
<td>38</td>
</tr>
<tr>
<td>Pupil F</td>
<td>27</td>
<td>30</td>
<td>13</td>
<td>12</td>
<td>30</td>
<td>27</td>
</tr>
<tr>
<td>Pupil G</td>
<td>29</td>
<td>29</td>
<td>15</td>
<td>11</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>Pupil H</td>
<td>30</td>
<td>29</td>
<td>16</td>
<td>13</td>
<td>36</td>
<td>37</td>
</tr>
<tr>
<td>Average Score</td>
<td>93.9%</td>
<td>97.1%</td>
<td>66.1%</td>
<td>77.7%</td>
<td>84.1%</td>
<td>84.7%</td>
</tr>
</tbody>
</table>
Table 2
Time taken for solving equations: computer teaching, Group I

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>E1</th>
<th>E7</th>
<th>E2</th>
<th>E3a</th>
<th>E3b</th>
<th>E4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil A</td>
<td>45:12</td>
<td>47:17</td>
<td>31:27</td>
<td>20:42</td>
<td>57:18</td>
<td>58:27</td>
</tr>
<tr>
<td>Pupil C</td>
<td>41:27</td>
<td>43:31</td>
<td>38:12</td>
<td>19:42</td>
<td>51:37</td>
<td>49:17</td>
</tr>
</tbody>
</table>

Table 3
Pre- and post-tests for solving equations: computer teaching, Group I

<table>
<thead>
<tr>
<th></th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (182)</th>
<th>Total Time hr: min: sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil A</td>
<td>10</td>
<td>67%</td>
<td>12</td>
<td>80%</td>
</tr>
<tr>
<td>Pupil B</td>
<td>11</td>
<td>73%</td>
<td>13</td>
<td>87%</td>
</tr>
<tr>
<td>Pupil C</td>
<td>11</td>
<td>73%</td>
<td>12</td>
<td>80%</td>
</tr>
<tr>
<td>Pupil D</td>
<td>9</td>
<td>60%</td>
<td>11</td>
<td>73%</td>
</tr>
<tr>
<td>Pupil E</td>
<td>13</td>
<td>87%</td>
<td>14</td>
<td>93%</td>
</tr>
<tr>
<td>Pupil F</td>
<td>9</td>
<td>60%</td>
<td>12</td>
<td>80%</td>
</tr>
<tr>
<td>Pupil G</td>
<td>10</td>
<td>76%</td>
<td>10</td>
<td>67%</td>
</tr>
<tr>
<td>Pupil H</td>
<td>11</td>
<td>73%</td>
<td>11</td>
<td>73%</td>
</tr>
<tr>
<td>Averages</td>
<td></td>
<td>71%</td>
<td></td>
<td>79%</td>
</tr>
</tbody>
</table>

Solving equations

The teaching of the solution of equations involved the pupils in tackling a set of exercises. There were six sets of exercises to be completed. For ease of reference, each test set of exercises was given an identifier (ID) related to [4]. Table 1 shows the raw scores obtained for the six sets of exercises for the pupils who were taught to solve equations by the computer. The table also shows the maximum attainable score (as an absolute value) and the average score (as a percentage) for each test set. The time taken by each pupil to answer the questions in each set, given in minutes and seconds, is displayed in Table 2. Table 3 shows the results of the pre- and post-tests for computer teaching for Group I, and also summarizes Tables 1 and 2.

Tables 4, 5 and 6 give the corresponding results for Group II who were taught to solve equations using conventional (noncomputer) methods.

Factorisation

The teaching of factorisation also involved the pupils in tackling a set of six exercises, identified as F1 to F6. The results are presented in a manner similar to those for solving equations.

Table 4
Scores for solving equations: noncomputer teaching, Group II

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>E1</th>
<th>E7</th>
<th>E2</th>
<th>E3a</th>
<th>E3b</th>
<th>E4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(31)</td>
<td>(31)</td>
<td>(24)</td>
<td>(14)</td>
<td>(41)</td>
<td>(41)</td>
</tr>
<tr>
<td>Pupil I</td>
<td>27</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>37</td>
<td>36</td>
</tr>
<tr>
<td>Pupil J</td>
<td>26</td>
<td>24</td>
<td>17</td>
<td>9</td>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>Pupil K</td>
<td>30</td>
<td>21</td>
<td>17</td>
<td>9</td>
<td>33</td>
<td>37</td>
</tr>
<tr>
<td>Pupil L</td>
<td>30</td>
<td>22</td>
<td>21</td>
<td>10</td>
<td>27</td>
<td>38</td>
</tr>
<tr>
<td>Pupil M</td>
<td>21</td>
<td>21</td>
<td>20</td>
<td>9</td>
<td>31</td>
<td>37</td>
</tr>
<tr>
<td>Pupil N</td>
<td>22</td>
<td>27</td>
<td>18</td>
<td>10</td>
<td>36</td>
<td>35</td>
</tr>
<tr>
<td>Pupil O</td>
<td>31</td>
<td>31</td>
<td>20</td>
<td>12</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>Pupil P</td>
<td>21</td>
<td>30</td>
<td>19</td>
<td>11</td>
<td>34</td>
<td>35</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>27</td>
<td>26</td>
<td>17</td>
<td>12</td>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>Average Score</td>
<td>84%</td>
<td>81%</td>
<td>76%</td>
<td>74%</td>
<td>83%</td>
<td>90%</td>
</tr>
</tbody>
</table>
Table 5
Time taken for solving equations: noncomputer teaching, Group II

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil K</td>
<td>20:12</td>
<td>22:05</td>
<td>17:18</td>
<td>15:33</td>
<td>26:57</td>
<td>26:00</td>
</tr>
<tr>
<td>Pupil N</td>
<td>20:14</td>
<td>22:00</td>
<td>16:35</td>
<td>16:12</td>
<td>25:45</td>
<td>26:00</td>
</tr>
<tr>
<td>Pupil O</td>
<td>19:01</td>
<td>21:45</td>
<td>14:59</td>
<td>14:55</td>
<td>24:00</td>
<td>23:45</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>19:55</td>
<td>20:55</td>
<td>18:45</td>
<td>17:40</td>
<td>26:11</td>
<td>26:45</td>
</tr>
</tbody>
</table>

Table 6
Pre- and post-tests for solving equations: noncomputer teaching, Group II

<table>
<thead>
<tr>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (182)</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No.</td>
<td>Perc.</td>
<td>No.</td>
</tr>
<tr>
<td>Pupil I</td>
<td>11</td>
<td>73%</td>
<td>12</td>
</tr>
<tr>
<td>Pupil J</td>
<td>7</td>
<td>46%</td>
<td>12</td>
</tr>
<tr>
<td>Pupil K</td>
<td>11</td>
<td>73%</td>
<td>11</td>
</tr>
<tr>
<td>Pupil L</td>
<td>9</td>
<td>60%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil M</td>
<td>9</td>
<td>60%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil N</td>
<td>13</td>
<td>87%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil O</td>
<td>15</td>
<td>100%</td>
<td>15</td>
</tr>
<tr>
<td>Pupil P</td>
<td>12</td>
<td>80%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>9</td>
<td>60%</td>
<td>12</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>71%</td>
<td></td>
</tr>
</tbody>
</table>

Table 7
Scores for factorisation: computer teaching, Group II

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(30)</td>
<td>(12)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
</tr>
<tr>
<td>Pupil I</td>
<td>26</td>
<td>9</td>
<td>21</td>
<td>22</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>Pupil J</td>
<td>20</td>
<td>8</td>
<td>21</td>
<td>21</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>Pupil K</td>
<td>27</td>
<td>10</td>
<td>27</td>
<td>29</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>Pupil L</td>
<td>21</td>
<td>9</td>
<td>27</td>
<td>28</td>
<td>30</td>
<td>27</td>
</tr>
<tr>
<td>Pupil M</td>
<td>26</td>
<td>12</td>
<td>30</td>
<td>27</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>Pupil N</td>
<td>28</td>
<td>11</td>
<td>29</td>
<td>29</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>Pupil O</td>
<td>28</td>
<td>11</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>Pupil P</td>
<td>21</td>
<td>8</td>
<td>27</td>
<td>26</td>
<td>25</td>
<td>24</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>19</td>
<td>7</td>
<td>24</td>
<td>21</td>
<td>24</td>
<td>25</td>
</tr>
<tr>
<td>Average</td>
<td>80.0%</td>
<td>78.7%</td>
<td>87.4%</td>
<td>86.3%</td>
<td>88.1%</td>
<td>87.7%</td>
</tr>
</tbody>
</table>

However, the reader should note that the roles of the two groups were reversed: Group II were taught using the computer, but Group I were exposed to noncomputer teaching. Table 7 gives the raw scores for the pupils who were taught by the computer. The time, measured in minutes and seconds, taken by each pupil in Group II to answer the questions in each set using the computer is given in Table 8.

Table 9 shows the results of the pre- and
Table 8
Times taken for factorisation: computer teaching, Group II

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupil N</td>
<td>39:58</td>
<td>19:07</td>
<td>38:03</td>
<td>38:11</td>
<td>41:00</td>
<td>34:14</td>
</tr>
<tr>
<td>Pupil P</td>
<td>40:00</td>
<td>22:06</td>
<td>41:50</td>
<td>42:51</td>
<td>47:15</td>
<td>39:12</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>44:02</td>
<td>24:51</td>
<td>42:37</td>
<td>44:37</td>
<td>48:43</td>
<td>41:11</td>
</tr>
</tbody>
</table>

Table 9
Pre- and Post test for factorisation: computer teaching, Group II

<table>
<thead>
<tr>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (162)</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No.</td>
<td>Perc.</td>
<td>No.</td>
</tr>
<tr>
<td>Pupil I</td>
<td>5</td>
<td>33%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil J</td>
<td>8</td>
<td>53%</td>
<td>11</td>
</tr>
<tr>
<td>Pupil K</td>
<td>4</td>
<td>27%</td>
<td>9</td>
</tr>
<tr>
<td>Pupil L</td>
<td>5</td>
<td>33%</td>
<td>13</td>
</tr>
<tr>
<td>Pupil M</td>
<td>11</td>
<td>73%</td>
<td>11</td>
</tr>
<tr>
<td>Pupil N</td>
<td>8</td>
<td>53%</td>
<td>12</td>
</tr>
<tr>
<td>Pupil O</td>
<td>6</td>
<td>40%</td>
<td>11</td>
</tr>
<tr>
<td>Pupil P</td>
<td>6</td>
<td>40%</td>
<td>10</td>
</tr>
<tr>
<td>Pupil Q</td>
<td>5</td>
<td>33%</td>
<td>10</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>43%</td>
<td></td>
</tr>
</tbody>
</table>

post-tests and a summary of Tables 7 and 8.

Group I were taught factorisation using non-computer methods and their results for the same exercises, pre- and post-tests are shown in Tables 10, 11 and 12.

Analysis of the results

Table 13 compares the scores of the two groups when tackling the exercises for solving equations. The computer-taught group scored better in three exercises, worse in two and one was about the same.

Table 14 shows the comparison of pre- and post-tests for the two groups for solving equations. Group I (taught by the computer) seemed as well prepared as Group II but did not progress as well as the noncomputer-taught group, although the average mark on the exercises completed during the experiment exhibit little difference between the two groups.

Table 14 also shows a comparison of the time taken to complete the exercises for solving equa-

Table 10
Scores for factorisation: noncomputer teaching, Group I

<table>
<thead>
<tr>
<th>Test set ID</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>F6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Score</td>
<td>(30)</td>
<td>(12)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
<td>(30)</td>
</tr>
<tr>
<td>Pupil A</td>
<td>21</td>
<td>7</td>
<td>21</td>
<td>23</td>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>Pupil B</td>
<td>24</td>
<td>9</td>
<td>26</td>
<td>30</td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>Pupil C</td>
<td>22</td>
<td>9</td>
<td>24</td>
<td>27</td>
<td>29</td>
<td>24</td>
</tr>
<tr>
<td>Pupil D</td>
<td>19</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>Pupil E</td>
<td>23</td>
<td>6</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>21</td>
</tr>
<tr>
<td>Pupil F</td>
<td>24</td>
<td>7</td>
<td>19</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>Pupil G</td>
<td>27</td>
<td>8</td>
<td>26</td>
<td>26</td>
<td>24</td>
<td>23</td>
</tr>
<tr>
<td>Pupil H</td>
<td>21</td>
<td>8</td>
<td>22</td>
<td>21</td>
<td>22</td>
<td>24</td>
</tr>
<tr>
<td>Average Score</td>
<td>75%</td>
<td>64%</td>
<td>77%</td>
<td>81%</td>
<td>81%</td>
<td>78%</td>
</tr>
</tbody>
</table>
Table 11
Times taken for factorisation: noncomputer teaching, Group I

<table>
<thead>
<tr>
<th>Pupil</th>
<th>Test set ID</th>
<th>F1 (hr:min:sec)</th>
<th>F2 (hr:min:sec)</th>
<th>F3 (hr:min:sec)</th>
<th>F4 (hr:min:sec)</th>
<th>F5 (hr:min:sec)</th>
<th>F6 (hr:min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Pupil E</td>
<td>28:15</td>
<td>17:01</td>
<td>27:45</td>
<td>28:33</td>
<td>31:12</td>
<td>33:23</td>
</tr>
</tbody>
</table>

Table 12
Pre- and post-test for factorisation; noncomputer teaching, Group I

<table>
<thead>
<tr>
<th>Pupil</th>
<th>Pre-Test (15)</th>
<th>Post-Test (15)</th>
<th>Exercises (162)</th>
<th>Total Time (hr:min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>9</td>
<td>117</td>
<td>72%</td>
</tr>
<tr>
<td>B</td>
<td>8</td>
<td>11</td>
<td>144</td>
<td>89%</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>10</td>
<td>135</td>
<td>83%</td>
</tr>
<tr>
<td>D</td>
<td>7</td>
<td>11</td>
<td>121</td>
<td>75%</td>
</tr>
<tr>
<td>E</td>
<td>11</td>
<td>12</td>
<td>122</td>
<td>75%</td>
</tr>
<tr>
<td>F</td>
<td>6</td>
<td>8</td>
<td>110</td>
<td>68%</td>
</tr>
<tr>
<td>G</td>
<td>7</td>
<td>9</td>
<td>134</td>
<td>83%</td>
</tr>
<tr>
<td>H</td>
<td>6</td>
<td>9</td>
<td>118</td>
<td>73%</td>
</tr>
<tr>
<td>Average</td>
<td>46%</td>
<td>66%</td>
<td>77%</td>
<td></td>
</tr>
</tbody>
</table>

Table 13
Comparison of the two Groups: solving equations

<table>
<thead>
<tr>
<th></th>
<th>E1</th>
<th>E7</th>
<th>E2</th>
<th>E3a</th>
<th>E3b</th>
<th>E4</th>
</tr>
</thead>
<tbody>
<tr>
<td>computer-taught (Group I)</td>
<td>93.9%</td>
<td>86.3%</td>
<td>66.1%</td>
<td>77.7%</td>
<td>84.1%</td>
<td>84.7%</td>
</tr>
<tr>
<td>noncomputer (Group II)</td>
<td>84%</td>
<td>81%</td>
<td>76%</td>
<td>74%</td>
<td>83%</td>
<td>90%</td>
</tr>
</tbody>
</table>

The comparison of pre- and post-tests for the two groups in factorisation is shown in Table 17. Group II made the better progress from a slightly poorer base.

Note that the whole group (I and II) had met
the solution of linear equations earlier in their school careers. This is reflected in the higher pre-test scores for equations compared to factorisation.

The ratio of time taken using the computer to the time taken using noncomputer methods for solving equations is approximately 2 (see Table 14) whereas a similar ratio for factorisation is 1.4 (see Table 17). That is, the computer time is about the same for both solving equations and factorisation whereas the noncomputer time significantly increases for factorisation. The reason is straightforward: the pupils had not previously encountered factorisation but had met the solution of equations before. Hence, factorisation by noncomputer methods required more time for thinking and writing compared with factorisation carried out by the computer.

The total time for answering 1638 questions by all pupils by hand was 2092 minutes, which can be compared with 1456 questions answered by all pupils by computer in 2205 minutes. If one reduces the computer time to allow for disk to memory swapping (approximately 1 minute per question) the total computer time reduces to 749 minutes. That is, using the computer, the pupils required about one third of the time needed for hand calculations.

### Statistical analysis

To obtain a feeling for the significance of the results some very straightforward statistical analysis can be performed on the data. In particular, we want to show two things:

1. that the two groups were not significantly different in ability;
2. that the results for computer teaching are no worse than for noncomputing teaching.

To do this we shall simply indicate the process and refer the interested reader to standard texts for further details of the mathematics involved.

In what follows we shall use the following notations:

- \( n \) is the sample size,
- \( s^2 \) is the sample variance.

The abbreviation C will be used for a computer-taught group, and N for a noncomputer-taught group. Thus, for example, the sample size of a
computer-taught group will be represented by \( n_C \), and the variance of a sample of a noncomputer-taught group will be represented by \( s_N^2 \).

The homogeneity of the background population can be examined using an \( F \)-test. Under the null hypothesis—that the variances of the two groups are the same—we should obtain the following correspondence:

\[
\frac{n_C s_C^2}{n_N s_N^2} \sim F(n_C - 1, n_N - 1).
\] (1)

Using the pre-test data in Tables 3 and 6 for the solution of equations, the ratio in (1) is 3.45. The critical \( F \)-value (2-tail) is: \( F_{2.5\%}(8, 7) = 4.90 \). Since 4.90 > 3.45 we can accept the null hypothesis. A similar calculation for the post-test results for solving equations reveals that we can accept the same null hypothesis. This leads us to conclude that the groups can be considered as samples from the same background population, and that we can justifiably use their sample variances to calculate an unbiased estimate of the population variance.

Similar calculations, this time using a \( t \)-test, show that the pre-test populations for both equation solving and factorisation are “identical”. The same conclusion is reached for the post-test results. That is, the difference in post-test means between the computer-taught and the noncomputer-taught groups is not significant at the 5% level.

The only case where the statistical tests show a significant difference in the results of the two groups is in tackling the factorisation exercises where the computer-taught pupils show an increase in achievement over the noncomputer-taught pupils. Of course, the significant difference between the two is that, in the case of factorisation, this was the first time that the pupils had met this topic. With the limited amount of data available it would be unwise at this stage to postulate the reason for the difference; much more research needs to be carried out.

Chi-square goodness of fit tests were performed on the change in performance between pre- and post-tests. For equation solving, the test did not reveal a significant improvement (at the 5% level) using the computer compared to the conventional teaching method. However, for factorisation, the results for the computer-taught group are significantly better (5% level) than for the other group.

### Conclusions

Table 18 summarises all the results. The highlighted values are for the computer-taught components of the experiment.

One group seemed to be as equally well prepared as the other both for equation solving and for factorisation with little difference shown in the pre-test results. The statistical analysis of the post-test results show no significant difference between the two groups for either equation solving or factorisation. However, the computer-taught group did do better than the noncomputer-taught group on the exercises.

The post-test indicate that both groups made progress whether using the computer or not.

The most significant point to arise was the degree to which factorisation seemed to be better learned using the computer than with more conventional means. Also, the exercises done during the experiment seem to be answered better using the computer.

A conjecture is that with conventional methods a significant amount of time is spent writing down the solutions since, with the computer, only about one-third of this time is spent thinking about the problem. The results, shown in Table 15, from a later study with a faster computer (which reduced the waiting time between problems) reinforce this view.

The conclusion is that computer teaching using a CA system is viable in the classroom. Importantly, no evidence was found which indicated that pupils fared worse using the computer.
Acknowledgement

The authors would like to thank Peter Mitic for his contribution, particularly in the statistical analyses of the results.

References


Mike Rickhuss gained his Cert.Ed. from the University of Birmingham, England and a BA (Hons) from the Open University. He is currently completing a M.Phil. with the Open University. He is resident in Wolverhampton where he is a part-time computer consultant and teacher.

Peter Thomas gained his B.Sc. from the University of Leicester, England and his M.Sc. and Ph.D. from the University of Dundee, Scotland. He is currently the Dean of the Mathematics Faculty at the Open University. He is a member of the British Computer Society, a Chartered Engineer and an Associate Fellow of the Institute of Mathematics and its Applications.