



apple user

Vol. 4, No. 2

February 1984

E1

-the new name for
Windfall

Macintosh revealed!

We take a
close look
at what
promises to
be the big
computer
sensation
of 1984



DOS 3.3 successor reviewed

Design your own graphics package

Cashflow calculations using Visicalc

Books

● We review a series of books from Usborne which cater for a wide range of newcomers and are particularly suitable for children. **29**

Business

● How a British clothing manufacturer and retailer used an Apple II to help decide on a £4.25 million takeover bid. **24**

● Nick Levy continues his discussion on developing a cash flow model with Visicalc. He also shows the value of @ISERROR **58**

Darts update

● Many people have played darts on the Apple using our listing published in March 1983. This routine gives voice to your darts game. **50**

Feedback

● Your letters on failed keyboard keys, our Patience listing, the ALS CP/M Plus card, and on problems with Applewriter II, the RND function and an 8in disc drive controller. **70**

Games

● Reviews of The Quest (an adventure game with more than one route to the solution),

apple user

Volume 4
Number 2
February 1984



Macintosh, the mighty mite

Story Machine (instant authorship for children); Repton (probably the best defender game yet on the Apple) and Sammy Lightfoot (the daring young man on the trapeze, trampoline and other hazardous equipment). **42**

Graphics

● Develop your own library of integrated graphical routines. Peter Gorry explains the basic considerations to take into account in designing such a system. **40**

Lisa

● Mike Glover concludes his article on the Lisa Workshop software with a look at the Pascal and Quickdraw packages. **37**

Logo

● Looking at Lists – and

Logo's power for manipulating languages. **33**

Macintosh

● The next best seller in the Apple range. We give full details and first impressions of the remarkable new micro that is inexpensive, powerful and refines the revolution that was embodied in Lisa. **17**

New products

● Round up of the latest products on the market for Apple users. **66**

News

● Apple and Franklin settle their differences; more lookalikes slip through the net; price war imminent? **13**

Pascal

● Produce animation with Pascal – Jonathon

Lewis shows you the technique. **56**

● Tutorial, Part II. Extensions, conditional and compound statements, Begin-End pairs, loops and examples of nested statements/conditions. **53**

Pro DOS

● A new operating system for the Apple II which will replace DOS 3.3 and give the Apple II much closer links with the Apple III. The first of two articles by Keith Lander. **30**

Technique

● Manipulate dates within programs using two algorithms from Max Parrott and a date conversion routine listing from Basil Shall. **47**

Reviews

● Bank Street Writer: A word processor aimed at and suited to the educational market. **61**

● Keystar. Linked to the Apple II via a serial card, this peripheral provides a simple way to enter commands using the WordStar program. **65**

● Word Weaver: A poor WP program for the Apple III – but it provides an interesting example of how not to write and present software. **73**

Apple User Managing Editor **Derek Meakin** Production Editor **Peter Glover** Features Editor **David Creasey**
 Technical Editors **Cliff McKnight** **Max Parrott** Editor-in-Chief of Database Publications **Peter F. Brameld**
 Marketing Manager **Sue Casewell** Advertisement Manager **John Riding** Sales **Peter Nowell**

Telephone: 061-456 8383 (Editorial) 061-456 8500 (Advertising) Telex: 667664 SHARET G. Prestel: 614568383.

Published by Database Publications Ltd, Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY.

Trade distribution in UK and Ireland by Wells, Gardner, Darton & Co Ltd, Faygate, Horsham, West Sussex RH12 4SU. Tel: Faygate 444.

Apple and the Apple symbol are the registered trade marks of Apple Computer Inc. Windfall is an independent publication and Apple Computer is not responsible for any of the articles in this magazine, nor for any of the opinions expressed.

Writing for Apple User: Articles and programs relating to the Apple are welcome. Articles should preferably be typed or computer-printed, using double spacing. Unsolicited manuscripts, discs etc, should be accompanied by a self addressed stamped envelope, otherwise their return cannot be guaranteed. Unless agreed, material is accepted on an all rights basis.

© 1984 Database Publications Ltd. No material may be reproduced in whole or in part without written permission. While every care is taken, the publishers cannot be held legally responsible for any errors in articles or listings.

Subscription rates for 12 issues, post free:
 £12 UK
 £13 Eire (IR £16)
 £18 Europe
 £15 USA (surface)
 £25 USA (airmail)
 £15 Rest of world (surface)
 £30 Rest of world (airmail)

Franklin fracas is over

THE long-standing squabble between Apple and rival computer manufacturers Franklin appears to have ended – with Apple winning its fight in defence of its operating system copyrights.

Nearly two years ago Apple took Franklin to court, alleging that Franklin's Ace series of Apple-compatible micros infringed Apple copyrights.

Preliminary court hearings seasawed between the two

companies.

The first rejected Apple's appeal for an injunction. The second ruled that Apple's operating system could be protected under law.

Apple had asserted that Franklin had copied 14 of their computer operating system programs.

Franklin, in response, challenged the validity of Apple's copyright and also asserted that Apple was guilty of anti-trust violation and unfair competition.

Now the two companies have agreed to a settlement.

Franklin is to pay Apple \$2.5 million dollars and has agreed not to infringe Apple's copyrights.

In turn Apple has agreed to terminate its legal action for damages in the Federal Court.

In an announcement last month the two companies say they will establish arbitration procedure for any future copyright disputes "when the copies relate to any substantively Apple-compatible computer

programs".

Franklin will remain in business and have been given a period of grace in which to sell its remaining stocks of Ace machines.

It will then start producing micros that do not infringe Apple copyrights.

Franklin has apparently written its own object code which is not a carbon copy of the Apple operating systems but which will make its machines compatible with most, but not all, Apple software.

Lots to see at Apple '84

APPLE '84, the largest annual three-day Apple get-together in Europe, will take place from Thursday, May 24 to Saturday, May 26.

It is the third Apple exhibition and user convention organised by Database Publications.

The venue is again the Fulcrum Centre in Slough, which has been fully air-conditioned since Apple '83.

The variety of new products released by Apple alone since the last show will ensure considerable public interest in the event.

Macintosh, announced last month, should be on

general sale in the UK by then, and Apple officials have assured us that other new offerings are "on the boil".

There will be a chance to see the new mouse-orientated and integrated software packages for the Apple IIe and III.

Also rumoured for release by then should be a second version of Lisa and an upgraded version of Applewriter.

This year's User Convention will be carefully balanced to ensure that it caters for all users, ranging from new owners of Macintosh and Lisa to stalwart supporters of the Apple II Plus.

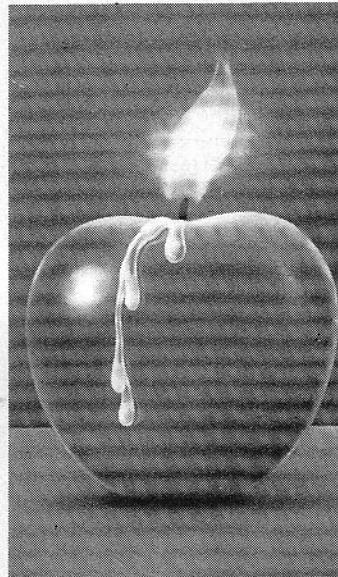
Apple's the key ...

THE Apple III is a splendid computer but it does have its critics.

So all the more interesting to see it featured in a major, non-Apple sponsored advertisement.

The ad is to promote the new Software '84 Exhibition being held at Earls Court in June.

It pictures an Apple III, with a large key attached to it, on which is written, "The Key to Successful Software Sales".



Burning a candle for IBM?

THERE are two big names in the micro computer world – Apple and IBM.

IBM, the largest computer company in the world, was late entering the micro arena, but when it did it had a major impact on the market and has now overtaken Apple in volume of sales.

Apple is not unduly concerned and will almost certainly be one of the few companies to continue to dominate the field alongside IBM, mainly because of its flair for innovation and pioneering research.

It might, temporarily, have lost its leading edge, but it hasn't lost its sense of humour.

Apple UK, for example sent out Christmas cards featuring an Apple candle (non-regulation colours) with its flame proudly burning and inside the message:

"Here's to a 1984 without Big Brother". That could be a subtle reference to either George Orwell or IBM.

Apple refers to itself as "one computer company that's doing more than just wishing for a Happy Christmas and a prosperous New Year".

ROYAL PATRON

A CHARITY that provides Apple computers and other electronic aids to the disabled will premiere a film about its work on February 28 in London.

The Aids Trust, whose patron is Prince Philip, will host a charity lunch at the Drapers' Hall for 200 guests, including the Lord Mayor.

In these pages we have often reported how Apple computers can be used as a stimulus for handicapped children and the severely spastic for communicating with other people.

Taiwan lookalike tries a new ploy

A FRESH crop of Apple lookalikes is posing a growing problem in the UK.

For a steady flow of copy Apples – from a variety of sources – has been harvesting rich pickings on the British computer marketplace of late.

"It is an increasing problem", admits Mike Spring, marketing manager of Apple UK, "but we are closely monitoring the situation and taking the necessary legal action to protect our interests".

A major concern is that many of the new lookalikes are of much better quality than their predecessors.

"As a rule they used to be dreadful", Mike Spring told *Apple User*, "but they are getting cleverer. However some 50 per cent are still very inferior machines".

But the superior copies – often at around half the price of their legitimate counterparts – have begun to make a dent in the UK market.

"Now we all know Apple computers are relatively cheap to make", said Mike Spring, "but our prices have to include research and development, support and the like, whereas the lookalikes don't.

"So the people behind them are using us to finance them. Because of this, we will always take issue with those who deliberately infringe our copyright".

Just as the Apple copies are becoming more sophisticated, so too are the methods of peddling them to prospective customers.

At present a machine made in the Far East is finding its way

to Hong Kong and from there to Saudi Arabia – a country which does not recognise copyright – before being sold mail order direct to the UK.

The one fear for potential purchasers of machines direct from abroad has been "what happens if it goes wrong?"

But one enterprising lookalike manufacturer seems to have solved even this problem.

For Sapling Devices, the manufacturer based in Taiwan, has made an arrangement with an electronic assembly specialist company based in Yorkshire – to provide a one year back-up warranty for a fee of £15.

Known as the Colt PC, the machine is being offered for sale direct to customers in the UK for £245 – and that includes delivery.

The computer contains a standard 48k Apple compatible motherboard, while the keyboard is an enhanced version of the Apple in that it has upper and lower case characters.

As the keyboard has its own microprocessor, it provides a 188 function key operation.

Claiming to be better than the Apple IIe, it falls down in that it doesn't have the extra 16k main memory. But it can be upgraded with a 16k card for £30.

The Colt's case is completely different from the Apple's, but it does have a removeable lid.

The manufacturer has appointed its own advertising agency in this country – also known as Sapling Devices.

Proprietor of this operation is Tony Roberts, an electronics engineer. "We are being very blatant about the Colt in our advertising by describing it as a copy Apple – in fact better than the Apple", he said.

"And we feel that by people sending their money direct to Taiwan and the machine being sent air parcel post to the customer within 28 days, we are doing nothing to infringe the Apple copyright here.

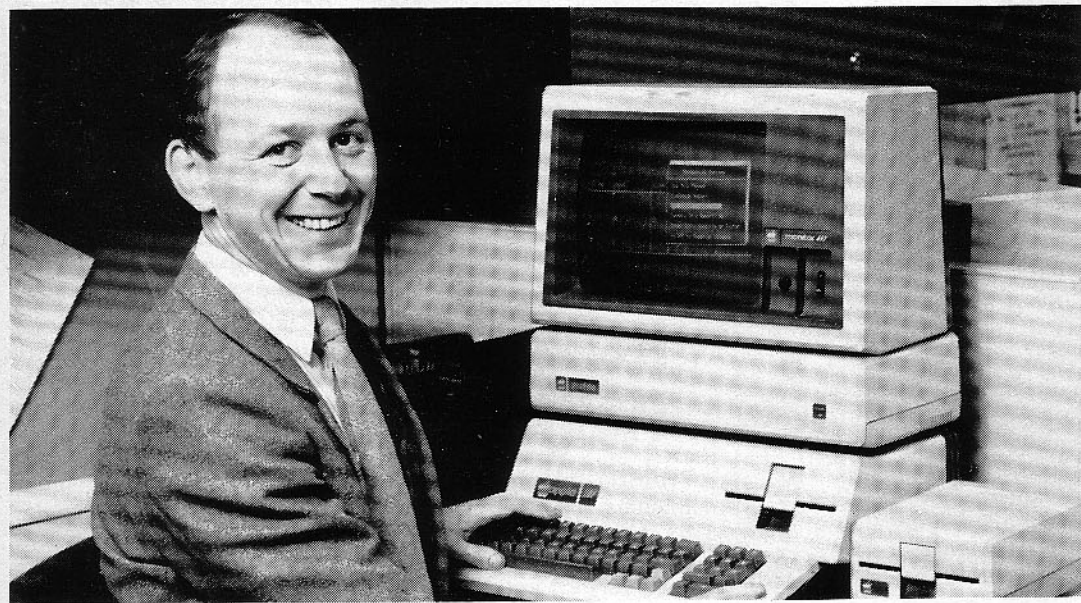
"If we took the money in this country it would be a different matter as we would be the seller and so get into trouble immediately".

Tony Roberts expects that the Colt PC will achieve sales of 100 a month initially and 200 a month later in the year.

But this will not happen if Apple UK has its way.

"It all sounds very dubious to me", said Mike Spring. "As far as we are concerned if he offers the machine for sale in this country then that's an offer of sale – and he can be proceeded against accordingly.

"And anyone who decides to send their money direct to a box number in Taiwan for one of these machines is really taking a risk, no matter what the price".



Making the Apple III sing...

APPLE UK has a new manager for its Apple III product line. He is Roy Stringer, formerly of U-Microcomputers in Warrington.

A qualified draughtsman, he entered the micro field with

Micro Digital and then joined U-Microcomputers one and a half years ago as a systems consultant.

He has a long association with this magazine as a regular contributor, and is also the

author of a variety of business programs.

"The III is a remarkable machine, although people do need some technical help to make it sing and dance properly", he said.

Prices ready to tumble?

PERHAPS the most significant and surprising feature about the just-launched Macintosh is the price.

Certainly it is the one that is most likely to worry manufacturers of machines like the Apricot, Sirius, IBM and Epson which are competing in the same market place.

Macintosh, incorporating a built-in 9in monitor and 3½in Sony disc drive, detachable keyboard and mouse, is expected to cost under £1,700.

Apart from other manufacturers' micros, other likely casualties could be Apple's own IIe and III, although Apple is adamant that its 8 bit and 16 bit families are complimentary rather than in conflict.

To keep the IIe and III competitive the company might have to drop their prices significantly.

The current price of the IIe, plus a 12in monitor and Apple II disc drive, comes to £1,419.

The Apple III – which has a disc drive built-in – plus an Apple III monitor, retails at £2,395.

How long Apple can maintain these prices in the light of the Macintosh price breakthrough is a question that will be in the minds of many Apple dealers – and their prospective customers – in the next few weeks.

● **The full story of the new Macintosh starts on Page 17.**

Windfall for schools

APPLE UK has announced a 40 per cent reduction in the price, for educational users, of its IIe and III micros and all related equipment.

The discount to schools, universities and colleges applies to software, printers, disc drives, monitors, 80 column cards and other peripherals.

Under the scheme an Apple IIe computer on its own will cost £499.



NOW APPLE RULES THE WAVES

DEVON cider, or scrumpy as it is called, is famous. Unofficially it is considered to attract people to take their holidays in South West England.

Now we've learnt that taking their place alongside the cider variety of Apples are hardware Apples – or more specifically, harbour Apples.

A pair of Apple IIes, each with 5mbyte hard disc drives, have been installed by Southern Computer Systems at Torquay

and Brixham to help the harbourmasters there.

And at Paignton the Apple is used to administer a thousand private moorings, several hundred commercial vessels, numerous dinghy racks, storage and crane facilities.

Traffic into the three Torbay harbours has increased dramatically over the past few years.

Before the advent of the Apples, officials were becoming bogged down with the adminis-

trative work involved in allocating boat moorings, recording owners and boat details, sending fee invoices, and monitoring the daily movement of vessels in and out of the ports.

Good news for the harbour officials is that they are now finding it easier to identify and trace specific owners or vessels.

The bad news for the boat owners is that annual invoices covering harbour fees will now be issued automatically.

.. and helps boffins keep an eye on environment

TEAMS of government scientists up and down the country are now monitoring the environment (hence the recent outcry over acid rain).

And one of those teams – in Bangor, North Wales – is using an Apple IIe.

At one time boffins used to tramp hither and yon over the countryside taking soil samples, holding wetted fingers up to the breeze, and other fairly unscientific activities.

But nowadays, of course, it is all computerised. Every so often a satellite rumbles overhead and spies out the land.

Instead of taking photographs, it converts what it sees to digital data which finally reaches the scientists on mag-

netic tape.

Trouble is there's rather a lot of it. A satellite picture represents an area 185km square – which converts to no less than 25mbytes of data.

This has to be analysed on an absolutely gynomous computer in Cambridge.

And there's a further complication. The data transmission rates for analysing these complex images are so high that the equipment cannot be used remotely.

So every time the boys from Bangor use the system they are in for a round trip of 400 miles.

That is why six months ago they installed an Apple-based system in Bangor to enable

them to do some of the work locally.

They use a 10mbyte Winchester disc system from Independent Computer Engineering of Ashford, Middlesex.

High resolution colour graphics are provided by a Digisolve VGP64 vector graphic processor and high-res colour monitor.

This enables the Bangor team to produce a high resolution image of whatever part of a satellite picture they want to study.

So they can do a great deal of their research locally without going all the way to Cambridge and using time on the mainframe.

Enter MACINTOSH

AFTER months of intense speculation throughout the computing world, Macintosh has finally arrived. It has everything all the experts were hoping Apple would include in their new micro. But much, much more as well.

It looks like a cut-down version of the revolutionary Lisa, which itself was launched exactly 12 months ago. And it contains most of the features that created such a sensation when Lisa was unveiled.

But it has even more refined capabilities – which is perhaps only natural as it has enjoyed an additional 12 months' development time.

Producing Lisa took four years and \$50 million. Developing Macintosh cost an extra \$25 million.

The result is a machine that stands head and shoulders above anything else on the market.

The few people outside

It's Lisa technology for everyone in a super pint-sized package

By DEREK MEAKIN

Apple who were permitted to try it out prior to its January 24 launch raved about it. And with good reason.

But even more remarkable is its price. Although this had not been finally fixed before Apple User went to press, it is expected around £1,500 to £1,700 – less than a quarter of Lisa's launch price.

Naturally, Apple are expecting massive sales and are gearing up for volume production. The Fremont factory, where it is being produced initially, will be turning out one Macintosh every 27 seconds.

Who will be buying Macintosh? The target customers are what Apple calls "information workers". They are aiming at people, not big corporations, and they say that is the big difference between their approach and that of IBM.

Macintosh is by far and away the most advanced micro at its price. Because it does not have the technical complexities of the conventional micro, it overcomes the fear element that still frightens many office workers.

An Apple executive gave me this description of Macintosh: "In so many ways it steps outside the existing boundaries of personal computers.

"It sits neatly on your desk, unobtrusive. It's just like your phone – you pick it up and use it when you need it.

"But it's very much a productivity tool. It helps you get through more work in a day – and far more efficiently. It opens the door to a mass of applications you may never have thought of before.

"And while it is principally designed for the office it is also a home machine as well.

"Because of its portability – it can be easily carried around in its canvas holdall – I can foresee many people taking it with them when they leave the office at night.

"What for? Well, they could be taking work home with them. Or they could use it for management training games.

UNLIKE Lisa – which is only available through a handful of specialised outlets – Macintosh will be sold by all Apple's 420 dealers in Britain.

Although machines specially designed for the British market will not be shipped from Apple's Fremont factory until April, a number of UK Apple dealers should have first supplies this month.

However these will be US-specification machines and will only be used for demonstration purposes.

Or other even more relaxing purposes".

When Macintosh goes on sale in Britain in April a wide range of Apple-produced software will be available – all on 3½in microfloppy discs.

They will include a word processor, spreadsheet, project management program, an extremely sophisticated drawing utility and a number of program languages. All will be priced at £99 or below.

But as was done so successfully with the old Apple II, every encouragement is being given to third party development of software.

More than 100 software houses – big names like Microsoft and Lotus – are already well ahead in producing Macintosh programs. And they are so enthusiastic about the machine that they are all going in for it in a big way. One major company is now devoting 50 per cent of its total R & D to Macintosh developments.

Initially, a third of all Macintosh programs are expected to be produced by key software houses in Britain and Europe, which should give a strong foothold in the profitable export markets that Macintosh will be opening up all round the world.

What Mac packs...

- ★ 16 bit machine based on the Motorola 68000 processor, using 32 bit technology
- ★ 192k memory, consisting of 128k RAM and 64k ROM.
- ★ 21k bit-mapped screen display.
- ★ 512 x 342 square-shaped pixels.
- ★ Built-in 3½in Sony microfloppy disc drive, with 400k disc capacity.
- ★ Detachable 128 character keyboard.
- ★ Built-in 9in black and white monitor.
- ★ Mouse pointing device for cursor control.
- ★ Two RS422 ports.
- ★ Ports for mouse and external disc drive.
- ★ Removable reset and interrupt buttons.
- ★ Complex sound output circuitry.
- ★ Weight: 22lb.
- ★ Size: 13in high, 10in wide, 10in deep.
- ★ Availability: Launched in USA on January 24, available in UK by end April.

MACINTOSH weighs 22lb, measures 10in x 10in with a height of 13in and contains the computer, screen, a 3½in Sony disc drive and various ports along the back.

A special screwdriver is needed to get into the machine, but there is no attempt to hide things – indeed the inside of the cover has the signatures of the design teams moulded into it. Some very famous names are represented.

The machine is designed for both the serious programmer/hardware specialist and the end user who just wants the right tool for the job.

For example, on the left side of the box, low down, are two buttons – the reset and an interrupt button.

These can simply be pulled away from the side so that their availability is lost to the user if you desire. But they can then be snapped back when required.

The interrupt button is interesting for programmers. It allows you to edit the program while it is running, and you can see your changes actually happening in the window.

Macintosh also has a detachable keyboard and a mouse. The keyboard is a full, 128 character, international standard unit which appears to be programmable.

There are a couple of extra keys labelled OPTION and COMMAND. These are used in conjunction with the mouse to change, for example, the displayed character set.

INSIDE MACINTOSH

MAX PARROTT takes a mouse tour round the latest micro to come out of Apple

Macintosh comes with a set of Greek and mathematical symbols as well as standard alphabetical characters.

All of the characters are displayable in a dozen or so fonts at different pitches, ranging from 15 to the inch downwards in size. You can also design your own character sets.

The machine suggests the best pitches for each font at the time of choosing but you don't have to accept this. Cursor movements are generally controlled via the mouse but not slavishly so. Keyboard control is also available.

The disc drive houses 3½in Sony microfloppies. We cannot really call them floppies any more as they come in rigid plastic boxes which they never leave.

The boxes are automatically opened on insertion into the machine and so data and programs are much safer than with the more familiar 5¼in

floppy discs.

Microfloppies have a metallic hub to ensure accurate registration.

In addition, Apple's unique formatting process allows 400k to be saved to each disc, which is more than the Sony standard of 320k.

Apple's own software is to be distributed unprotected and hopefully third party software houses will do the same.

Generally a data disc can be used as well as the program disc in the same drive since the operating system remembers what's where and what's going on.

For example the catalogs of the last used five discs are remembered and up to 30 shadows of their icons can remain on screen.

A system disc has a 120k operating system and housekeeping code on it and so has to be in the machine sometimes, but when it is needed you are prompted to put in the right disc.

Macintosh knows when you've done so and starts the drive up. It also ejects a disc when it is no longer required.

With Macintosh you receive two copies of the system disc, a blank disc, and a "tour of Macintosh" which, when run in conjunction with a cassette audio tape, guides you round the system.

I cannot believe that much guiding will be necessary since, like Lisa, once the modus operandi is understood you're away. There *is* a manual but you won't need it for a long time.

The computer itself hides behind and under a 9inch (glass, not anti-glare) monochrome screen with a resolution of 512

x 342 pixels, which are square-shaped for easy programming.

Macintosh could handle colour but Apple hasn't yet found a colour monitor with the required resolution at the required price.

However as Macintosh provides an infinite range of black and white shades which will be more than adequate for an office environment, colour isn't very important.

The system and its programs – applications might be a better word – are controlled via pull down menus, icons, overlapping and moveable windows – all under the direct control of the mouse.

It is driven by a Motorola 68000 running at 8MHz and has 192k of memory consisting of 64k of ROM and 128k of RAM.

The ROM looks after the I/O and screen handling generally via managers available to the programmer as something like 480 system calls.

There is a text editor in the ROM and a resource editor, enabling software to be personalised by the end user. For example, error messages can be easily changed.

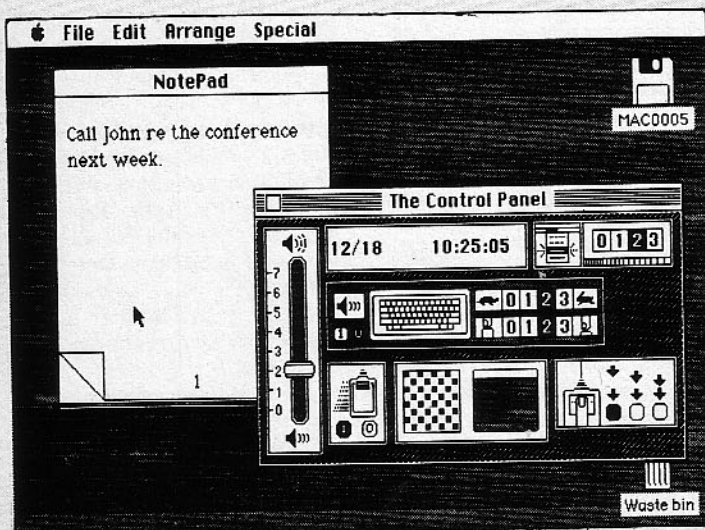
More importantly, the end user can have control over input. A spreadsheet application can be made to accept only numerics in one column or text in another but without programming.

Software therefore will use the full Lisa technology since it will be so easy to implement.

Furthermore, since the text editor is available with full, easy control over the pulldown menus and windows and icons, software writers should, in theory at least, be able to combine the flexibility of their own ideas into a common technique of program control.

Applications generally are written so that there is a hard and a soft part of the code. The resource editor allows you to change some parts to suit yourself – an interesting and useful idea.

Apple has produced a series of the more usual kinds of applications. For the programmer there is an assembler, debugger, Pascal, under which most software is expected to be



Pages or windows can be overlaid, enlarged, reduced, moved to foreground or background, saved on disc – or thrown in the bin!



Now Apple promise even more in '84

MACINTOSH is only the first of some remarkable products to come from Apple this year.

The company says that while 1983 was undoubtedly the Year of Lisa, it was also the Year of IBM — "who entered our market to help make it respectable".

Throughout 1984 Apple will be making giant strides to increase the lead they already hold, principally by concentrating on new products based on Lisa technology.

One forecast from a usually reliable source is that the same kind of mouse-and-window could well be adapted to Apple's existing 8 bit range.

And if that comes true it could mean a dramatic shot in the arm for the Apple IIe and III.

written, boasts all kinds of useful utilities around it and procedures within it.

Syntax is checked and part compilation, at least, occurs line by line when entered.

Also there is a structured Basic and Logo. The Basic uses labelled subroutines, doesn't

need line numbers and in common with the other languages can be edited in one window while being run in another.

There is also an interpreted Pascal available which, like all the other languages, is given extensive window handling cap-

abilities. Fortran, Forth and C are promised to appear very soon.

Apple's other utilities include MacWrite, a very powerful line-orientated word processor — one of the very few which genuinely exhibit on the screen what you will see on paper. The

only other that I've seen is LisaWrite.

MacPaint is a drawing utility that comes with MacWrite and is like LisaDraw but with certain differences.

For example, objects are not moveable but the bit map is, and the bit map can be exploded on

Now desktop computing has really arrived

HOW big is your desk? Can you make do on one that is 4ft by 2ft 6in? Or do you need an executive-sized 6ft by 3ft?

When you become a Macintosh user you'll be able to reduce your working area to an incredible 7 $\frac{3}{4}$ in by 5 $\frac{3}{4}$ in.

That is the size of the Macintosh screen. In theory all the work you normally do at your desk can be carried out on a screen that is smaller than a sheet of notepaper.

And much of the work can be done with your hand resting on the plastic mouse — without hardly ever needing to touch the keyboard.

You can inspect files, transfer documents, schedule projects, perform detailed financial planning, draw graphs, write reports, send telex messages, talk directly to other computers — the list of possibilities is virtually endless.

When Lisa was announced 12 months ago one of the features that created most comment was how easy it was to use.

A newcomer to computing needed no more than 20 minutes to understand what it was all about and have it up and away.

Macintosh is even friendlier. Fear of menus is a thing of the past. No longer are

you faced with lists of options to get you into parts of the programs — and have you forever leafing through the manual to find out what to do next.

When you switch on Macintosh the first thing you see on the screen is a picture of a disc. That asks you to insert one. You do so and your new "desktop" suddenly becomes alive.

Next to materialise are the ikons. These are graphic symbols that tell you what is available to you — the contents of your files, the different application programs, a calculator, a real-time clock... even a waste bin at the foot of the screen

into which you can deposit material you no longer need.

Documents you pull out of your files can be placed anywhere you want on the screen, and can be overlaid one on top of another.

In fact, anyone who habitually has an untidy conventional desk can just as easily clutter up his Macintosh screen.

But there's one big difference. Decide to have a tidying up operation and within seconds, the mouse scurrying around the screen will collect everything and put it neatly away.

Yes, with Macintosh the office of the future has really arrived.

the screen and adjusted bitwise using a lasso to capture the bits required.

A vast range of shades and patterns is available which can be extended almost infinitely since they can be edited by the user.

Drawings can be edited, therefore, at the highest resolutions and to this end there is a "rubber" and an "aerosol can" for brushwork.

The other available Apple applications are MacTerminal – for communications enabling VT 100/VT 52/teletype emulation – and MacProject which does what the name suggests.

There is also MacPlan – a spreadsheet similar to Multiplan.

What's in a name?

THERE have been lots of guesses as to the origin of the name Macintosh.

One theory is that it is named after a species of apple in the USA – big, red, juicy and, like Apple itself, very Californian.

But it most likely took its name from a codeword used during the development of the machine – **MAC, standing for Mouse Activated Computer.**

More than 130 software houses are already engaged in creating – rather than updating existing – software for Macintosh.

Apple has extended the Apple II philosophy of "openness" to Macintosh so we can expect a tidal wave of third party software and hardware add-ons.

This bodes well for the user for, with such competition, software quality and price will have to be excellent.

The games and general leisure software should be incredibly good and plentiful for a relatively low cost machine like this.

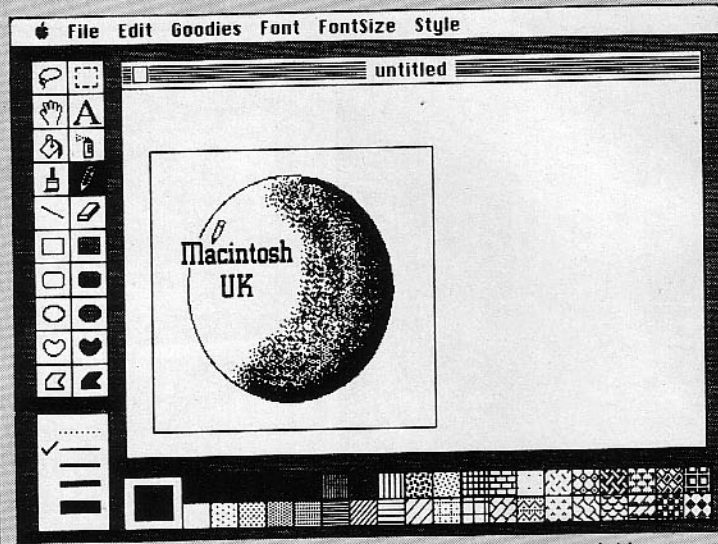
Macintosh, unlike Lisa, runs only one application at a time but swaps so quickly between them that to most users the difference will appear minimal.

The tools provided include help screens, a calculator, a real time clock with battery back-up, a clipboard for cutting and pasting between applications – but only one item at a time – and an eight page notepad for general use.

There is also a control panel to set your preferences over sound, keyboard sound response, repeat times for keyboard and mouse clicks and the flick rate of the menu bars.

The latter is the means by which Macintosh tells you what it thinks the best option is at any one time.

File handling and copying at



The aerosol option can be used to shade in areas, picking varying shades from the 38 boxes on the bottom of the screen

the immediate level is handled completely by icons. Pick up an icon, drop it on an image of the disc and it's saved.

The back of Macintosh sports a socket for attaching an external drive, one for sound and two identical RS422 ports working at speeds up to 1 Mbit/sec. Although identical, one is marked as a printer port and one as a modem port.

Apple's disc driver routines are in firmware but OEMs should easily be able to implement their own in software thus enabling a range of other drives to be supported. It won't be long before hard discs appear on Macintosh.

In the near future Apple is releasing Applebus, which will enable many more peripherals

to be added to Macintosh and which will interface to Lisa and others.

The computer's power consumption is low – the main logic board is rated at 8 watts. Total power consumption under normal operation is 50 watts, and about 65 watts with the drive running.

Macintosh supports its own printer, a dot matrix with friction and pin-feed with either a 10in or a 15in platten. It is fast, being capable of 120 characters a second and it dumps graphics more than twice as fast as the Apple DMP. It's known as the ImageWriter and will fit Lisa as well.

The windows in our illustrations were all produced with this printer.

Macintosh crashes the price barrier

FOR anyone buying a micro this year the choice is clear – it will either be a conventional computer or a Macintosh.

There are dozens of good conventional computers to choose from, ranging from cheap home micros to more sophisticated business machines.

But they all have one thing in common. They are all based on that pioneering, handbuilt Apple that was born in a Californian garage back in 1976.

Of course they all boast minor, individual refinements to the original concept. But there

was no real breakthrough in design until February of 1983 when Apple unveiled Lisa. Mouse technology had arrived.

In the last 12 months a number of manufacturers have also announced mouse enhancements. But they are doing little more than cashing in on a name and an idea that has taken the computing world by storm.

None of them has got anywhere near to emulating the truly revolutionary concept that was built into Lisa – and is now available in Macintosh.

The only thing wrong with Lisa was its price. Its launch

price tag was £7,999, including a selection of software. This was reduced to £6,500 last September, and it fell to £5,500 in December, when Apple decided to sell it without the software.

Even so sales have been few – except to other manufacturers who couldn't wait to get their hands on one and find out what made Lisa tick.

Everyone in the trade expected that Macintosh would break that price barrier. What they didn't anticipate was that it would get down to anywhere below the £2,000 mark.

What this means is that a

number of leading business micros will now be not only facing formidable competition on the technical front, but on price as well.

A retail price of £1,700 would make Macintosh £825 cheaper than the Sirius, £690 cheaper than the IBM PC, £295 cheaper than the Epson QX10 and £1,014 cheaper than the DEC Rainbow.

One thing seems certain. The arrival of Macintosh heralds an era of new style computing for many thousands of new users – and a period of intense price cutting as well.

TWO years ago an Apple II was given a key role to play in a successful £4.25 million take-over bid by a major British clothing manufacturer and retailer, Steinberg plc.

BRIAN PEMBERTON examines how that Apple has been used and explains practical ways in which small businessmen can benefit from a big company's experience.

He says Steinberg's Apple represents one of the most sophisticated, thoughtful and thorough financial modelling applications he has seen.

STEINBERG'S equipment at its Milton Keynes HQ includes impressive Hewlett Packard and DEC mainframe computers.

However, it was an Apple II that provided the information and financial modelling capabilities which, in 1982, facilitated the management's decision-making in the successful £4.25 million take-over of Redhouser Securities, whose subsidiary, Claremont Garments, manufactured a complementary range of clothing.

Together with an off-the-shelf Visicalc program, the Apple became a sophisticated financial modelling tool which, in the right hands, was able to produce in a short time the information on which a substantial business decision was based and followed through.

The Apple II, which was supplied to Steinberg by Milton Keynes Computer Centre, is well known for its versatility and ease of operation.

But what does this really mean to the business user?

Looking at the use Steinberg has made of its Apple gives some useful indications of the nature of the machine as a piece of business equipment — its friendliness and its potential sophistication.

Regular readers will have seen a number of articles relating to the Visicalc program, but for those who are not familiar with it a brief

An invaluable aid for small (and large) businesses

description may be in order.

Visicalc is essentially an electronic worksheet where the relationships between rows and columns of numerical data are controlled by a combination of formulae held in the computer's memory and entered data.

This means that the worksheet is not only a quick calculator but becomes a modelling and forecasting tool.

Changing data in any one column/row will automatically change the data in any other related column/row, and changing formulae themselves changes the relationships between the variable data.

You can alter quantities to see what happens (playing 'what if...') and also change easily assumptions, relationships and parameters that provide the framework for the data.

Steinberg's Apple, I would wager, represents one of the most sophisticated, thoughtful and thorough financial modelling uses that you are likely to find with this type of equipment.

It was acquired for this purpose by the managing director, Mike Harvey, who, with accountant Tony McManus, has filled some 50 discs with Visicalc models of

various types.

These provide 'fine tune' management accounting information, leaving the powerful number-crunching accounting to the mainframes — which consequently supply summary information to the Apple to use in its models.

The first job the Apple was asked to do was to produce various financial models to support the proposed reorganisation and Department of Industry grant funding of the company's factory in West Auckland.

Visicalc was used to analyse performance over recent years and to make cost and sales performance projections for net operating costs and profits. Additional variables were put in to determine effects of different levels of investment and interest.

Unit sales prices were, for a number of reasons, fairly unpredictable, so the models were used to show what would happen on various price prognoses. The results indicated that fairly high levels of investment were needed with high productivity gains.

A comparative model was drawn up to show the effects of a hypothetical move to South Wales. The actual result of all this planning were new premises in West Auckland, a

Department of Industry grant, and a new set of budgets.

Encouraged by this successful experience, the Apple quickly came to be used for all sorts of management accounting with variations on the original models — gaining acceptance as a quick and easy aid to management decisions in both day-to-day work and monthly reports.

This culminated in the enormous modelling exercise undertaken before the take-over.

Steinberg's organisation is fairly complex. It involves free standing shops, shops-in-shops, jobbing, wholesaling and contract manufacturing.

Different sections behave very differently and require different models to make sound projections and conclusions.

And of course many more variables had to be taken into account to assess the effect, and therefore the feasibility, of the take-over.

Many models were produced to cover all aspects of business activity and performance — costs and selling prices, capital expenditures, overhead and divisional expenses projections, consolidated profit and loss accounts, risk analyses, cash flow forecasts, source and application of funds statements, depreciation schedules, stocks and stock provisions, factory plans, range plans, and so on.

Moreover, printouts of summary models, which presented the information in a readily appreciated form, were actually used in the boardroom presentation for the take-over — which as we have seen was successfully completed.

If all this seems a bit mind-boggling to the small business Apple user or potential user, let me say right away, as a small businessman myself, it is mind-boggling — especially if you look through the thick folders of printouts in Tony McManus' office!

But the essential points of Steinberg's application which we can draw lessons from are quite simple.

Firstly, the company knows what it is doing. This is very

important. Steinberg knows its industry, its own operation and the sort of information needed and how to get it.

Its Apple II is a tool in very capable hands and staff have pushed it to a high level of sophistication.

The models produced by Steinberg are not really mind-boggling. They only appear to be so when considered in isolation from this knowledge of the industry, the operation and what the company is trying to do.

The lesson? Know your industry, your operation and what you want to achieve and the Apple can help you enormously.

Secondly, a good deal of experimentation was required before the most useful models were produced. Steinberg was either lucky or, more probably, far-sighted, enough to have an accountant and MD who were knowledgeable in the sense described above and who could put together on the computer screen the things they knew in their minds.

They are not skilled computer operators in a professional sense, and are by no means systems analysts.

Again, the Apple with Visicalc provided a quick and easily used tool to play around with the data. It works in the same way that a person would work on paper, and this seems to be the factor which makes Apple equipment so accessible, so friendly.

The lesson? You don't have to be a computer whizz to get the best out of the Apple, it just makes it look as if you are!

Thirdly, all this modelling would have been done anyway, with or without the Apple. Pre-Apple it was done manually.

This is important too because it shows that good business practices are enhanced by using such equipment, not created.

Of course such practices can be improved – in the Steinberg case, for example, the modelling exercise indicated the need for several new types of information to be gathered as a matter of course, but the essential ingredients were there to begin with.

The Visicalc fragment below is one of a large number put together with the Apple II at Steinburg. Fifty discs were needed to hold them all.

Note: All figures are fictitious.



SOURCES	C.G.L.	ALEXON	CONTRACT	TOTAL	C.G.L.	ALEXON	CONTRACT	TOTAL
PROFIT BEFORE TAX	1858	234	481	3573	2370	577	584	
ADJUSTMENTS								
:DEPN ON NEW F.A		203	74	277		745	133	
:GRANTS REVENUE		257	61	318		300	96	

The lesson? You'll get more out of your Apple if you realise that on its own it is useless – just as a hammer by itself is useless, and worse than useless if you just go about hitting any old nail with it.

For many small businessmen who either have or are thinking of getting an Apple this is where good advice from the supplier comes in handy.

A good supplier will not only know his equipment and

software intimately but will be able to appreciate and develop with the customer those very business practices which can be enhanced by the equipment.

The better he does this the better the use of his machinery will be, the happier will be his customer, and the more sales will result. What more incentive?

Not all business applications of the Apple will be as

sophisticated as the Steinberg case, but it does provide a very good example of the principles involved.

No matter how simple the job, if these principles are followed the application is much more likely to achieve the best possible results.

Just remember – when you get to the stage of making multi-million pound take-over bids, your Apple will still be there to help you!



Mainly for the children, but...

... you'll probably pick up a few hints yourself from this new series

IF you've been into a large bookshop recently, the chances are you were impressed by the sheer number of computer-related books on the market.

If you've ever looked for books for your children, assuming they don't already know more than you, you might be interested in a series from Usborne Publishing.

There will be a dozen or so in the series when it is complete, of which about half are available now. So far I've seen five and I'm looking forward to seeing more.

The first I encountered was the "Introduction to Computer Programming: Basic for Beginners", which I bought for my young daughter.

I must admit I'd been largely influenced by the attractive, colourful presentation and extensive use of illustrations — particularly the little robots and space invaders.

My first inkling that something had happened was when my wife suddenly started spotting bugs in my programs.

I thought that maybe Pebble Mill At One had been teaching Basic in between the recipes and interviews, but it transpired that she'd picked up Janet's book in an idle moment.

Following this sobering experience, I contacted Usborne and they kindly sent some more of the series for review.

The series is concerned with micros in general, but the Apple is one of the specific machines catered for in the program examples.

In fact, the Introduction to Computer Programming proudly proclaims "No com-

puter needed" on the front cover, and the examples are almost machine-independent. I say "almost" because occasionally it is necessary to point out that some machines have slightly different commands (for example, Newline/Return/Enter).

At a more elementary level, "Understanding the Micro: How It Works and What It Can Do" takes you slowly through the micro jungle, from Ascii to ZX81.

It's written in the same simple style (although the occasional big word like terminology creeps in), and the colourful robots and space invaders once again brighten the page.

The other three books I've seen are all concerned with games. Of these, "Computer and Video Games: How They Work and How To Win" is probably the least interesting if you already have a machine.

As the title suggests, it covers the whole range of micro-driven games, and the suggestions on "how to win" obviously only make sense if you know the game. However, it made interesting reading for some adults I showed it to!

The final books are companion volumes, "Computer Spacegames" and "Computer Battlegames".

The distinction between them isn't always clear since the former is often concerned with battling and the latter sometimes involves a fight with aliens. In fact, the introductory pages are identical, which must say something about the overlap.

Both contain programs which run not only on the Apple but also on the ZX Spectrum, ZX81, BBC, TRS-80, Vic and Pet.

The listing is usually the ZX81 version (and many run in 1k), but lines which need changing for the various machines are marked and appropriate alternatives are listed at the bottom.

Most programs have notes advising what each line or section does. This means that a beginner may well pick up some useful programming tips just by reading them.

Many of the games also have suggestions for how you might make them harder, and some also contain a puzzle corner.

Answers to the puzzles are given at the end of the books.

All the listings we have entered have worked, although there has been the occasional problem.

For example, because zeros are printed without a slash there is sometimes a possibility of confusion with the letter O.

Also, being ZX81 listings, they terminate with 'Stop' which produces a "Break in line xxx" message on the Apple.

However, the introduction warns that this is how the programs end, and the Basic summary points out that computers other than the ZX81 can use 'End' instead.

Because most machines have idiosyncratic graphics systems, almost all of the games rely on use of text. However, both books contain one game which uses graphics.

In these cases, it has been necessary to list separate ver-

THE books reviewed here, all published by Usborne, are:

Introduction to Computer Programming: Basic for Beginners (£1.65).

Understanding the Micro: How It Works and What It Can Do (£1.65).

Computer and Video Games: How They Work and How to Win (£1.65).

Computer Spacegames (£1.99).

Computer Battlegames (£1.99).

Understanding the Micro, Computer Programming and Computer Spacegames (£4.50).

sions rather than try to annotate a common version.

Talking of common versions, both books contain an identical 10 pages at the end of the games.

These give suggestions for adding to the programs, writing your own programs, and a summary of Basic.

There is also a conversion chart showing how some of the commands are implemented on each machine. For example, the Apple's familiar 'Home' becomes 'CLS' on most other machines.

You may feel that since the two books have a common 12 pages out of 48, it is not worth buying both.

On a more positive note, each contains 13 programs and only costs £1.99.

Even better value is the composite volume which contains Understanding The Micro, Computer Programming, and Computer Spacegames.

All in all, I think that these books from Usborne are reasonably well written and very well produced.

The style is attractive, and I shouldn't be surprised if they found their way into quite a few primary and secondary school libraries.

The catalogue describes them as suitable for "anyone aged 11 and over", but I think they are quite good for much younger children than that.

With micros being used in primary schools, it's nice to see books which are attractive to this age group.

Cliff McKnight



PRODOS was designed to facilitate an easy transition for users and applications from DOS 3.3 while providing a better environment for new, higher performance, enhanced capability, more professional applications.

Such applications might require the ability to utilise the interrupt capability of the Apple II, faster disc accesses, or a device independent disc driver not available with either DOS 3.3 or Apple Pascal.

To be specific, ProDOS provides:

- Much faster disc access than DOS 3.3.
- More powerful file handling facilities.
- Improved hardware support.
- Device independent disc drivers.
- Better development tools, particularly for assembler programmers.
- A transition path from DOS 3.3.
- High degree of compatibility with SOS on Apple III.

The prerelease version of the system issued to licensed software developers consists of two software discs and four technical manuals.

One of the discs contains a copy of the operating system and the set of software development tools.

The other disc contains a collection of examples of how to exploit the features of the system using Basic. The manuals are final draft copies of:

- ProDOS user's manual.
- Basic Programming with ProDOS.
- The ProDOS technical reference manual.
- ProDOS Assembler tools.

At the time of writing it is not known how general release versions of the system will be packaged.

The Assembler tools manual leads me to believe that the Assembler tools software will be issued on its own disc rather than on the operating system disc as it has been with the review copy.

Furthermore, it became clear after reading the manual that the other components issued with the developer's system will probably be packaged differently.

In particular, rather more extensive use of menu selection

ProDOS

KEITH LANDER examines the latest disc operating system from Apple

techniques will most likely be employed throughout the general release system. More about this later.

The manual explains how to use the ProDOS user's disc. This contains a ProDOS system program called the Filer which is used to manipulate files and volumes of programs and data. It is analogous to the Filer in

system program rules.

The Filer is a system program, as is the package that supports all the Basic operating system commands.

Finally, ProDOS Assembler tools covers the software to be released on the ProDOS Assembler tools disc.

Essentially, this consists of an upgraded version of the

ProDOS is the new disc operating system for the Apple II Plus and IIe. First copies of the system were made available to software developers during the final quarter of 1983 and this review is based on practical experience of using one of the pre-release versions.

Apple will be supplying ProDOS on a disc with all new Apple IIes, instead of the traditional System Master DOS 3.3.

In a later article KEITH LANDER will discuss the system in greater depth from a software developer's viewpoint.

the Pascal operating system. It also contains a program called Convert for converting DOS 3.3 files to ProDOS form and vice versa.

This is similar to the Muffin program used to convert DOS 3.2 to DOS 3.3.

Basic Programming with ProDOS explains the ProDOS facilities for supporting Basic programs. This is all held on another disc called PRODOS.

The ProDOS technical reference manual contains an in-depth discussion of a wonderful thing called the MLI - Machine Language Interface.

If you are familiar with the RWTS subroutine in DOS 3.3 then you will have some idea of what the MLI is likely to be.

The manual also describes the steps you need to follow if you want to produce a system program - that is, any program that makes calls to the MLI and that adheres to a set of standard

EDASM editor/assembler/loader system sold as part of the Applesoft toolkit, together with another goodie called Bugbyter which is a very powerful interactive debugging tool for Assembler programmers.

It takes an awful lot of the pain out of debugging machine code programs.

Altogether, in A4 draft form, these manuals contain more than 700 pages of information. The final manuals will presumably be in A5 format using a smaller typeface and should therefore occupy less space.

Nonetheless, it's all good well-structured documentation and, no, you don't have to read it all before you can start.

In fact, I found ProDOS to be one of the easiest systems to get into for software development purposes.

ProDOS has been designed to run on any 64k Apple II, and if

Applesoft is required, it must be installed on motherboard ROM. In other words, you won't be able to use ProDOS on the older 48k Apple II.

Unlike DOS 3.3, ProDOS-based applications will be able to access and utilise the capacity and speed of any disc drive conforming to Apple Disc protocol without having to be upgraded when a new storage device becomes available.

It is incidentally rumoured that a version of the Profile hard disc system will be made available on the Apple II.

In addition to being device independent, ProDOS will look for and use a Thunderclock, or any Thunderclock lookalike, to automatically time and date stamp files.

Closely related to this is the defined interrupt protocol provided via the MLI. This feature is useful for networked semi-realtime data acquisition or simple background-foreground processing such as print-spooler support.

ProDOS stores a variety of configuration data in its Global Page at the top of the 48k bank.

An application can examine this to find out which peripheral slots contain ROM, or if an 80-column card following the UCSD protocol for external terminals is in slot 3, or a Thunderclock is present.

It can also find out what machine it is running on and how much store is available (ProDOS supports the IIe extended memory card as a RAM disc).

The biggest difference between DOS 3.3 and ProDOS is the file structure.

ProDOS supports a hierarchical file structure identical to that of Apple III's SOS, and more powerful than that of MS-DOS 2.0. This is in direct contrast to DOS 3.3 and Apple Pascal which only support a single level file structure.

A hierarchic or multilevel file structure allows you to organise your files in a more logical fashion.

Whereas all the files on a DOS 3.3 volume are referenced from a single catalog, ProDOS allows you to create files which are themselves catalogs.

These catalogs, or Directories to use the correct ProDOS

jargon, may contain both files and further directories.

The first directory created on a volume when it is formatted is called the volume directory, and may contain up to 51 files. All other directories on the volume are known as subdirectories, and the number of files in a subdirectory is only limited by the size of the volume.

ProDOS filenames are limited to 15 characters consisting of letters, digits and fullstops. Names must begin with a letter. In order to reference a file in a subdirectory you have to construct the file's pathname. This is obtained by simply stringing together the names of the directories leading to the file from the volume directory.

the pathname to a subdirectory to the system Prefix. The Prefix to the file PRODOS in the above example is /MY.DISC/ARTICLES/APPLE.USER.

If I now simply use the name PRODOS, the system will append the prefix to the front before looking for the file.

If you begin a pathname with a slash ProDOS assumes that the first name in the path denotes a volume directory, and it looks for that volume among its set of disc drives.

If you are wondering why you must go to all this bother – after all, directories take up valuable space that could be used for storing useful data – then consider how many 10k data files you can store on a 5mbyte Profile and imagine looking

communicate with the ProDOS file handler via the normal Basic file commands OPEN, CLOSE, READ, WRITE and APPEND (which can be used with random files).

Machine code programs must use the MLI documented in the ProDOS reference manual.

Whether you use Basic or machine code, one thing you will notice is the disc transfer rate. ProDOS transfers data to and from a disc drive eight times faster than DOS 3.3 and it shows.

One application of mine written in Assembler takes half an hour to assemble under DOS 3.3 and about four minutes under ProDOS. That's quite an improvement and one I greatly welcome.

Before moving on to talk a bit about Basic and compatibility I must mention the ProDOS Filer and, in particular, the way it interfaces with the user.

As I said earlier, this is a ProDOS system program that you can use to perform file and volume housekeeping activities such as formatting volumes, copying files, displaying directories, and so on.

In common with all other ProDOS system programs, the Filer is menu driven. If you invoke the Filer you will be presented with a list of options and asked to choose one (by typing the first letter of the option you want).

One of the options allows you to Quit the Filer, and another, called Tutor, will display information about the other options in the menu.

The remaining options allow you to choose between File commands and Volume commands.

If you choose either of these options another menu will appear, listing the set of commands available to you. Again, each list contains a Tutor option to save you burrowing in the manuals to find out what to do next.

You can always leave a sub-menu and return to the one you came from by pressing the Escape key. In fact, the notable thing about ProDOS is the degree of consistency of the user interface between the various system programs.

What of Basic? First, there is no Integer Basic available under ProDOS. Applesoft loses a few commands such as INIT, FP, INT, MON and NOMON. However, it gains some new commands such as a new CATALOG command that allows you to specify file type. It also supports 40 and 80 column screens (see Figure 1).

There is also a built in CHAIN command that enables you to chain together Applesoft programs without worrying about preserving variables.

There are new commands for supporting the new file structure and commands that let you write all your variables to disc and bring them back again. Many existing commands have additional parameters.

One final goodie – there is a new garbage collector for string variables that runs in seconds, not minutes.

In general, DOS 3.3 Applesoft programs that use only standard Basic and no PEEKS or POKES to DOS may run without modification under ProDOS. Apple have certainly worked hard at trying to maintain a high degree of compatibility between the two versions of Basic, a task that is a lot harder than you might think.

One point to be aware of – ProDOS does not support Pascal. Apple says that we should think of ProDOS as the successor to DOS.

In conclusion I would say that ProDOS is a welcome successor to DOS 3.3. From the point of view of the serious applications developer it presents a much more powerful and flexible interface to the system resources than DOS could ever have hoped to do.

It should instil new life into a machine that many would say was due for the scrapheap. Certainly when Apple releases a Profile II during the next few months, a viable upgrade path from the II to the III will be far more feasible with ProDOS.

MY.DISC
RECIPES

FRUIT.CAKE
APPLE.PIE

ARTICLES

APPLE.USER

FORTH
PRODOS

COMPUTER.WEEKLY

LETTER

CORRESPONDENCE

XMAS.CARD.LIST

FINANCE

BANK.STATEMENTS
BILLS

Figure 1

Figure 1 shows how one might organise a set of personal files on a volume called MY.DISC. The volume directory might be called MY.DISC and contain four subdirectories called RECIPES, ARTICLES, CORRESPONDENCE and FINANCE.

ARTICLES contains two subdirectories called APPLE.USER and COMPUTER.WEEKLY. The remaining names in the structure denote data files.

Suppose I want to reference the file PRODOS. I would write its pathname as /MY.DISC/ARTICLES/APPLE.USER/PRODOS using slash as a delimiter.

If all this seems a little longwinded, then rest assured that there is a shortcut called the Prefix.

ProDOS allows you to assign

through a directory listing of all the file names!

Then you might begin to see the main purpose of structuring data this way. What's more the bigger discs will come – after all, 640mbyte discs are used on some mainframe computers.

In addition to supporting directory files, ProDOS provides a number of predefined and application definable file types.

These can be easily processed by applications programs written in either Basic or machine code.

The Basic programs com-

NAME	TYPE	BLOCKS	MODIFIED	CREATED	ENDFILE	SUBTYPE
/BASCOPI						
*PRODOS	SYS	29	1-SEP-83 0:00	3-SEP-83 11:50	1436	
*BASIC.SYSTEM	SYS	21	1-SEP-83 0:00	3-SEP-83 11:47	10240	
STARTUP	BAS	7	22-JUL-83 11:41	22-JUL-83 0:00	54	
HELP	DIR	1	22-JUL-83 0:00	22-JUL-83 11:27	512	
DIRECTORY	DIR	1	28-MAR-83 0:00	28-MAR-83 0:00	512	
PRACTICE	DIR	1	15-JUL-83 17:51	28-MAR-83 0:00	1536	
PROGRAMS	DIR	1	29-AUG-83 10:07	28-MAR-83 0:00	512	
DATA	DIR	1	29-AUG-83 17:53	28-MAR-83 0:00	512	
EXTRAS	BAS	8	17-JUL-83 0:00	17-JUL-83 0:00	3273	
POSTAGE.RATES	BAS					
BLOCKS FREE: 100			BLOCKS USED: 172		TOTAL BLOCKS: 200	

Figure 2

WHEN computers were first invented 35 to 40 years ago, they were generally used to speed up complex numerical calculations, which took human "computers" weeks or months.

Typical uses involved the calculation of planetary orbits, the trajectory of artillery shells, the statistical significance of early returns in an election, stresses in aircraft frames, or the reactions required to detonate a nuclear weapon.

Most of the early programmers and designers of programming languages were mathematicians, numerical analysts.

The first high level computer language, Fortran (literally **For**mula **Tran**slation) was designed by a mathematician, John Backus, to allow engineers to simply translate their equations into programs.

But by the mid-1950s, another group of researchers was emerging, interested in using computers not as giant calculating machines but rather as devices for manipulating symbols.

In particular, they were interested in natural language, and the automatic translation of one language (say, English) to another (say, Russian).

An American, John McCarthy, realised that natural languages could not easily be reduced to mathematical formulae of the type encouraged by Fortran (or its descendants), which allocated fixed blocks of memory, and whose fundamental operations were arithmetic and Boolean, operating through assignment statements (Let $X = X + 1$), and loops.

These languages maintain a very clear distinction between data and instructions for dealing with the data.

McCarthy argued that programs dealing with natural language would operate more effectively using a single structure, a list, without any clear distinction between data and instructions, and with a minimally structured employment of memory.

He called the language which would operate in this way Lisp, for **LIST** Processing language.

As I explained in an earlier article, Logo is a version of Lisp.

There are differences, and Apple Logo is only a tiny subset

List-en to the Logo language

CHRISTOPHER ROPER demonstrates Logo's power as a tool for manipulating human language – and sets a problem

of what highly developed Lisp programming systems can achieve on specially designed hardware.

I have also mentioned that the crucial list operators are **FIRST** and **BUTFIRST**. Let's look at how these are used.

Lists in Logo are conventionally contained in square brackets, []. The most basic list is the empty list, without any elements.

A list is made up of words, numbers and, you've probably guessed, lists.

We will begin by creating a list.

MAKE "NOUNS [FRUITS VEGETABLES FLOWERS]

We now can apply our operators

PRINT FIRST :NOUNS
and LOGO would return
FRUITS

Remember that the use of quote marks " before a word in Apple Logo signifies that it is the name of a variable (or of a procedure or of a list).

Whereas a colon : in front of a word denotes the value of the variable or the contents of a list.

PRINT BUTFIRST :NOUNS
would have returned
VEGETABLES FLOWERS.

That is essentially all there is to it. **LAST** and **BUTLAST** operate in exactly the same way on the tail end of the list.

FPUT and **LPUT** add items to the beginning and end of the list, respectively. So:

FPUT "TREES :NOUNS

followed by

PRINT :NOUNS

would return:

**TREES FRUITS
VEGETABLES FLOWERS**

A simple example of how this might be used can be seen in the Logo procedure **REVERSEL**. (See Figure 1.)

If we then apply this to our original list:

PRINT REVERSE :NOUNS
we will see:

**FLOWERS VEGETABLES
FRUITS TREES**

There are aspects here that will begin to indicate the power of the language, but which will also make your head hurt. The first point to note is the use of **OUTPUT**.

This means that this procedure returns a value to the calling procedure.

If I had simply typed:

REVERSE :NOUNS
the response would have been:
**I DON'T KNOW WHAT TO
DO WITH FLOWERS
VEGETABLES FRUITS
TREES**

The second point is that the second line (**IF EMPTY :X [OUTPUT :X]**) simply stops the recursive procedure, which would other-

wise keep calling **REVERSE** in the last line, which is the one that does all the work.

EMPTY is a conditional which returns true if applied to an empty word or list.

This last line takes the first element and puts it to the end of the reversed remainder of the list.

It is able to complete the process only when all four elements have been stripped out and at its fifth call **REVERSE** encounters the empty list.

I can almost hear you thinking nostalgically about Basic. Well, while I was in America, I interviewed Alan Perlis, who is one of the world's greatest authorities on programming languages, and the author of a delightful collection of epigrams on programming.

Sample, "Bringing computers into the home won't change either one, but may revitalise the corner saloon".

Professor Perlis, now teaching at Yale University from a wheelchair, told me of an instructor at another university setting a problem for first year students:

"Take a string of letters and numbers, of any length. The first three characters, however, must be digits.

"Write a program to strip off the first three characters, reverse the remaining characters in the string, and print them out as many times as indicated by the three digit number stripped off the front of the string".

Thus if the original string were **005APPLE**, the output should be:

**ELPPA ELPPA ELPPA
ELPPA ELPPA**

Right? Right. The students were working in the IBM language PL/1, but it could equally well have been Pascal or Fortran or Basic.

They produced programs varying between 60 and 80 lines in length, with on average 13 syntactical errors, which meant a long debugging session

```
TO REVERSE :L
IF EMPTY :L [OUTPUT :L]
OUTPUT LPUT FIRST :L REVERSE
BUTFIRST :L
END
```

Figure 1

to persuade the program to work as intended.

Perlis went back to his own first-year students and asked them to write the equivalent program in APL.

None of the programs required more than five lines of code, or contained more than three errors.

Now you may think this goes to show only that Yale students are smarter than the rest, but Perlis assured me this was not the case.

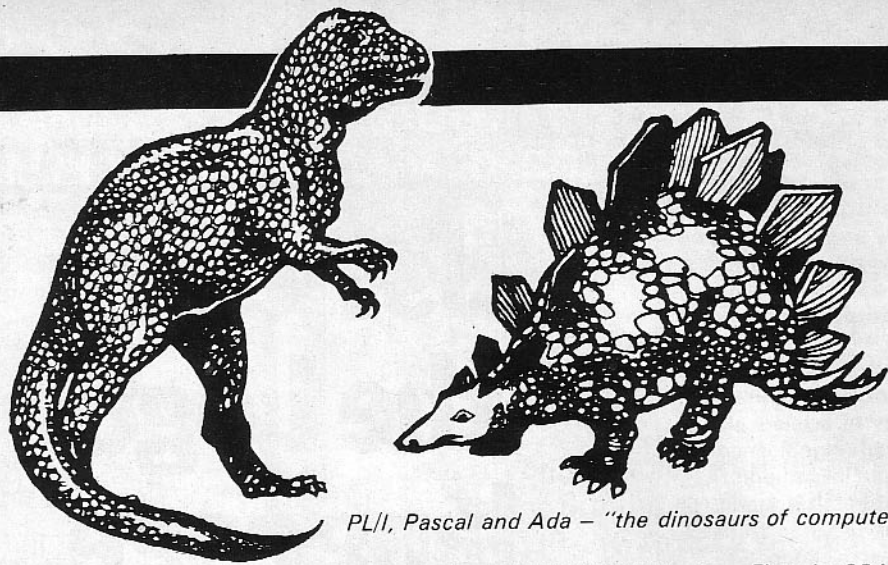
Although he played a large part in the development of Algol, the ancestor of PL/1, Pascal and Ada, he describes them as "Dinosaurs, with a lot of armour plating and very primitive nervous systems". He has turned completely to APL and Lisp.

I wondered as I listened to him how long it would take me to write such a program in Logo. It turns out that either I am not as smart as the Yalies, or Logo is not as efficient as APL.

It took me 39 lines of code, broken down into six procedures.

The number of lines of code drops to 27 if you don't count the names of the procedures or the word END.

There are another two lines to make sure the first three characters are digits, and six lines asking the user to type in a



PL/I, Pascal and Ada - "the dinosaurs of computer languages"

string.

So we are still a good deal better than PL/1. It took me half an hour to write and debug the procedures. (See Figure II.)

All this does is give the program a string of the correct type to work with. Perhaps Perlis assumed the string was already there.

The hard work is done in PROCESS and REPROCESS. (See Figure III.)

I tried to consider all the different types of string that might be typed in. Some would have spaces, and come in the form:

234 APPLE COMPUTERS

Others would be simple alphanumeric strings in the form:

123A5PLSON

PROCESS deals with the first kind, and its first line is a test to see if it conforms to the type. If

asks whether FIRST :STRING is a number.

In the example given above, this is 234, and is indeed a number (if it was not, the procedure REPROCESS would have been called).

This gives us the number of times the string ultimately has to be output.

However, it might equally well have been 234567. So before proceeding to create our REPEATER variable, we apply a little procedure called REDUCE, which is set out below.

REDUCE simply strips off all but the first three digits.

Next we need to strip the repeater digits off the string, before reversing it ready for action. Now I have two possible types of string to cope with at this stage, one with three digits only, the other with more.

There cannot be less or it would not have satisfied the original condition of starting with three digits.

The first case is simple, I
MAKE "STRING BUTFIRST :STRING

In other words, :STRING loses its first element. The second case is harder.

MAKE "STRING FPUT BF BF BF FIRST :STRING BF :STRING

Let us imagine that :STRING was:
234567 APPLES

First is 234567 and BUTFIRST (BF for short) is APPLES.

So BF FIRST :STRING is 34567; and BF BF BF FIRST :STRING is 567.

FPUT simply means FIRST PUT, so I take 567 and put it in front of BUTFIRST :STRING, which is APPLES. Easy.

It does make your head spin when you begin, but quite soon it's the rest of the world that seems crazy.

We then want to reverse our stripped down string.

Our REVERSE procedure described above won't do. It would simply switch the order of the words and numbers in the list, and would print out APPLES 567, whereas we are looking for SELPPA 765.

So we need two REVERSE procedures, one (REVERSEL) to reverse the elements in a list, and the other, REVERSE, to reverse the letters in a word or the digits in a number.

Once this has been done, it's all plain sailing. You repeat the number of times signified by :REPEATER the order to type :STRING, which now runs backwards. (See Figure IV.)

REDUCE is a typical small Logo procedure, very neatly chopping a number down to size.

As soon as repeated division by ten reduces it to a three figure number, it returns the answer to the recalling procedure.

INT, short for integer, drops all the numbers after the decimal point.

REPROCESS copes with strings in the form 123APPLES and sends you back to the beginning if the first three characters are not digits. Otherwise, it is not so very different from PROCESS. (See Figure V.)

Even now, after extensive playing with Logo over the past

```
TO SETUP
PRINT [TYPE IN A STRING]
PRINT [OF UP TO 255 CHARACTERS]
PRINT []
PRINT [MAKING SURE THE FIRST]
PRINT [THREE CHARACTERS ARE DIGITS]
PRINT []
MAKE "STRING READLIST
PROCESS "STRING
END
```

Figure II

```
TO PROCESS :STRING
TEST NUMBERP FIRST :STRING
IFFALSE [REPROCESS :STRING STOP]
MAKE "REPEATER REDUCE FIRST :STRING
IF FIRST :STRING > 999 [MAKE "STRING FPUT BF BF
BF FIRST :STRING BF :STRING]
[MAKE "STRING BF :STRING]
MAKE "STRING REVERSEL :STRING
REPEAT :REPEATER [TYPE SE :STRING ]
END
```

Figure III

year, the manipulation of lists can make my head spin.

There are many easier things to do with lists, and I did this exercise only because it came to me arbitrarily, not as a problem designed for a Logo text book.

Of course, Logo does have loops and Boolean operators and all the rest, but what I hope you will see is its power as a tool for manipulating language, human language.

Mathematics is itself a language, a very formal and compact language, well understood by its practitioners, so there are good reasons why it was the first language to be made machine readable.

Lisp represented an attempt to make programming more of an art and less of a science.

Obviously, you would not normally have to write standard procedures from scratch. You would already have REVERSEL, REVERSE and REDUCE in your toolkit; and it took me only half an hour because I had written REVERSE before.

SETUP, too, is a fairly standard item. My time was taken up working out the different forms in which a string might be typed in and still satisfy the conditions.

In fact, if any of you try it out, as I hope you will, you may find my bug. If you enter 0032 APPLES, the program should respond
SELPPA 2 SELPPA 2 SELPPA 2.

Infuriatingly, it prints out

```

TO REVERSEL :STRING
IF EMPTY? :STRING [OP :STRING]
OP LPUT REVERSE FIRST :STRING
REVERSEL BUTFIRST :STRING
END

TO REVERSE :W
IF EMPTY? :W [OP :W]
IF WORDP :W [OP WORD REVERSE BUTFIRST :W FIRST :W]
[OP LPUT FIRST :W REVERSE BUTFIRST :W]

TO REDUCE :R
IF :R < 1000 [OUTPUT INT :X]
MAKE "X :X / 10
OUTPUT REDUCE :X
END
    
```

Figure IV

```

TO REPROCESS :STRING
MAKE "REPEATER WORD FIRST FIRST :STRING
(WORD FIRST BUTFIRST FIRST :STRING
FIRST BUTFIRST BUTFIRST FIRST :STRING)
TEST NUMBERP :REPEATER
IFFALSE [PRINT [THE FIRST 3 CHARACTERS
MUST BE DIGITS] PR []
PRINT [TRY AGAIN] SETUP
MAKE "STRING FPUT BUTFIRST BUTFIRST BUTFIRST
FIRST :STRING BUTFIRST :STRING
MAKE "STRING REVERSEL :STRING
REPEAT :REPEATER [TYPE SE :STRING " ]
END
    
```

Figure V

SELPPA 32 times, as it does not recognise the first two zeros, and treats the number exactly as if I had typed in 32 APPLES.

The problem, does not occur

if you type in 0032APPLES. I leave you with two problems. The first is to get rid of the bug; and the second is to write an equivalent program in Basic.

● In my next article I shall look at conversational programs, in which we teach the computer to hold a simple conversation with the user.

AppleTip

A Sideways scrolling from right to left can be achieved using the hex dump of a routine shown here.

Each **CALL** to 768 will scroll the entire text screen one byte to the left.

The routine has within it two look-up tables to calculate the screen line address (Lo-byte and Hi-byte).

By using the following Basic commands in a program you can position any character anywhere on the screen, sometimes quicker than VTAB,HTAB.

```

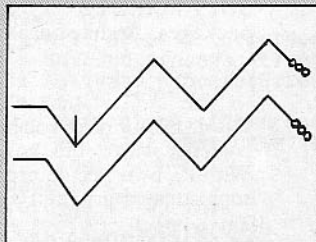
10 LD = 830:HI = 854
20 AD = PEEK (LD + Y) + 256
+ PEEK (HI + Y)
30 POKE AD + X,CHR
    
```

Where AD = the leftmost byte of your line.

X and Y = your co-ordinates in range 0-39 & 0-23.

CUR = your specified character.

A scramble type game can easily be programmed using this routine by drawing two dots on the right hand side of the screen and calling 768, then POKEing two more dots.



Sample screen

One scroll tip:
 POKE your playfield information into AD+38 - then when the screen is scrolled, all the blanks in colour 40 are shifted into 39 giving you a clean column to POKE your next information into.

Greg Hartland

0300-	4C	05	03	18	00	EA	A9	00
0308-	8D	03	03	8D	04	03	AE	03
0310-	03	BD	56	03	85	FD	B5	FB
0318-	BD	3E	03	85	FC	AA	EB	8A
0320-	B5	FA	AC	04	03	B1	FA	91
0328-	FC	C8	C0	27	90	F7	A9	00
0330-	8D	04	03	EE	03	03	AD	03
0338-	03	C9	18	90	D1	60	00	80
0340-	00	80	00	80	00	80	28	A8
0348-	28	A8	28	A8	28	A8	50	D0
0350-	50	D0	50	D0	50	D0	04	04
0358-	05	05	06	06	07	07	04	04
0360-	05	05	06	06	07	07	04	04
0368-	05	05	06	06	07	07	00	00
SAVE USING: BSAVE SCROLL,A#30								
0,L#6F								



Two things at once? It's child's play...

LAST month we looked at Lisa's workshop environment, which included the system filer and mouse editor. In this concluding article MIKE GLOVER looks more closely at the operating system, Pascal and Quickdraw – the powerful graphics package for Lisa.

THE operating system on Lisa enables multiple processes to coexist, with the ability to share data and communicate with each other. It also handles exceptions (interrupts) and memory management.

Processes (programs) can use up to 128 memory segments and each segment can occupy 128k. The segments can be either code or data and the total amount of memory used by any one process can exceed Lisa's available RAM.

To help the operating system in swapping in data segments, calls are provided to give programs the ability to define which segments must be in memory at a given time.

Data segments can also be accessed from disc as direct-access files, and processes can interchange information by sharing them.

The operating system currently supports Configuration System Calls but may not do so in the future. However the currently defined card types may be of interest. These are:

```
card-types = (no_card,
              apple_card,
              n_port_card,
              net_card,
              laser_card);
```

To reduce the impact of an error the file system maintains duplicate copies of critical information stored on different places on the media.

It also maintains a working directory to avoid having to use prefix volume names. Information is stored in pages which need not be contiguous.

One of the features that sets Lisa above most other micros is its ability to work on more than one process at the same time (multi-tasking).

This means that rather than wait an age (in its terms) for the user to press a key, or the printer to print a character, it nips off and does something else until the scheduler sends it back.

Figure 1 gives an idea how



Like father like son: Mike Glover, his two sons and Lisa

```
program father;
uses {$u syscall} syscall;

var ErrorCode, i : Integer;
    proc_id: Longint;
    progname: Pathname;
    null: Namestring;
    Info_rec: ProcInfoRec;
    Answer: Char;

begin
  progname := 'son.obj';
  Null := '';
  make_process(ErrorCode, proc_id, progname, null, 0);
  if (ErrorCode <> 0) then
    writeln ('Error ', ErrorCode, ' during process management');
    for i := 1 to 15 do { not a lot }
    begin
      writeln(' father is executing...');
      yield_cpu ( ErrorCode, false); { hand over control to son }
    end;

    Write ('K(ill S(uspend A(ctivate I(nfo');
    readln (answer);
    case answer of
      'K', 'k': Kill_process (ErrorCode, proc_id);
      'S', 's': Suspend_process (ErrorCode, proc_id, true);
      'A', 'a': Activate_process(ErrorCode, proc_id, true);
      'I', 'i': begin
                  info_process(ErrorCode, proc_id, Info_rec);
                  writeln('sons name is ', Info_rec.ProgPathName);
                end;
    end {case};
    if (ErrorCode <> 0) then
      writeln ('Error ', ErrorCode, ' during process management');
    end.

program son;
uses {$u syscall} syscall;
var ErrorCode : integer;
    null: namestring;

begin
  while true do
  begin
    writeln (' Son is running....'); { execute briefly }
    yield_cpu(ErrorCode, False); { hand back to daddy }
  end;
end.

end.
```

Figure 1

this works. Note the command YIELD_CPU.

Pascal is Apple's preferred development language and their preference shows in the amount of facilities built into this language compared to Basic and Cobol.

It is similar to the Pascal on the Apple II and III and is based on the ISO standard.

Among the differences between Lisa Pascal and Apple II Pascal are the absence of Turtle graphics — Lisa has Quickdraw which I will describe later.

Lisa has even stronger type checking and a '@' operator which can compute a pointer to a variable and even yield the entry point to a procedure or function.

Programs are compiled into native code and run very fast. I tried the Eratosthenes' Sieve bench mark which takes 516 seconds on my Apple IIe running in Pascal. On Lisa it took just eight seconds — 64 times faster! (See Figure II.)

For the record, Applesoft needs nearly 47 minutes to complete the program.

Lisa compiles code into 68000 object code and requires three stages to do it:

- Compile intermediate code (I-code)

- Generate object code (.obj)
- Link I/O and other object files into executable programs.

At first glance this seems a bit complicated, but in fact using an EXEC file it is quite easy. Here is the File I used to compile Eratosthenes' Sieve:

```
$exec
pmgg/primes
mgg/bugs { Just in case!! }
{default 1-code file}
gmgg/primes
{accept default output}
lmgg/primes
iospaslib
{no more}
-console
mgg/primes
$endexec
```

Figure III shows the dialog it had with Lisa. All I had to do was watch!

The listing is sent to a file called MGG/Bugs which I was able to have on the screen at the same time as the program source listing. This made debugging the source very easy. You will also see another program source on the screen display called KL.8QUEENS.

This program had some routines I wanted, and it was simple to transfer them from one source to the other using the mouse to copy and then paste.

Various aspects of the mouse can be accessed and controlled

using the hardware interface software provided with Quickdraw. Among these are location, update frequency, scaling and even a Mouse Odometer to tell how far the mouse moves in its lifetime!

The following extract from the manual shows the importance the Lisa team attach to mouse activity:

```
"...Function MouseOdometer:
ManyPixels;
```

In order to properly specify, design and test mice, it is important to estimate how far a mouse moves during its lifetime. Mouse Odometer returns the sum of the X and Y moves since boot time.

This is a typical mouse procedure:

```
Procedure MouseLocation(var x:
Pixels;var y:Pixels){x range 0..719,
y range 0..363}
```

Quickdraw is an amazing graphics package and is the means by which the windows and icons were made possible in the Lisa office system.

Despite the striking graphics already in use on Lisa, we have only just seen a fraction of the capabilities of this remarkable piece of software.

In addition to the two

dimensional monochrome graphics currently implemented on Lisa, Quickdraw actually supports three dimensional colour graphics!

Quickdraw can be accessed from both Pascal and machine code. It enables programmers to organise the screen into individual areas and draw in each area:

- Text in a number of different proportional fonts.
- Lines of any length and width.
- Regular shapes, circles, squares, polygons, etc.
- Arbitrary shapes.
- Pictures that can be a combination of the above accessed with a single call.

Quickdraw handles clipping, viewports and offscreen drawing and is very fast. Coordinates range from -32768 to +32767 in both axes. It can support devices that have up to 32 bits of colour information per pixel.

It would not be possible to list all the Quickdraw commands in this article. There are 10 pages in the summary alone. But Figure IV, taken from one of the example programs, may give some indication to the power and versatility of this package.

The Graf3D three dimensional routines allow the use of real (Quickdraw uses integers for speed of execution) and supports three dimensional clipping. The range of the reals is 1.4×10^{-45} to 3.4×10^{38} .

The view point can be set up independently from the object, and focal length can be set to telephoto, wide angle or normal. Objects can be rotated (yaw, pitch, roll) and translated along any plane. Scaling up or down is also supported. (See Figure V.)

Conclusion: Lisa offers the professional programmer a rich environment to develop powerful software of quality not before possible on a micro.

I look forward to seeing some of the developments that will surely emerge in the near future when the full potential of the Lisa starts to be realised.

With Lisa Pascal you get: Pascal software, Compiler, Assembler, Linker, Mouse Editor, Debugger, Utilities, Workshop Manager.

Pascal Manuals: Reference, MC68000, Workshop Manager, System software.

```
program prime;
uses {$u syscall} syscall;

const Side = 8190;
var Flags : array [ 0 .. Side ] of boolean;
    i,primes,k,count,iter:integer;
    err_flag,time_taken:integer;
    time_started,time_finished :Time_rec;

begin
  writeln(' Ten Iterations');
  get_time(err_flag,time_started);
  for iter := 1 to 10 do
  begin
    count:=0;
    for i:= 0 to side do
      flags[i]:= true;
    for i:= 0 to side do
      if flags[i] then
      begin
        primes := i+1+3;
        {writeln (primes);}
        k:= i + primes;
        while k<= side do
          begin
            flags [k] := false;
            k:=k+primes
          end;
        count := count +1;
      end;
    end;
  get_time(err_flag,time_finished);
  writeln(count, ' primes');

  Time_taken := 60*(time_finished.minute - time_started.minu
    +time_finished.second - time_started.secon
  writeln('Time taken',Time_taken,' seconds')

end.
```

Figure II


```
{V1.0} WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobo  
Quit, ?R
```

Run what program? <HGG/COMPILE

```
{V1.0} WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Bas, Quit, ?
```

Lisa Pascal Compiler V1.0
(c)1981 SVS, Inc. (c)1983 Apple Computer, Inc.

6-July-83

```
Input file - [.TEXT] mgg/primes  
List file - [.TEXT] mgg/bugs  
I-code file - [mgg/primes] [.I]
```

Elapsed time: 40 and 85/1000 seconds.

[F] 653 lines. 0 errors.

```
{V1.0} WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol  
Quit, ?g
```

Lisa Pascal Code Generator V1.0
(c)1981 SVS, Inc. (c)1983 Apple Computer, Inc.

20-Jul-83

```
Input file - [.I] mgg/primes  
Output file - [mgg/primes] [.OBJ]  
PRIME - PRIME Code size = 440  
Total code size = 440  
Elapsed time: 8 and 157/1000 seconds.
```

```
{V1.0} WORKSHOP: FILE-MGR, SYSTEM-MGR, Edit, Run, Pascal, Basic, Cobol  
Quit, ?l
```

Linker - PASCAL Program/Intrinsic Library Linker
Copyright 1983, Apple Computer, Inc.

24-Mar-83

```
Beginning memory - 277848  
After static allocation, memory - 121663  
Input file [.OBJ] - mgg/primes  
Input file [.OBJ] - iospaslib  
Input file [.OBJ] -  
Listing file [-CONSOLE] - [.TEXT] -console  
Output file [.OBJ] - mgg/primes  
Reading file: mgg/primes.OBJ  
Reading file: iospaslib.OBJ  
Read 2 files, max = 100  
4 segments, max = 128  
15 modules, max = 1450  
14 entries, max = 2000  
8 ref. lists, max = 8000  
15 references, max = 16000
```

```
Linking Main Program.  
Active: 3 of 15 read.  
Visible: 1 of 14 read.  
Global data: $002022  
Common data: $000000  
Linking segment #: 0 : file (JT) seg: 1 size: 440  
Beginning memory - 118567  
Ending memory - 118462  
0 Errors detected.
```

The output is executable.
Elapsed time: 23 and 545/1000 seconds.
That's all Folks !!! . . .

```
{ ----- Three D example ----- }  
PROCEDURE DrawBox(pt1,pt2: Point3D);  
{ draws a 3D box with shaded faces. }  
{ only shades correctly in one direction }  
VAR tempRgn: RgnHandle;  
BEGIN  
tempRgn:=NewRgn;  
OpenRgn;  
MoveTo3D(pt1.x,pt1.y,pt1.z); { front face, y=y1 }  
LineTo3D(pt1.x,pt1.y,pt2.z);  
LineTo3D(pt2.x,pt1.y,pt2.z);  
LineTo3D(pt2.x,pt1.y,pt1.z);  
LineTo3D(pt1.x,pt1.y,pt1.z);  
CloseRgn(tempRgn);  
FillRgn(tempRgn,white);
```

```
{ ----- draw line samples ----- }
```

```
MoveTo(330,34);  
Drawstring('Lines');
```

```
MoveTo(280,25);  
Line(160,40);
```

```
PenSize(3,2);  
MoveTo(280,35);  
Line(160,40);
```

```
PenSize(6,4);  
MoveTo(280,40);  
Line(160,40);
```

```
PenSize(12,8);  
PenPat(gray);  
MoveTo(280,61);  
Line(160,40);  
PenSize(15,10);  
PenPat(myPattern);  
MoveTo(280,80);  
Line(160,40);  
PenNormal;
```

```
{ ----- draw Wedge samples ----- }
```

```
MoveTo(570,148); Drawstring('Wedges');
```

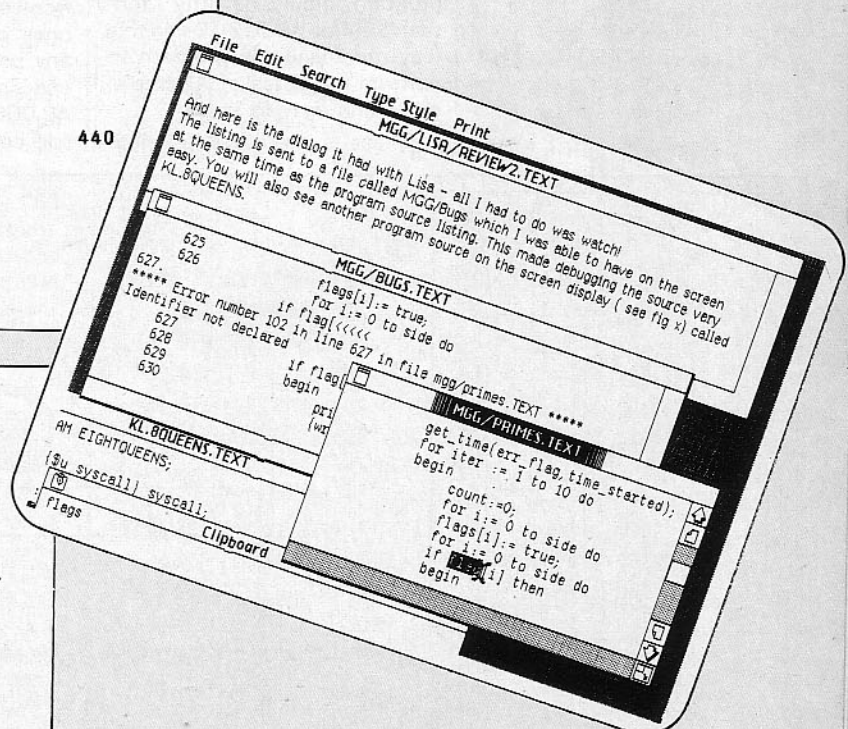
```
SetRect(tempRect,520,153,655,243);  
FillArc(tempRect,135,65,dkGray);  
FillArc(tempRect,200,130,myPattern);  
FillArc(tempRect,330,75,gray);  
FrameArc(tempRect,135,270);  
OffsetRect(tempRect,20,0);  
PaintArc(tempRect,45,90);
```

Left: Figure III

Above: Figure IV

Below left: Figure V

Below right: Lisa's screen



THIS is the first in a series of occasional pieces devoted to building up a comprehensive package for displaying hi-res graphics on the Apple.

There are several commercial packages available that perform a variety of graphical tasks, and if one of these meets your requirements then don't bother to write your own - it's just too much work!

However, if you need to do something novel, or the graphics are to be part of a larger program, then there is little choice but to re-invent the (graphical) wheel.

What I am hoping to achieve through this column is to speed up the process by preventing you going through triangles and squares before reaching the circle.

The aim is to produce a set of integrated graphical routines that form a flexible and easily expanded package. The complete package can then be added to any program to produce hi-resolution displays with minimum effort.

This first article considers how best to go about designing such a system and also presents the first few routines. Further routines will be given in future issues of *The Apple User*, and they will build up into a complete library.

Build up your own graphics package

PETER GORRY begins a new series designed to help you produce hi-res displays with minimum effort

ONE of the most important factors to consider when designing a graphics package is which programming language it should be in.

A choice of a good language can greatly simplify life, so the temptation to use Pascal or Logo for this project was strong. Unfortunately not enough people have these languages, so I decided to adopt good old trusty Basic instead. I know it's slow and hopeless with subroutines but at least it's there with every Apple II.

I also decided to use as few machine code routines or special tricks as possible so that the package can be easily understood by all.

The key to a good graphics system is to have as many tasks as possible relegated to subroutines. The main driving program should be little more than a series of GOSUBs. In this way modifying the program to perform new tasks is simple, quick and easy to write.

These subroutines fall into

two main types - those that do things, like draw and label, and those that ask for information, such as size and colour.

It is essential to keep these separate, otherwise you will find yourself constantly editing questions out of the routines when you don't need them.

A major nuisance when using subroutines in Basic is having to keep track of variable names to ensure that they aren't used elsewhere for a different purpose. This is particularly important if the graphics routines are to be added to other programs at a later date.

It is essential to adopt a consistent approach right from the beginning.

With these points in mind I shall make all variables - with just a few exceptions - begin with the letter Z. The "doing" subroutines will be numbered from 40,000 and the "asking" ones from 50,000. As long as any program you write doesn't use Zs or line numbers above 40,000 the graphics routines can be added at any time.

One final restriction I decided upon was to limit the routines to two-dimensional graphics, X and Y.

I shall, however, deal with 3-D techniques at another time.

The most immediate problem in any graphics application is that the subject to be drawn has "natural" or "user" units - such as feet and inches, tons of fish, or percentages - while the computer only has screen points to plot or erase.

Any program should allow the user to work in the natural units while behind the scenes it carries out all the necessary calculations to fit the picture onto the screen. This is shown schematically in Figure 1.

It can be seen from Figure 1 that we need to map the corners of our real world onto the corners of the screen image.

This process is achieved by writing two simple equations which can calculate the screen value (x,y) corresponding to any point on the real object (X,Y) providing we know the upper and lower values of the rectangles.

The equation for x is:

$$x = \frac{(X-XL)*(xu-xl)}{(XU-XL) + xl}$$

and similarly for y (remembering to substitute y for x in the equation).

If the algebra doesn't mean a thing to you, don't worry. The subroutine provided does all the work needed.

The accompanying listings consist of the first routines for our library, and a small example program that uses them. You can change the values in the example program to provide a variety of different versions of the same picture.

Each routine has comments at the front to explain any

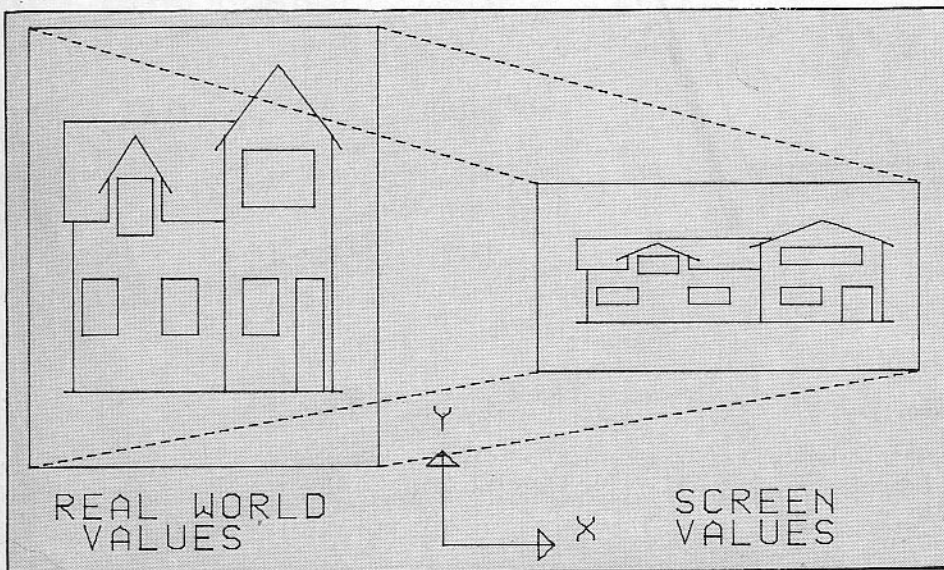


Figure 1

parameters required.

SET UP ROUTINE: This is used to set up one of the hi-res screens and to set default values for SCALE and ROT.

ZP, ZF and ZC set the page, graphics/text mode and plotting colour. So ZF=1, ZP=1 and ZC=3 will set full graphics on Page 1 with the colour set to white.

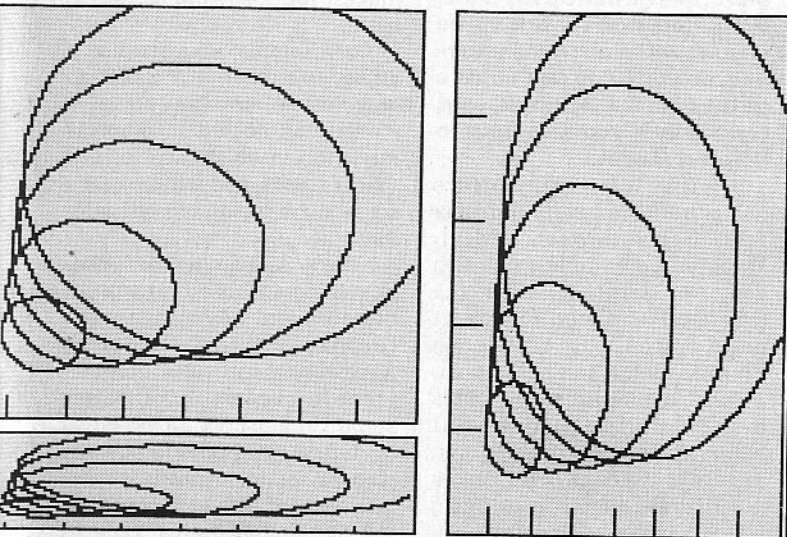
MAPPING ROUTINE: This is the routine responsible for setting up the equations for transforming between the real world and the screen - the heart

drawing graph axes.

The number of divisions for each axis is stored in another array ZG(). This array will eventually hold all sorts of information about the screen image. For the moment only ZG(1) and ZG(2) are used.

CIRCLE ROUTINE: This draws circles and clips them off at the edges of the border. The centre of the circle is stored in ZX, ZY with radius ZR.

These are all in user values, which are converted to screen points in line 40650. Also seen



Three plots from the example program

of the system.

The mapping information is stored in an array ZM(10) and, since this is the default length, it doesn't need dimensioning.

The order of information in the array is very important. Suppose one wants to plot a graph of percentage employment for 1973-1983 and we want it to occupy the whole screen we would have:

ZM(1) 1973 lowest user X
 2 1983 highest user X
 3 0 lowest user Y
 4 100 highest user Y
 5 0 lowest screen x
 6 279 highest screen x
 7 192 lowest screen y
 8 0 highest screen y

Take care with 7 and 8. The apparent contradiction stems from the fact that the Apple screen points are numbered from 0 starting at the TOP of the screen!

BORDER ROUTINE: This draws a border round the screen size chosen and puts tick marks along two sides if required. It will be especially useful for

in that line are the only variables not starting with Z.

I shall always denote screen-points by XP, YP. The routine is somewhat slow but clarity rather than speed was the overriding feature.

Finally a comment about plotting in colour. These routines are intended to make maximum use of the Apple resolution - which means that they use black and white only.

Should you use another colour the number of horizontal screen points for that colour would be halved. So if you plot everything in blue, say, half the vertical lines will be missing.

This can be overcome by plotting the picture twice but with the x screen points shifted across by one the second time. This is easy to accomplish since all you have to do is add 1 to ZM(5) and ZM(6) before the second plot.

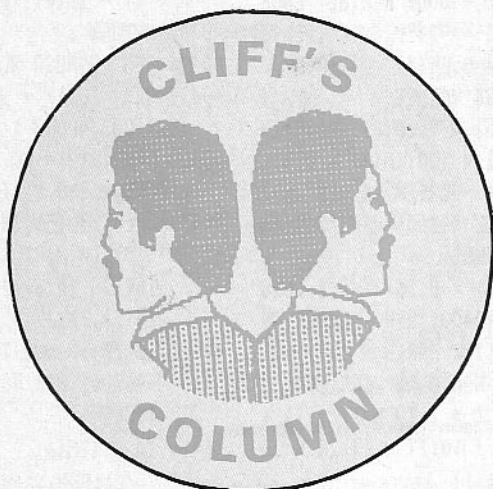
● Next month Peter Gorry will show how to move long basic programs past the hi-res screen and present routines to plot histograms.

```

100 REM EXAMPLE PROGRAM
110 ZC = 3:ZP = 1:ZF = 1: GO
SUB 40000: REM SET PAGE
120 ZM(1) = 0:ZM(2) = 1:ZM(3)
) = 0:ZM(4) = 1: REM
USER VALUES
130 ZM(5) = 0:ZM(6) = 140:ZM
(7) = 150:ZM(8) = 0:
REM SCREEN VALUES
140 GOSUB 40200: REM SET MA
PPINGS
150 ZB(1) = 8:ZB(2) = 5:GOSU
B 40400: REM BORDER AN
D TICK MARKS
160 REM NOW DRAW CIRCLES
170 FOR I = 1 TO 5:ZX = (I +
1) / 10:ZY = ZX:ZR = .0
9 * I
180 GOSUB 40600: REM CIRCLES
190 NEXT
200 END :

40000 REM SET UP ROUTINE
40010 REM ZP = 1,2 SELECTS
HI-RES PAGE
40020 REM ZC = 0-7 SELECTS
HI-RES COLOUR
40030 REM ZF = 0,1 SELECTS
TEXT + GRAPHICS OR FULL
GRAPHICS
40040 IF ZP < 1 OR ZP > 2 TH
EN ZP = 1: REM DEFAULT
PAGE 1
40050 IF ZC < 0 OR ZC > 7 TH
EN ZC = 3: REM WHITE
40060 IF ZP = 1 THEN HGR : 6
OTD 40080
40070 HGR2
40080 IF ZF = 1 THEN POKE -
16302,0: GOTO 40100:
REM FULL PAGE
40090 POKE - 16301,0: REM
TEXT + GRAPHICS
40100 HCOLOR= ZC: SCALE= 1:
ROT= 0
40110 RETURN :
40200 REM MAPPING ROUTINES
40210 REM RM(1)-RM(8) HOLD T
HE MAPPING VALUES
40220 REM IN THE ORDER: XL,
XH,YL,YH
40230 REM USER VALUES FIRST,
THEN SCREEN VALUES
40240 ZM(9) = (ZM(6) - ZM(5)
) / (ZM(2) - ZM(1))
40250 ZM(10) = (ZM(8) - ZM(7)
) / (ZM(4) - ZM(3))
40260 DEF FN XCN(X) = (X - Z
M(1)) * ZM(9) + ZM(5)
40270 DEF FN UXCN(X) = ZM(1)
+ (X - ZM(5)) / ZM(9)
40280 DEF FN YCN(Y) = ZM(7)
+ ((Y - ZM(3)) * ZM(10))
40290 DEF FN UYCN(Y) = ZM(3)
+ (Y - ZM(7)) / ZM(10)
40300 RETURN :
40400 REM BORDER ROUTINE
40410 REM ZB(1) = NO OF DIV
ISIONS ALONG X AXIS
40420 REM ZB(2) = NO OF DIVIS
IONS ALONG Y AXIS
40430 HPLLOT ZM(5),ZM(7) TO Z
M(5),ZM(8) TO ZM(6),ZM
(8) TO ZM(6),ZM(7) TO
ZM(5),ZM(7)
40440 IF ZB(1) < 2 THEN GOTO
40500: REM NO X AXIS
TICKS
40450 ZA = (ZM(6) - ZM(5)) /
ZB(1):REM X INCREMENTS
40460 ZB = (ZM(8) - ZM(7)) /
20: REM TICKS 1/20 OF
HEIGHT
40470 FOR ZI = 1 TO ZB(1) -1
40480 HPLLOT ZM(5) + ZA * ZI,
ZM(7) TO ZM(5) + ZA *
ZI,ZM(7) + ZB
40490 NEXT : REM X AXIS DONE
40500 IF ZB(2) < 2 THEN GOTO
40560
40510 ZB = (ZM(8) - ZM(7)) /
20: REM KEEP TICK MARK
S THE SAME LENGTH
40520 ZY = (ZM(8) - ZM(7)) /
ZB(2):REM Y INCREMENTS
40530 FOR ZI = 1 TO ZB(2) -1
40540 HPLLOT ZM(5),ZM(7) + ZA
* ZI TO ZM(5) - ZB,ZM(
7) + ZA * ZI
40550 NEXT : REM Y AXIS DONE
40560 RETURN :
40600 REM CIRCLE ROUTINE
40610 REM CENTRE AT ZX,ZY AN
D RADIUS ZR - USER VAL
UES
40620 ZD = 0:ZE = ZR:ZA = CO
S (.1):ZB = SIN (.1):Z
P = 0
40630 FOR ZI = 1 TO 64
40640 ZZ = ZD * ZA - ZE * ZB
:ZE = ZE * ZA + ZD * Z
B:ZD = ZZ
40650 XP = FN XCN(ZX + ZD):Y
P = FN YCN(ZY + ZE)
40660 IF XP < ZM(5) OR XP >
ZM(6) OR YP > ZM(7) OR
YP < ZM(8) THEN ZP = 0
: GOTO 40700
40670 IF ZP = 1 THEN 40690
40680 HPLLOT XP,YP:ZP = 1
40690 HPLLOT TO XP,YP
40700 NEXT:REM END OF CIRCLE
40710 RETURN :

```

Cliff McKnight muses on the state of play in the world of computer games

IT'S a funny thing is time, don't you think?

I mean, I don't think I'll ever get used to phoning America and saying "Good afternoon" when it's been pitch dark outside for hours.

Also, I'd like to know where time goes to, because I never seem to have enough of it these days.

Well, to be honest, I can usually find a few minutes for a game of something exciting, and adventure games sometimes swallow huge chunks of it.

Talking of time, leads me once again to Time Zone. Since we reviewed it in December 1982 Denise and I have happily answered a constant stream of requests for hints.

However, some of the problems have turned out to be disc problems. In such cases we have to refer people back to their supplier, and some suppliers have made a fairly heavy (circa £10) charge since the discs were bought some time ago.

Now what is the point of advertising "It takes a year to play" if you're only going to provide a back-up service for faulty discs for 30 days?

The problems have been fairly well into the game – in one case almost at the end – so it is unreasonable to expect the user to find it within the back-up period.

I think that as long as the disc hasn't been user-corrupted, back-up should be free (allowing for a small handling charge) for a period of time related to

the game's expected life. So there!

On the subject of service for American products, let me bore you with another of my experiences. Some time ago I reviewed some discs from The Learning Company.

My eldest daughter, who was four at the time, did most of the work on them and in the process discovered a "secret" in one of the games.

As suggested in the text, we took a photo of the screen and Janet sent it off.

A few months went by, during which we told Janet that letters take a long time from America, and eventually I wrote to the head of the company.

Neither Janet's nor my letter have been answered. I think this is particularly bad, since the company obviously encourages children to write to them.

It means a lot to a four-year-old to have a letter ignored, and it presumably indicates the kind of service you can expect from The Learning Company.

Oh well, time to get my copy in the post. As Thoreau once said, "Time is but the stream I go a-fishing in".

I haven't caught much recently.

GREMLINS get everywhere – not just in games programs, but in games reviews as well.

Last month's review of Buzzard Bait was written by Irish author Herbie Brennan and NOT by Cliff McKnight. Apologies to them both.

A question of dragons

Title: The Quest

Authors: Dallas Snell, Joe Toler and Joel Ellis Rea

Publisher: Penguin Software

Requirements: None stated

"LET it be known throughout the land – The Quest is about to begin".

That's what it said in the blurb, so I girded up my loins. Well, you don't want to be caught with your loins ungirded, do you? You can't beat a quick gird I always say, especially if you are about to go on a Quest.

In the latest adventure game from Penguin, you play King Galt's newest adviser. There you are, quietly minding your own business, when the king decides to send you to sort out a dragon who is pestering Princess Diana and her people.

Well, actually the king sends his champion, Gorn, but although Gorn is a good fighter, he was at the back of the queue when grey matter was being handed out . . . Gorn orf, you might say.

Your job is to tell Gorn what to do. But don't bother telling him to talk to anybody – he reserves all his best lines as rebuffs for your less obvious instructions.

If it sounds like another American Standard hi-res adventure game, that's not too far from the truth. However, Penguins do it differently, so there are a few extras.

The major difference is that there is more than one route to the solution of The Quest. When you make it to the end (which is not as straightforward as it

might initially seem), you will almost certainly have missed some bits.

This means you can continue to explore and find another route, unlike other adventure games which become "swaps" when you've done them.

If you want to get somewhere quickly, you can toggle into text mode and move without generating the graphics. You can also issue multiple commands which, combined with text mode, makes for really rapid movement.

For example, you can use a string like N,W,N,W,S,S,E to make seven moves at once.

You can do this in graphics mode too, in which case you can then sit back and watch as the screens are generated one after the other.

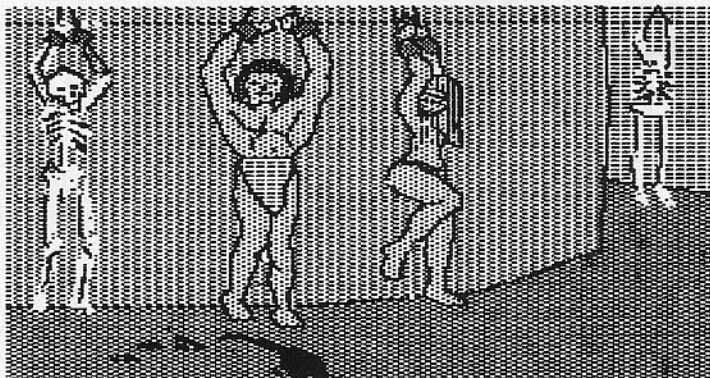
Of course, coming from Penguin the graphics are very good, having been created with The Graphics Magician. They seem to have paid particular attention to the female form, which raises some interesting possibilities for future adult-oriented games (excuse my chauvinism)!

However, if I had to nominate my all-time favourite piece of hi-res art it would be one of the still life studies in The Quest.

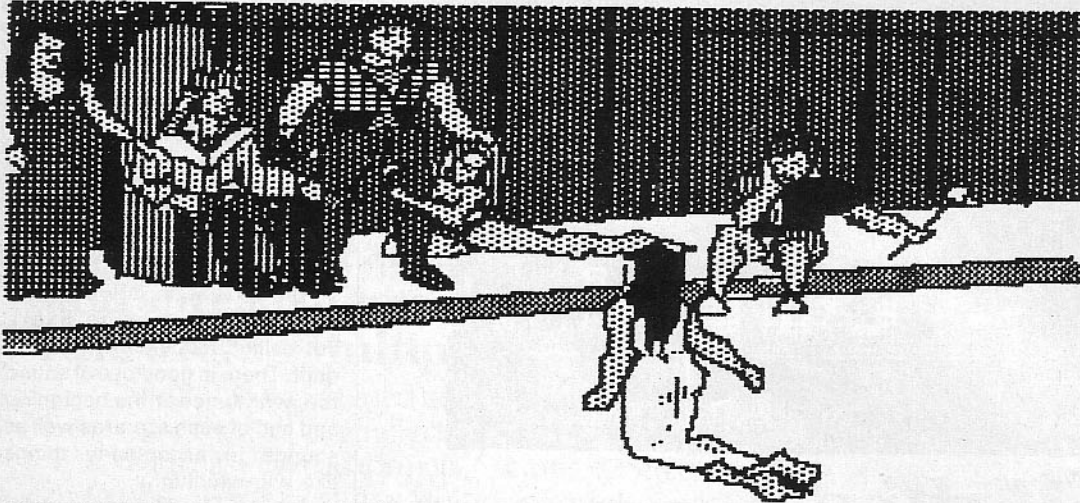
In monochrome, "trees reflected in a lake at dusk" doesn't look all that good, but in colour the effect is wonderful. It makes me wish I had a colour printer so I could take a permanent copy to hang on my wall.

For some reason, each key-press produces a click, a bit like working on a manual typewriter. I can't think why Penguin chose to include such a feature. It does not detract from the game, but it doesn't add anything either.

When you do complete The



Hanging around in The Quest



Lady Diana meets King Galt

Quest, you are rewarded with a party in your honour. Watch out for the juggler – he's magic! It makes a nice change from the usual "well done" message that greets you after hours of toil and tears.

In terms of difficulty, *The Quest* is about average. There are some tricky bits, but nothing the intrepid adventurer couldn't cope with. There are over 200 locations, filling two sides of a disc, but a large maze accounts for at least a quarter of that.

Even so, it's a big game and I enjoyed playing it. I've only found one and a half routes so far, so I suppose I'd better leave my loins girded for a bit longer.

Cliff McKnight

DIY Jackanory

Title: Story Machine
Authors: Clark Quinn, Margaret Weinstein and James Schuyler
Publisher: Spinnaker Software
Requirements: 48k Apple

THE first program we wrote for our youngest child simply put a new picture on the hi-res screen each time she pressed a key. At the age of eighteen months she was delighted with it.

However, it occurred to us as she began to recognise letters that we could capitalise on this by presenting a picture beginning with the particular letter pressed.

Our dream program (that is, the one we *didn't* write) was going to allow a noun to be typed and the machine would respond with an appropriate picture. It would even "learn" by allowing a picture to be built up if the word was not in its vocabulary.

At first sight, I thought *Story Machine* from Spinnaker had gone one better than our dream.

The idea is that by using a vocabulary of about 40 words, a child can type sentences, combine them into stories and see the stories animated. For example, typing "the boy walks to the tree" results in just that – a boy appears and walks to a tree.

Of the vocabulary, there are 13 nouns ranging from boy, girl and apple to bumpus. (Bet you don't know what a bumpus is!) The 11 verbs also contain a surprise: to zot.

As each sentence is entered, it is terminated with a "full stop" and this causes it to be acted out in the picture. Pressing ESC will cause the whole of the story so far to be acted out.

Additionally, stories can be saved to disc or recalled. Each specially initialised story disc can contain 15 stories, and there is no limit on the number of story discs you can have.

If it sounds like a fabulous idea, that's just what it is. Unfortunately, I think that machine limitations mean that the implementation of the idea doesn't do it justice.

For example, the vocabulary is quite limited, with only 24 nouns and verbs, the rest being articles, adjectives (like "this"

and "that"), pronouns, possessive pronouns and prepositions.

To be fair, the program claims to do more than simply animate noun-verb combinations. For example, incorrect grammar is not admissible and the program crosses out (and then removes) mistakes of this sort.

It is suggested in the manual that the program encourages a positive attitude towards writing, helps children to write longer stories and also teaches keyboard familiarity.

However, most of the youngsters I know write longer stories than *Story Machine* can accept. Also, only the present tense can be used and several of the children who tried the package found this a bit hard to accept. They are used to telling stories of the form "... and then ... and then ...".

The manual's suggestion to think of the story as a script for the computer didn't impress them much since they didn't really understand the concept of "script".

Only a certain number of actors are allowed in the story at any one time and their activities are limited by the surroundings.

It's not too unnatural to get a "can't do it from here" error message if the actor isn't next to the thing to be acted upon, but this message actually occurs if something is blocking the way. If nothing is in the way, the "must be closer" message appears.

As I said, I think this is a great idea, which has been limited by the machine. The two main

limitations are storage space (since only 48k is required) and speed.

In a sense they are not independent since the lack of storage space means it is necessary to access the disc regularly, and this leads to a loss of speed.

Imagine how the idea could be implemented with, say, 1 megabyte of internal RAM. It's not an idle dream because you can get that much RAM on a single card these days, and you can fit seven of them into the Apple!

The increased space would mean a bigger vocabulary without the necessity to access the disc repeatedly.

For all my criticisms, I have to applaud Spinnaker for *Story Machine*. With colour, sound and animation it is a novel (no pun intended) package that the children largely enjoyed.

It's obviously no use producing a package which requires a megabyte in a market where 48k is the norm, but I can't help thinking what it *could* be like.

Cliff McKnight

Defend your base

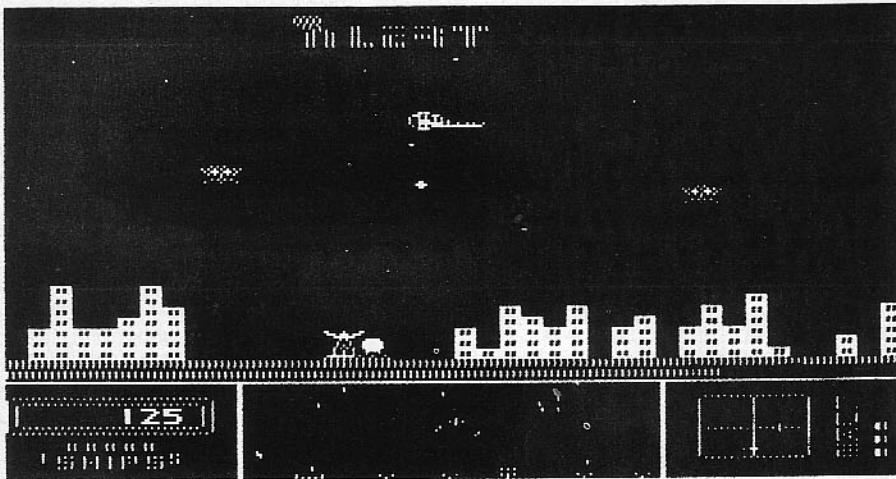
Title: Repton
Authors: Andrew Kaluzniacki and Dan Thompson
Publisher: Sirius Software
Requirements: 48k Apple

IF you're one of the elite who can make it past wave one in *Defender*, Sirius might just have the thing for you. It's called *Repton* and is probably the best *Defender* game I have played on the Apple.

In fact, not only is it a good version of the arcade original, it also has a mini version of *Scramble* thrown in for good measure. It's fast and furious, and has the meanest bunch of nasties Sirius has ever thrown at you.

Alongside the dreaded *Nova Cruisers*, the fearsome *Quarriors* and the frightful *Single-ships* are a whole host of other invaders, all dedicated to your destruction.

Piloting your *Armageddon* ship, you cruise over a scrolling



Armageddon here we come

landscape of buildings which represent your home planet Repton. Blocks are ripped off the buildings by the aliens and taken to build their own surface base. Your ultimate aim is to stop them completing this, but survival comes first.

The Armageddon ship "released from Sirius Base in view of this dire situation" has a limited supply of "nuke bombs" and an energy shield, as well as the usual laser gun.

Releasing a nuke bomb will send anything on the screen into oblivion (bar you, naturally!). Switching on the shield renders the ship untouchable, and uncontrollable!

Aside from attacking you and building the base, the invaders are keen on stealing energy from the planet. If allowed to, they will use the energy to build their base - not good news for you.

Should they complete the base, the game explodes into a frenzy of sound and animation as the ship's failsafe mechanism takes over.

Automatically an Armageddon bomb is detonated which destroys everything on the planet (apart from you-know-who), and the alien base falls below ground.

Now it's your mission to fly underground to the alien's main generator. On the way, rockets are fired at you and ships peel off from the roof to dive into an attack. Make it past these, and the generator appears. Just one carefully placed shot, and you've blown up their entire base. Then for wave two!

Repton has all the usual CTRL functions for restarting the game and so on, plus an

option to change the joystick axes.

To help you in your mission, there is a radar display at the bottom of the screen, as in Defender, and a speed indicator. There's also an iron-on T shirt transfer so all your friends can tell you've got the game.

If you like hot arcade action with good graphics and sound (and who doesn't?) you'll enjoy Repton.

Admittedly I'm a terrible Defender player, but I think Repton is a challenge worthy of any arcader.

Julian Brewer

Trapezes and traps

*Title: Sammy Lightfoot
Author: Warren Schwader
Publisher: Sierra On-Line
Requirements: 48k Apple*

HE flies through the air with the greatest of ease... oh dear, he's missed the trapeze! Sammy Lightfoot is the daring young man on the trampoline, trapeze and various other hazardous bits of equipment.

Unlike the rest of us who sustain hernias or broken limbs in contact with such devices, Sammy's miscalculations lead to no more than his wig being whirled.

Each scene has Sammy starting at the bottom of the screen and having to get to a particular place at the top. On two of the three scenes he must challenge a pumpkin wearing sunglasses.

To get to the pumpkin in scene 1, he must bounce on trampolines, jump gaps and swing on trapezes while avoiding balls which are falling down and rolling about.

Scene 2 involves jumping across a set of stepping stones which have a nasty habit of disappearing, while avoiding blocks coming from above. Eventually, he makes it to a flying carpet which will transport him to the end of the scene if he can stay on it.

In scene 3 he really has his work cut out. He must avoid a ball bouncing around a box, dodge under a series of deadly moving poles, grab one trapeze and use it to reach another trapeze (which is moving out of phase with the first) and finally jump to challenge the pumpkin.

Success on level 0 leads to level 1. The basic scenes stay the same, but the hazards get progressively harder to dodge as the levels increase.

There are 12 levels in all and so far I haven't got farther than scene 2 on level 1. The starting level can be selected, though, so I've had a go at higher levels.

Points are only scored by getting through the scene and they're dropping all the time so Sammy can't afford to hang about. If he doesn't make it by the time the counter hits zero, it's not fatal - just bad for the score.

There's a hall of fame for the top 10 scores, and Sammy has four lives each game.

The animation is well done, with Sammy looking like a pot-bellied teddyboy with a big quiff. There is good use of sound too, with tunes at the beginning and end of each scene as well as sounds to accompany things like wig-whirling.

All the usual convenience controls are there, too, as you'd expect from Sierra On-Line.

My first thought on seeing Sammy Lightfoot was that it was an adaptation (or rip-off, as we say in the trade) of Apple Cider Spider.

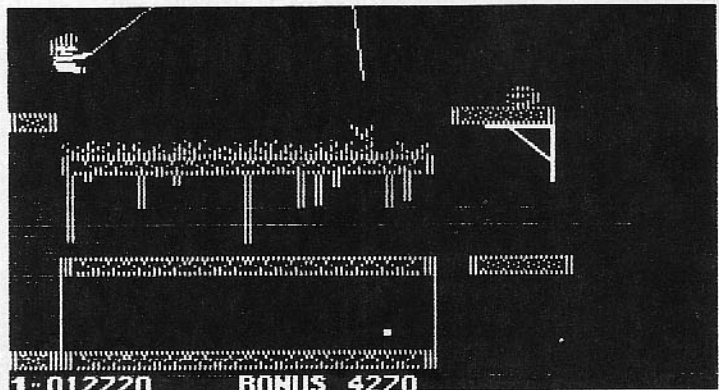
However, I then remembered that Spider is also from Sierra. That might explain the similarities - three scenes in each level, moving from bottom to top, avoiding hazards, jumping, climbing and so forth.

You might think there's a world of difference between a spider and a trapeze artiste, but when it comes to manipulating both sets of pixels around the screen, they feel alike.

They're good games, but I don't think many people would want them both.

Of the two, Sammy Lightfoot gets more difficult at lower levels. Also, success at level 0 leads to level 1, whereas starting Spider on level 0 keeps it there. I can see arguments for both systems so it boils down to personal preference, with mine being for Sammy Lightfoot.

Cliff McKnight



Having a swinging time with Sammy Lightfoot

JUDGING from our postbag, the most exciting Apple events from a programmer's point of view are dated, lower-case, beautifully listed programs.

The Windfall lower case adapter offer generated particular interest, and many, many suggestions were received on ways to incorporate lower case into Basic programs.

We have also been sent several programs which prepare well formatted Basic listings and many routines which handle dates in different ways.

Some of these are very clever routines and I plan to give them a public airing over the next few months.

Basil Shall has written to say that manipulating dates within programs can become quite a headache for the programmer, and we present below his subroutine for doing just that.

Before we move on to Basil's work, however, I would like to mention two algorithms, previously published in many

COMPILED THOUGHTS by MAX PARROTT

books, which are useful for date manipulations.

The first is a method of calculating the weekday on which the first day of January falls for any year between 1582 and 4902 inclusively.

Let the year be the integer Y , then we can calculate the integer DAY (where Sunday is 0, Monday is 1, up to Saturday which is 6) as follows:

```
10 A = INT((Y-1)/100)
20 B = Y - 1 - 100*A
30 DAY = 799 + B + INT(B/4) +
  INT(A/4) - 2*A
40 DAY = DAY/7 - INT
  (DAY / 7)
```

The second algorithm calculates the number of days between any two dates.

If the first date is represented

by the variables $D1$, $M1$, $Y1$ and the second by $D2$, $M2$ and $Y2$ (the years $Y1$ and $Y2$ should be in their full form, that is 1984 not 84) then perform the following to calculate the difference:

```
10 A = INT(365.25*
  (Y1-(M1<3))) + INT
  (30.6*(M1+1+
  (M1<3)*12))+D1
20 B = INT(365.25*
  (Y2-(M2<3))) + INT
  (30.6*(M2+1+
  (M2<3)*12))+D2
30 D = ABS(B-A): REM The
  difference
```

These routines, combined with others which will be given in *Apple User* during the next few months, will prove useful to many. Basil Shall has a pro-

gramming method which he describes below.

Next month, I'll present some of the lower-case routines we have received, but as a titbit it's worth noting that lower case strings and REM statements can be introduced into your programs using a word processor like Applewriter II which writes out standard DOS TEXT files.

Simply type the program into the word processor, save the file and EXEC it back into memory.

Of course you will need the lower case adapter to display your program correctly on the screen.

If you try this tip without an adapter, using for example, Applewriter's technique of highlighting upper case characters, in the hope of having your program at least function correctly on a printer, you won't be able to EXEC the file back into memory.

● *There is an order form for the lower case generator on Page 57.*

DEALING WITH DATES

RECENTLY I was required to design and implement a credit control system. Dates were vital to the system as strict penalty clauses were enforced for each day that payment was delayed.

The system specification was quite simple. If we invoice Mr X on January 28, 1984, and he has 10 days to pay, we need to know when to expect payment. And if payment does not arrive on time, a letter must be printed every Friday thereafter, until payment is made, reminding him of when he should have paid, and the cost of the penalty clause to date.

As most programmers are aware, adding 10 days to January 28, 1984, can prove to be a complicated process. Simple addition is not the problem, but rather converting the date into a number (day-number) on which the computer can perform arithmetic operations. But this is only half the problem.

Even more intricate is the transformation of the day-number into a date for reporting

purposes. After all, which manager wants to know that money is due in on day 724318 rather than January 28, 1984?

To overcome this problem, I developed the routine shown on Page 48.

The program is divided into two main processing areas:

- The driver routine in lines 5-140.
- The main subroutine in lines 3000-3340.

The driver routine can be adapted by the user to suit his specific requirement. It must, however, establish the variables required for the subroutine, before the subroutine is called. The subroutine must be incorporated within the main body of the program.

The variables that must be passed to the subroutine are:

H\$: This variable indicates whether the routine is to

convert a date or a daynumber.

Then either **DT\$**, which is a date in the form DDMMYY (eg, 280184 for January 28, 1984) or **NUM**, which is the day-number of the date to be determined.

I decided to have the date entered in this format because date entry would be done from a numeric keypad, and the operator would find it much easier to key in numerics only rather than alphas and numerics.

Note that the variable $MV()$ must be dimensioned in the driver routine.

The subroutine calculates either the date, or the day number and returns with the date in variables **DY** (day), **M** (month), and **YR** (year). The daynumber is returned in the variable **DN**. In both cases, the day of the week is returned as

DOW, with 0 being Friday.

Although there is no validation section in the driver routine of this example program, I have included a very simple validation test in the subroutine (lines 3280-3290).

This checks to see that the month was entered as a numeric. If not, the variable **M\$** is set to "ZZZ" and the subroutine returns to the driver.

In line 100 of the driver, I check that the value of **M\$** is not "ZZZ" to satisfy myself that the date was passed in a valid format.

There are two possible processing routes. One for converting daynumbers, and one for converting dates.

By first examining the DATE CONVERSION route, it can be seen that the heart of the calculation lies in lines 3300-3330. These three lines contain the magical mathematical formula for calculating a daynumber for any given date.

The daynumber is calculated after the relevant values of variables, **DY**, **M** and **YR** have

**BASIL SHALL utilises the
Apple's iterative power**

TECHNIQUE

been established. Control is then passed back to the main driver routine.

Unfortunately this formula is only a "one-way" formula. That is, it is not possible to calculate a date from it, given a particular daynumber.

If you now examine the DAYNUMBER CONVERSION route you will see that the date is calculated by passing an approximate date through a loop, until the daynumber being sought matches the daynumber produced from the calculation in lines 3300-3330. At this point, I

know that the date value would be given by the variables, **DY, M, YR.**

I felt that using the iterative powers of the Apple would be the easiest way for me to achieve my objectives, rather than trying to invent a clever formula to convert daynumbers to dates.

A more detailed examination of this route reveals the following:

□ Lines 3060-3120 set up an approximate date from which to start. Empirical evidence led me to deduct 465 from the original

daynumber in order to arrive at this initial estimate (see line 3060).

□ The calculations of this original estimate are not subject to any validation, so in order to ensure that the variables remain reasonable, I check that the month lies between 1 and 12. If not, it is adjusted (lines 3130-3140).

□ Similarly, days are checked for their validity. Days may not be greater than the maximum amount of days in that particular month (see line 3160).

□ Finally, lines 3170-3190

check for valid leap years when 29 days in February is allowed.

□ The estimated date is then passed to the date conversion subroutine in lines 3300-3330.

□ If the daynumber resulting from the estimated date is equal to the original daynumber, then the date we are looking for must be the estimated date!

□ If the resultant estimated daynumber is not equal to the original daynumber, then the estimated date is adjusted by one day in the correct direction and the calculation is performed again until a match is found.

```

1  REM *****
   # DATE CONVERSION #
   # ROUTINE #
   # BY: BASIL SHALL #
   # C/D FLEXDATA #
   # 63 LONGACRE #
2  REM # LONDON WC2E 9LL #
   # VARIABLES USED: #
   # DN,DT#,DOW#,DY,F1#
   # FL,H#,I,M,M#,MV()#
3  REM # ND,NUM,WV,X,YR,Z #
   # *****
5  DIM MV(12)
10 HOME
12 FOR I = 1 TO 39: PRINT "#";: NEXT
   I: PRINT
14 PRINT " FLEXDATA DATE CO
   NVERSION"
16 FOR I = 1 TO 39: PRINT "#";: NEXT
   I: POKE 34,3: PRINT: PRINT
   : PRINT: PRINT
20 INPUT "(D)ATE, DAY(N)NUMBER, D
   R (Q)UIT?";H#
30 IF H# = "D" THEN PRINT: INPUT
   "DATE IN FORM DDMYY";DT#: GOSUB
   3000: GOTO 100
40 IF H# = "N" THEN PRINT: INPUT
   "DAYNUMBER ";:NUM: GOSUB 300
   0: GOTO 100
50 GOTO 140
100 IF M# = "ZZZ" THEN PRINT "E
   RROR IN DATE ": GOTO 20
105 IF H# = "N" THEN PRINT: PRINT
   "THE DATE IS "; STR# (DY) +
   " " + STR# (M) + " " + STR#
   (YR): PRINT
110 IF H# = "D" THEN PRINT: PRINT
   "THE DAYNUMBER IS:DN: PRINT
112 PRINT "DAY OF WEEK IS ";DOW#
   : PRINT: REM FRIDAY=0
115 PRINT
120 GOTO 20
140 POKE 34,0: END
2999 REM *****
   # DATE/DAYNUMBER #
   # CONVERSION #
   # ROUTINE #
   # *****
3000 IF F1 = 1 THEN GOTO 3030
3010 MV(1) = 31: MV(2) = 29: MV(3) =
   31: MV(4) = 30: MV(5) = 31: MV(
   6) = 30: MV(7) = 31: MV(8) = 3
   1: MV(9) = 30: MV(10) = 31: MV(
   11) = 30: MV(12) = 31
3020 F1 = 1
3030 IF H# = "D" THEN GOSUB 325
   0: RETURN: REM DATE AS VAR
   IABLE DT# IN FORM DDMYY(280
   283)
3040 IF H# = "N" THEN GOSUB 306
   0: RETURN: REM CONVERTDAYN
   UMBER TO DATE
3050 REM *****
   # CONVERT DAYNUMBER#
   # TO DATE. #
   # DATE RETURNED IN #
   # VARIABLES DY;M;YR#
   # *****
3060 ND = NUM - 465
3070 WV = ND / 365
3080 YR = INT (WV)
3090 WV = WV - YR
3100 WV = WV # 365: WV = WV / 31
3110 M = INT (WV): WV = WV - M: WV
   = WV # 31
3120 DY = INT (WV)
3130 IF M > 12 THEN M = M - 12: Y
   R = YR + 1: GOTO 3130
3140 IF M < = 0 THEN M = M + 12
   : YR = YR - 1: GOTO 3140
3150 IF DY > MV(M) THEN DY = DY -
   MV(M): M = M + 1: GOTO 3130
3160 IF DY < = 0 THEN DY = DY +
   MV(M - 1): M = M - 1: GOTO 31
   60:
3170 IF M = 2 AND DY = 29 AND ((
   YR / 4) - INT (YR / 4)) = 0)
   THEN GOTO 3200
3180 IF FL = 0 AND M = 2 AND DY =
   29 THEN FL = 1
3190 IF FL = 1 AND DY = 29 THEN
   FL = 0: DY = DY + 1: GOTO 313
   0
3200 GOSUB 3300: REM WORK OUT D
   AYNUMBER FOR ESTIMATED DATE
3210 IF DN > NUM THEN DY = DY -
   1: GOTO 3130
3220 IF DN < NUM THEN DY = DY +
   1: GOTO 3130
3230 RETURN: REM DATE IS IN DY
   +M*YR
3240 REM *****
   # CONVERT DATE TO #
   # DAYNUMBER. #
   # DAYNUMBER RET'D #
   # IN VARIABLE DN #
   # *****
3250 DY = VAL ( LEFT# (DT#,2))
3260 YR = VAL ( RIGHT# (DT#,2)):
   YR = YR + 1900
3270 M = VAL ( MID# (DT#,3,2))
3280 IF M > 12 THEN M# = "ZZZ": RETURN
3290 IF M < 1 THEN M# = "ZZZ": RETURN
3300 IF M < = 2 THEN X = 0: Z =
   YR - 1
3310 IF M > 2 THEN X = INT ((0.
   4 # M) + 2.3): Z = YR
3320 DN = (365 # YR) + (31 # (M -
   1)) + DY + INT (Z / 4) - X
3330 DOW = DN / 7: DOW = DOW - INT
   (DOW): DOW# = (DOW # 7) + 0.5
   : REM FRIDAY=0
3340 RETURN: REM ANSWER AS DN

```



One hundred and eighty!!!!

THE very popular Darts game published in the March 1983 issue of *Windfall* contained the suggestion that voice output be added if you have the necessary hardware. Here, from **DAVE ECKERSALL**, are the necessary patches to be added to that game to give you everything but a pint of beer and the television cameras!



```

890 NUMB = T: GOSUB 2000: NEXT M1
1100 GOSUB 120: GOSUB 2200
2000 REM SORT OUT NUMBER
2002 CO = 0
2005 IF NUMB = 100 THEN CO = 2: C
      % (1) = 1: C% (2) = 18: GOTO 21
      60
2010 IF NUMB > 99 THEN CO = 3: C%
      (1) = 1: C% (2) = 18: C% (3) = 1
      9: NUMB = NUMB - 100
2020 IF NUMB = 10 THEN CO = CO +
      1: C% (CO) = 10: GOTO 2160
2030 IF NUMB = 11 THEN CO = CO +
      1: C% (CO) = 11: GOTO 2160
2040 IF NUMB = 12 THEN CO = CO +
      1: C% (CO) = 12: GOTO 2160
2050 IF NUMB = 13 THEN CO = CO +
      1: C% (CO) = 13: CO = CO + 1: C%
      (CO) = 14: GOTO 2160
2060 IF NUMB = 15 THEN CO = CO +
      1: C% (CO) = 15: CO = CO + 1: C%
      (CO) = 14: GOTO 2160
2070 IF NUMB > 13 AND NUMB < 20 THEN
      CO = CO + 1: C% (CO) = NUMB -
      10: CO = CO + 1: C% (CO) = 14: GOTO
      2160
2080 IF NUMB > 19 AND NUMB < 30 THEN
      CO = CO + 1: C% (CO) = 16: GOTO
  
```

```

2140
2090 IF NUMB > 29 AND NUMB < 40 THEN
      CO = CO + 1: C% (CO) = 13: CO =
      CO + 1: C% (CO) = 17: GOTO 214
      0
2100 IF NUMB > 39 AND NUMB < 50 THEN
      CO = CO + 1: C% (CO) = 4: CO =
      CO + 1: C% (CO) = 17: GOTO 214
      0
2110 IF NUMB > 49 AND NUMB < 60 THEN
      CO = CO + 1: C% (CO) = 15: CO =
      CO + 1: C% (CO) = 17: GOTO 214
      0
2120 IF NUMB > 59 AND NUMB < 100
      THEN CO = CO + 1: C% (CO) = INT
      (NUMB / 10): CO = CO + 1: C% (C
      O) = 17: GOTO 2140
2130 GOTO 2150
2140 NUMB = NUMB - INT (NUMB / 1
      0) * 10: IF NUMB = 0 THEN 21
      60
2150 CO = CO + 1: C% (CO) = NUMB
2160 FOR X = 1 TO CO: FOR Y = 1 TO
      AZ (C% (X), 0): P = AZ (C% (X), Y):
      GOSUB 2170: NEXT : NEXT : RETURN
2170 POKE Y2, P
2180 FOR X2 = 1 TO 75: NEXT
2190 RETURN
  
```

```

2200 REM READ IN ARRAYS FOR PHON
      EMES
2210 DIM AZ (19, 10): Y2 = 3 * 16 +
      49280
2220 FOR I = 0 TO 19: READ AZ (I,
      0): FOR J = 1 TO AZ (I, 0): READ
      AZ (I, J): NEXT : NEXT : RETURN
2230 DATA 4, 13, 51, 13, 63
2240 DATA 4, 45, 50, 13, 63
2250 DATA 4, 42, 55, 55, 63
2260 DATA 4, 57, 43, 44, 63
2270 DATA 4, 29, 38, 53, 63
2280 DATA 5, 29, 21, 34, 15, 63
2290 DATA 5, 31, 10, 25, 31, 63
2300 DATA 6, 31, 1, 15, 35, 13, 63
2310 DATA 5, 6, 33, 42, 3, 63
2320 DATA 5, 13, 8, 34, 13, 63
2330 DATA 4, 42, 1, 13, 63
2340 DATA 7, 5, 24, 59, 15, 2, 13, 63
2350 DATA 6, 42, 45, 59, 24, 15, 63
2360 DATA 3, 57, 58, 63
2370 DATA 4, 42, 44, 13, 63
2380 DATA 4, 29, 11, 29, 63
2390 DATA 7, 42, 45, 59, 13, 42, 41, 63
2400 DATA 3, 42, 41, 63
2410 DATA 9, 27, 50, 49, 13, 30, 23, 9,
      30, 63
2420 DATA 5, 21, 13, 30, 3, 63
  
```


COMPUTER programs often need to have a choice built in – either for or against a single action, or a choice between two actions.

Pascal permits both these kinds of choice by means of the IF – THEN statement. Let's look at an example:

```
IF X=0 THEN X:=1;
```

This statement changes a zero value of X to one. The expression after the word IF determines whether or not the statement following THEN is executed.

If the expression ('X=0') is TRUE, the statement following THEN ('X:=1') is carried out. If the expression should not be true, the statement following THEN is ignored.

The expression between IF and THEN must be such that its value is either TRUE or FALSE. These expressions, called Boolean after George Boole, an English logician, may be constructed using the following operators:

```
< (less than)
<= (less than or equal)
= (equal to)
> (greater than)
>= (greater than or equal)
<> (not equal to)
```

You can have 'X=1', 'X<=Y', 'X<>2*Y' and so on. Expressions can be combined with AND and OR, but make sure you use parentheses properly: to test that X is between 0 and 279 for example, you could write:

```
IF (X>=0) AND (X<=279) THEN
```

Without the parentheses, the Pascal compiler cannot understand this statement properly.

To choose between two options simply extend the IF_THEN statement to IF_THEN_ELSE. For example:

```
IF TERM = 0
THEN WRITE (TOTAL)
ELSE TOTAL := TOTAL + TERM;
```

which may be paraphrased: "If the value of TERM is zero then execute statement WRITE (TOTAL), but if TERM is not zero execute the statement TOTAL:=TOTAL + TERM.

Notice that only one alternative can be chosen. The general form of the IF_THEN_ELSE statement is:

```
IF expression THEN statement-1
ELSE statement-2
```

which is equivalent to the flowchart in Figure 1.

Notice the punctuation. In particular there cannot be a semicolon immediately before the ELSE clause.

You can group statements together in Pascal using BEGIN and END as a compound statement. A compound statement can be used anywhere that one statement is permitted. The three statements:

```
A:=1;
B:=2;
C:=C+1
```

can be grouped together as

```
BEGIN A:=1;
      B:=2;
      C:=C+1
END
```

forming a compound statement.

Remember that the indentation is intended to reveal the structure – that the three simple statements are grouped together as one entity. Also notice the punctuation.

Semicolons are used to

PASCAL

The second step

The first article in this series introduced the fundamental ideas behind the Pascal language and developed an outline program.

This month GORDON FINDLAY expands his discussion by introducing several new constructions.

He looks at extensions, conditional and compound statements, BEGIN-END pairs, loops and some examples using nested statements/conditions.

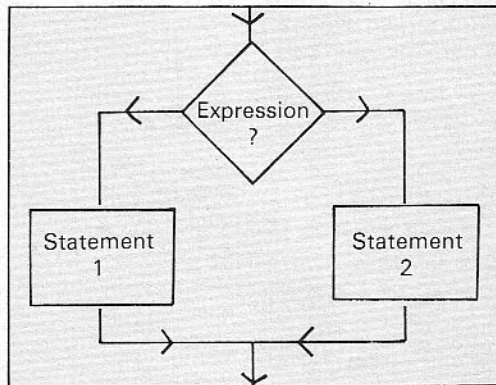


Figure 1

separate statements, not to end them, so semicolons are required to separate the first statement from the second, and the second from the third.

BEGIN and END act as punctuation – they are not statements, so don't need to be separated from anything!

Use of the IF_THEN_ELSE construction, with compound statements, permits neat expression of quite complex tasks.

In this next example three values A, B and C are supposed

```
IF (A>0) AND (B>0) AND (C>0)
THEN BEGIN
  S:= (A+B+C) /2;
  AREASQ:= S* (S-a) * (S-b) * (S-c);
  IF AREASQ >0
  THEN BEGIN
    AREA:= SQRT(AREA);
    Writeln ('AREA OF TRIANGLE IS',AREA)
  END
  ELSE Writeln (A,B,C,'DO NOT FORM A TRIANGLE')
END
ELSE Writeln ('TRIANGLES HAVE POSITIVE SIDES!')
```

Figure 11

to be the three sides of a triangle.

If they are all positive, and form a triangle, its area is output. If A, B, C cannot form a triangle, an error message is printed. (See Figure 11.)

Read this fragment carefully! Especially notice the way that the indentation (tries to) reflect the structure, and the punctuation.

Semicolons are a lot less common in Pascal than you might expect. Make sure that you pair up BEGIN-END and THEN-ELSE correctly.

An aside – equivalent coding in Basic or many other programming languages would require a number of GOTO statements – one third of some Basic programs consist of GOTOS.

Pascal does have a form of GOTO statement, but its use is considered bad form and it is not actually necessary.

In fact, Apple Pascal makes it difficult to use GOTO, requiring the setting of a compiler option. Avoid them – your programs will be much better.

The loop is another fundamental programming structure. Pascal is richly endowed with facilities for writing loops, but to begin with let's examine just one – the FOR construction.

This is very similar to the Basic FOR-NEXT loop, and looks like this:

For variable := start TO finish DO statement

Obviously "variable" stands for an identifier, of some previously declared integer variable. The statement is to be performed for each value of 'variable' from 'start' to 'finish' – inclusive.

So to print a table of squares from 1 to 10 we need a loop such as FOR I:= 1 TO 10 DO

PASCAL TUTORIAL

```

PROGRAM FIBONACCI;
VAR I, FIB1, FIB2 : INTEGER;
BEGIN
    FIB1:= 0;
    FIB2:= 1;
    WRITELN (FIB1);
    WRITELN (FIB2);
    (*TAKES CARE OF FIRST TWO!*)
    FOR I:= 3 TO 15 DO
        BEGIN
            FIB3:= FIB2 + FIB1;
            FIB2:= FIB3;
            FIB1:= FIB2;
            WRITELN (FIB3)
        END
    END
END

```

Figure III

WRITELN (I, ' ', I*I).

No word such as NEXT is needed to end a FOR loop, because only one statement is permitted after the DO.

But wait! Compound statements are statements. Put a compound statement after the DO, and you can control as many statements as you like. The program in Figure III prints the first 15 Fibonacci numbers.

There are a few things to notice about FOR-loops. To begin with, everything in a FOR-statement must be an integer (or whole number) — this includes the variable, and the starting and finishing values.

Unlike some other languages, Pascal does not allow you to choose the step size — it must be one.

FOR-loops can run backwards — just replace TO with DOWNTO. As an example, here is a countdown fragment:

```

FOR I:= 10 DOWNTO 0 DO WRITELN
(I); WRITELN ('BLAST OFF!').

```

At the conclusion of a FOR-loop, the controlled variable (I in the countdown example) is undefined. This means that the programmer cannot use the variable again without reassigning it.

In the countdown example, do not assume that I has the value 0 (or even -1) at the conclusion. This is a bit different to Applesoft.

To conclude here is a very heavily commented program,

```

PROGRAM PAYROLL;
VAR
    N,COUNT,IDENT : INTEGER;
    (* THESE MUST BE WHOLE NUMBERS *)
    HOURS, NETTPAY, GROSSPAY, TAX, RATE, TOTALGROSS, TOTALNETT : REAL;
    (*THESE MAY CONTAIN A DECIMAL PART *)
BEGIN
    (* START ACCUMULATIONS OFF: *)
    TOTALGROSS := 0.0;
    TOTALNETT := 0.0;
    (* FIND OUT HOW MANY EMPLOYEES *)
    READLN (N);
    (* THE FOR STATEMENT HANDLES REPETITION *)
    FOR COUNT := 1 TO N DO
        BEGIN (* BODY OF LOOP HANDLES ONE EMPLOYEE *)
            READLN (IDENT, HOURS, RATE, TAX)
            IF HOURS > 37.5
                THEN (* OVERTIME CASE *)
                    GROSSPAY := 37.5*RATE+ (HOURS-37.5) *RATE*1.5
                ELSE
                    GROSSPAY := 37.5*RATE;
            (* DEDUCT TAX - DATA READ WAS A PERCENTAGE *)
            NETTPAY := (1-TAX/100) *GROSSPAY;
            (* OUTPUT FOR EACH EMPLOYEE *)
            WRITELN (IDENT, HOURS, RATE, TAX, GROSSPAY, NETTPAY);
            (* KEEP TOTALS FOR ALL EMPLOYEES *)
            TOTAL := TOTALNETT + NETTPAY;
            TOTALNETT := TOTALGROSS + GROSSPAY;
        TOTALNETT := TOTALNETT + NETTPAY;
        END; (* OF THE LOOP *)
    (* OUTPUT SOME BLANK LINES THEN THE TOTAL *)
    WRITELN;
    WRITELN;
    WRITELN ('TOTAL GROSS PAY = ', TOTALGROSS, ' TOTAL NETT PAY
        = ', TOTALNETT)
    END. (* OF PROGRAM *)

```

Figure IV

using only the statements met so far, which runs a very simplified payroll.

The first piece of data read is the number of employees (N); then for each employee is read an identification number (IDENT), the number of hours worked (HOURS), the wage rate, per hour (RATE) and the tax rate (TAX), given as a percentage of the gross wage payable in tax.

Output for each employee is to be the identification number, the number of hours worked, the gross wage and nett wage after tax is deducted.

Overtime rates (time and a half) are paid for hours worked over 37.5; and the total gross and nett pay for all employees is also required. (See Figure IV.)



Of course, this is a very simple payroll — not a very practical application.

Later we will expand this program as far as input, output and other facilities.

● Next month's Pascal Tutorial will deal with strings.

POKE around and get things moving

Once you've got some method of POKEing, it's much easier to animate in Pascal than it is in Basic, says JONATHON LEWIS

EXPERIENCED Applesoft users may by now have discovered that you can get some very good animation effects by screen-switching. This means taking advantage of the two hi-res screens that Applesoft provides and using various POKES to allow you to draw on one screen while displaying the other.

At first glance it seems that this is a trick that Apple Pascal cannot cope with, since Turtlegraphics will draw only on one screen.

This is not true. Once you have got some method of POKEing, it is easier to produce animation in Pascal than it is in Basic.

The logic needed to get the effect is:

1. Show screen 2.
2. Draw the new design on screen 1.
3. Show screen 1.
4. Copy screen 1 to screen 2.
5. GOTO step 1.

The actual mechanics of the program are almost as simple.

The first thing to note is that the soft-switches listed in the Apple manuals aren't dedicated to Applesoft. They work whichever language you are using.

This means that once you have set up the Pascal equivalent of HGR (initturtle;) you can switch between the two HGR screens by referencing locations \$C054 and \$C055 (-16300 and -16299).

The second thing to note is that you can copy one screen to the other very easily by using pointers. I have used a pair of

variant records to do the job.

If you examine the sample program on this page you will see that I have first defined a type of object called a PICTURE that is in fact 8192 consecutive bytes, and followed this by defining a type of object that can be either an integer (so that I can set it to 8192 or 16384) or a POINTER to a PICTURE.

With this setup I merely set one POINTER to point at screen 1, and the other at screen 2. I then tell Pascal: "The thing pointed at by the first pointer is to become the thing being pointed at by the second pointer", and the whole screen moves in one step.

It's exactly the same as Applesoft does when you tell it

```
1 (*$***)
2 program wheel;
3 uses transcend,turtlegraphics,peeklib;
4 const
5     screenswitch=-16301;
6 type
7     picture=packed array[0..8191] of char;
8     picturepointer=record
9         case b:boolean of
10             true:(pointer:^picture);
11             false:(address:integer)
12         end;
13 var
14     count:integer;
15     x,y:array[0..360] of integer;
16     ch:char;
17
18 procedure initialize;
19 const
20     centrex=80;
21     centrey=80;
22     radius=50;
23     radians=57.29;
24 var
25     count:integer;
26 begin
27     for count:=0 to 90 do begin
```

Program to draw and rotate a nine-point star

LET A = B, but on a much larger scale.

As a demonstration of the technique, the sample program draws a nine-pointed star (see Figure 1), then rotates it through 91 positions. The program consists of three routines with a short driving section.

The first routine is a fairly

efficient way of calculating the coordinates of 360 points on the circumference of a circle of radius 50, centred at 80,80. The second routine copies one screen onto the other.

The third routine draws the star in the given colour by starting at the given position (start) and stepping round every 80th point on the circle. (If you want to change the star, just change the constant step.)

The animation isn't as fast as the Applesoft equivalent, and it is much harder to get into the machine code routines to speed things up in Pascal. And you can't run long programs with animation, as the program code and data in Pascal fill memory from the top downwards, so you either set the HEADPOINTER, and consequently run out of memory very quickly, or you find yourself destroying your program as you copy into screen 2.

For short simple diagrammatic programs, however, the effect is far more satisfying than either static displays, or the flickering displays you get by normal Pascal drawing techniques.

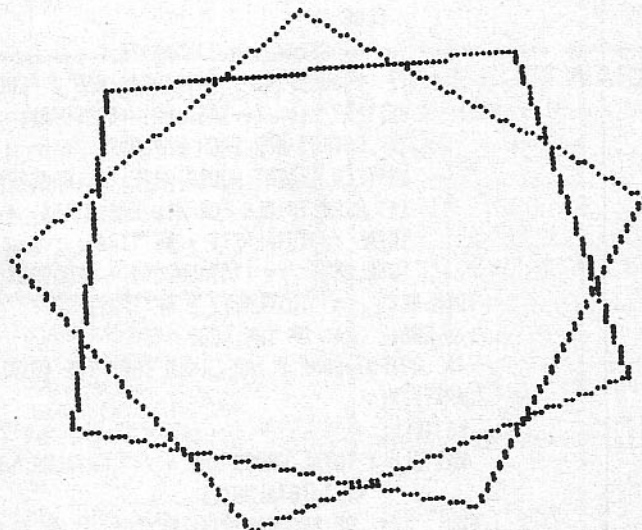


Figure 1


```

28  x[count]:=round(radius*cos(count/radians));
29  y[90-count]:=x[count];
30  x[180-count]:=-1*x[count];
31  y[90+count]:=x[count];
32  x[count+180]:=-1*x[count];
33  y[270-count]:=x[count+180];
34  x[360-count]:=x[count];
35  y[count+270]:=x[count+180];
36  end;
37  for count:=0 to 359 do begin
38    x[count]:=x[count]+centrex;
39    y[count]:=y[count]+centrey
40  end;
41  initturtle
42 end;
43
44 procedure copyscreen(first,second:integer);
45 var
46   pic1,pic2:picturepointer;
47 begin
48   pic1.address:=8192+first;
49   pic2.address:=8192+second;
50   pic2.pointer^:=pic1.pointer^
51 end;
52
53 procedure drawpattern(whatcolor:screencolor;start:integer);

```

```

54 const
55   step=80;
56 var
57   where:integer;
58 begin
59   start:=start mod 360;
60   where:=start;
61   pencolor(none);
62   moveto(x[start],y[start]);
63   pencolor(whatcolor);
64   repeat
65     where:=(where+step) mod 360;
66     moveto(x[where],y[where])
67   until (where=start);
68 end;
69
70 begin
71   initialize;
72   for count:=0 to 90 do begin
73     copyscreen(1,2);
74     poke(screenswitch+2,0);
75     drawpattern(black,3*count);
76     drawpattern(white,3*(count+1));
77     poke(screenswitch+1,0)
78   end
79 end.

```

Lower case text for everyone!



One of the plus points about the Apple IIe is its ability to display upper and lower case characters on the screen – something that has usually not been possible on the Apple II without an expensive modification.

A special offer for Apple User readers is a lower case generator that will enable you to have this valuable enhancement for just £25.

And that price includes a useful pair of chip extraction tongs (to ensure you don't bend any of the pins), installation instructions and a small Basic listing, plus copies of helpful articles on the subject.

This product is compatible with most Apple computers. However, some of the earlier versions (ie. Revisions 0-6 Mother Boards) require an adaptor board in addition to the generator chip itself. The cost of this ancillary board is £15.

(Users of the older Applewriter I Word Processing package should note that a

modification is needed before the program can use the generator. We can do this for you if you send a COPY of your program, together with the additional sum of £2.50.)

Allow 28 days for delivery




ORDER FORM

Lower case generator at £25

Adaptor board at £15

Total £

Cheque enclosed

Credit card   

No.

Name

Address

Send to: Apple User, FREEPOST, Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY. No stamp needed if posted in UK.

APPLE recently published its latest list of software packages which can be used on its machines but – incredibly – it contains no mention of Visicalc. I wonder why?

Unlike Apple I shall continue to look at Visicalc and its secrets, and this month shall continue my discussion on designing cashflow models on Visicalc.

However first a few comments on how to design and use VC worksheets.

The makers of Visicalc recommend that some time is spent planning the layout of a VC worksheet before you start to enter any text, values and formulas.

It is easy to move VC rows and columns as well as to insert new rows and columns. However some unexpected problems could be avoided if you plan your movements on paper first.

Most VC users, myself included, do not heed this advice. After all, why use paper and pencil to do something which can be worked out on an electronic worksheet?

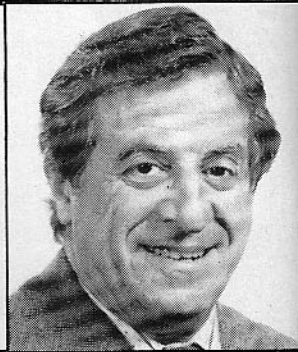
If you fit into this category take notice of the following advice.

Separate the various parts of your worksheet as shown in Exhibit I.

If the parts were adjacent to each other, one below the other, then inserting a row or a column in one part of the model, or moving a row or a column in another part of the model,

Make your cashflow calculations a model of perfection

NICK LEVY continues to unveil some of the secrets of VisiCalc



would affect the other two components.

But with a layout based on Exhibit I you can move, insert or delete columns in one area without affecting the layout in other areas.

This type of layout is not the prerogative of VC users with a vast amount of memory capacity in their computer.

Even Apple users with 64k memory can develop such block worksheets which stretch to, say, Z200.

Remember that when using this type of layout any empty cell between A1 and the bottom right hand cell in the model still takes up two bytes of memory, that is, it is as if you entered a two character label or value in the cell even though it appears to be empty.

The cashflow model which follows requires an introductory discussion.

Have you been wondering what is the purpose of the @ISERROR function in the VC

program? In that case try the following on your Visicalc.

Starting with a clear screen, put the cursor on C1 and type: **/F\$A1/B1*100 (Return)**.

The ERROR message which results is not because you made an error, but because VC gets confused when it has to divide zero by zero.

So now try this in C1: Type: **@IF(@ISERROR(A1/B1),0,A1/B1*100)**.

This time the response is 0. In this particular case you could also enter in C1:

@IF(B1=0,0,A1/B1*100) and VC will respond with 0.

The important thing is to remove the ERROR message not only for cosmetic reasons, but because such messages can perpetuate themselves.

Any legitimate calculation subsequently linked with a cell showing the ERROR message will also show ERROR.

Next, the question of forward and circular references. These

are calculations in which an earlier calculation depends on a subsequent calculation where the later calculation depends on the earlier calculation which depends on the later calculation – and so on!

Although the VC manual warns users against developing such worksheets, in the right hands this technique can be valuable.

We are now coming to the cashflow model itself, Exhibit II.

The top row shows the Minimum Lending Rate – what use to be called the Bank Rate – prevailing on any particular month.

Cells C2 and C3 show how much interest above the MLR the company is being charged for short term borrowing, and how much interest below the MLR they can earn for short term deposits.

Row 4 and 5 show the prevailing interest rates for borrowing and for depositing respectively.

These are annual interest rates and must be converted to equivalent monthly interest rates.

Note that strictly speaking monthly interest rate is not 1/12 of the annual rate, but less than that (because 1 per cent per month compounded is more than 12 per cent a year).

Row 6 is a calculation of the interest to be paid or to be gained every month, depending whether the company is depositing cash or borrowing cash from the bank.

The formulas in each cell in row 6 check whether the total receipts each month exceed total payments or vice versa.

If the cashflow is from the company to the bank then the formula will apply one set of calculations. If the cashflow

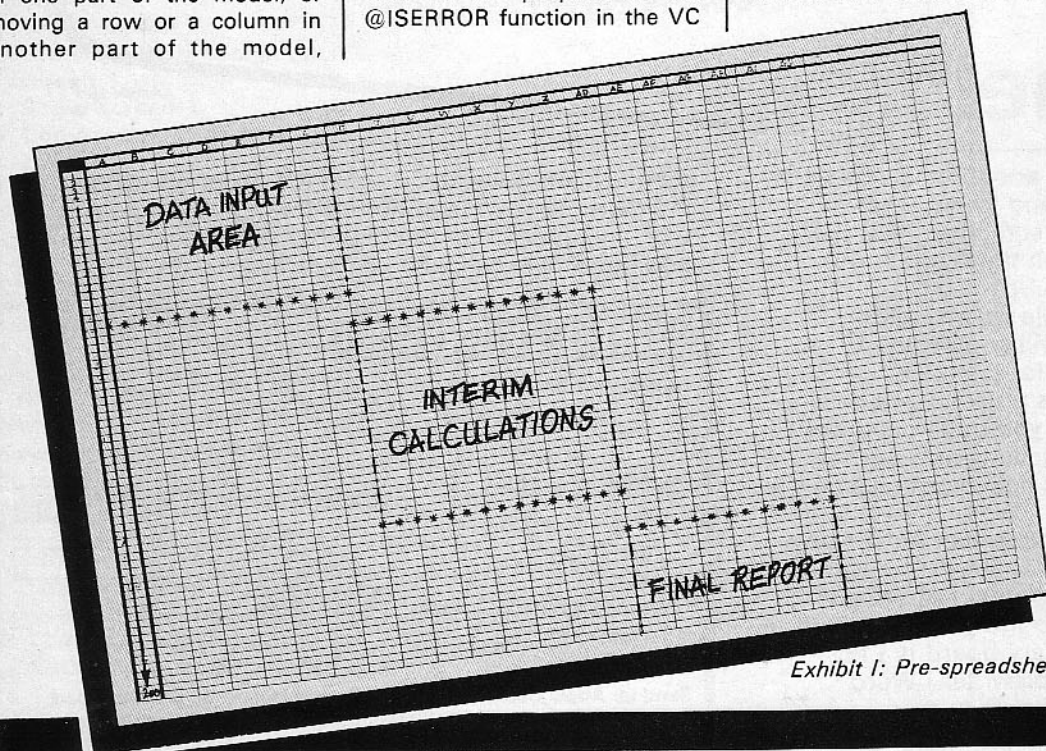


Exhibit I: Pre-spreadsheet

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1 Bank Rate (2)																
2 + For Borrowing	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
3 - For Deposit	2.50															
4 (1) Interest on Borrowing	4.50															
5 (2) Interest on Surplus	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00
6 Interest Received / Paid	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00
7	36.27	7.65	46.51	82.07	76.14	-76.29	-745	-175	-151	-128.00	-59.82	-12.81				
8 CASH BUDGET																
9																
10 OPEN BALANCE / OVERDRAFT	10075	385	1993	12464	13046	10623	-22623	-19129	-10805	-14930	-7059	-3119				931
11																
12 RECEIPTS:																
13 Credit Sales																
14 Bank Interest																
15 Other Income	18500	18500	19000	19000	19000	19000	20000	20000	13000	20500	21000	21000	21000	21000	21000	228500
16 Sales of Assets	36	8	47	82	76	0	0	0	0	0	0	0	0	0	0	249
17 Loans Receivable	0	0	825	82	76	0	0	0	0	0	0	0	0	0	0	825
18																
19 TOTAL RECEIPTS	18536	18508	35872	19082	21076	20825	20000	20000	13825	20500	21000	21825	251049			
20																
21 PAYMENTS:																
22 Goods and Services																
23 Wages and Salaries	7000	7000	7500	8000	8000	8500	9000	9000	6000	9500	9500	98000				
24 Taxation	4900	4900	5500	5500	5300	5500	3500	2500	6500	6500	6500	6500	17126			
25 Dividend (net)	17126															
26 Capital Expenditure																
27 Loan Repayments																
28 Bank Interest	5000	10000	5000	10000	40000	1762										
29 Loan Interest	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																
31 TOTAL PAYMENTS	29026	16900	25400	18500	21300	34070	18507	11475	17951	12629	17060	17775	260992			
32																
33 CLOSING BALANCE	385	1993	12464	13046	10623	-22623	-19129	-10805	-14930	-7059	-3119	931	TRUE			

Exhibit II: Cashflow model

reversed another set of calculations will be applied automatically.

The formulas for row 6 may appear formidable but this is only because the annual interest rates have been converted to true monthly interest rates.

A listing of the formulas is shown as Exhibit III. The list was prepared with a program by William Schertz of Batavia IL which appeared in InterCalc, the International Spreadsheet Users Group.

Considering that the interest payments are based on the average between the monthly opening and closing balances it could be argued that such precision in calculating the true monthly interest rate is not necessary.

On the other hand if we were to say that 12 per cent per year is the same as 1 per cent per month, then readers of *Apple User* would have flooded the Editor's office with letters spotting the inaccuracy!

Incidentally, I think that one of the best features in my past series of articles was the apt titles. These headlines are the creation of an anonymous *Apple User* sub editor to whom I am most grateful.

Being a perfectionist I usually find that on second reading of any of my articles I often feel like rewriting them all over, but this desire never applies to the titles. (We blush unseen, Anon.)

Returning to our cashflow model, I shall leave it to the readers to try and simplify the

formula in row 6. As a check, note that if the interest rate for January had been 1/12th of the annual interest rate then cell D6 would register 37.54 instead of 36.22.

Remember that Visicalc copies row 6 either into row 14 (if the cashflow is positive) or into row 28 (if the cashflow is negative).

This in turn affects row 33, which in turn affects row 6, which in turn affects row 33 and so on.

Fortunately we only have to press the exclamation mark

three times before the figures finally settle down.

This process is known as iteration, and some spreadsheets like Multiplan can control the number of iterations required to get the desired degree of accuracy.

Now if you were to calculate row 33 as row 10 plus row 19 minus row 31 you would get the right answer but you wouldn't keep it for long.

Let me explain. After saving your model with all the correct answers and loading it again from the disc, your model would be full of ERROR messages. Pressing the ! will not help.

What happens is that as a

Visicalc worksheet loads into the computer (bottom first), an ERROR message arising from O/O creeps in. As all the cells in the model are interdependent this ERROR message is perpetuated throughout the worksheet.

The only way to overcome this problem is to use the @ISERROR function combined with the @IF function in row 33.

One company faced with a similar problem used to rewrite the formulas in row 33 every time they loaded their model.

That tedious process has been eliminated now that they have discovered how to use the @ISERROR function.

Exhibit III: Part of the listing of the formulas

```

>P33: +033=(P19-P31+D10) >033:@IF (@ISERROR (D10+D19-D31)
>N33: @IF (@ISERROR (N10+N19-N31), 0, N10+N19-N31)
>M33: @IF (@ISERROR (M10+M19-M31), 0, M10+M19-M31)
>L33: @IF (@ISERROR (L10+L19-L31), 0, L10+L19-L31)
>K33: @IF (@ISERROR (K10+K19-K31), 0, K10+K19-K31)
>J33: @IF (@ISERROR (J10+J19-J31), 0, J10+J19-J31)
>I33: @IF (@ISERROR (I10+I19-I31), 0, I10+I19-I31)
>H33: @IF (@ISERROR (H10+H19-H31), 0, H10+H19-H31)
>G33: @IF (@ISERROR (G10+G19-G31), 0, G10+G19-G31)
>F33: @IF (@ISERROR (F10+F19-F31), 0, F10+F19-F31)
>E33: @IF (@ISERROR (E10+E19-E31), 0, E10+E19-E31)
>D33: @IF (@ISERROR (D10+D19-D31), 0, D10+D19-D31) >M31: @SU
>031: @SUM (D22...D29) >N31: @SUM (K22...K29) >J31: @SU
>L31: @SUM (L22...L30) >K31: @SUM (H22...H29) >G31: @SU
>I31: @SUM (I22...I29) >H31: @SUM (E22...E29) >D31: @SU
>F31: @SUM (F22...F29) >E31: @SUM (D28...D28) >028: @I
>P29: @SUM (D29...D29) >P28: @SUM (D28...D28) >028: @I
>N28: @IF (N6<0, @ABS (N6), 0) >M28: @IF (M6<0, @ABS (M6), 0)
>K28: @IF (K6<0, @ABS (K6), 0) >J28: @IF (J6<0, @ABS (J6), 0)
>H28: @IF (H6<0, @ABS (H6), 0) >G28: @IF (G6<0, @ABS (G6), 0)
>E28: @IF (E6<0, @ABS (E6), 0) >D28: @IF (D6<0, @ABS (D6), 0)
>P26: @SUM (D26...D26) >P25: @SUM (D25...D25) >P24: @
>P23: @SUM (D23...D23) >P22: @SUM (D22...D22) >P19: (
>O19: @SUM (D13...D17) >N19: @SUM (N13...N17) >M19: (
>L19: @SUM (L13...L17) >K19: @SUM (K13...K17) >J19: (
>I19: @SUM (I13...I17) >H19: @SUM (H13...H17) >G19: (
>F19: @SUM (F13...F17) >E19: @SUM (E13...E17) >D19: (
>017: @SUM (D17...D17) >014: @IF (D6>=0, D6, 0) >K1
  
```


THE Bank Street Writer runs on a 48k Apple II or a IIe with one or two disc drives plus a monitor.

In addition to the program disc you require one or more blank, formatted (prepared) discs on which to save your work files.

You cannot save files on the program disc itself and, because of this, if you are working with only one disc drive, then you must remove the program disc and replace it with the data file disc whenever you want to save or retrieve your work.

This is inconvenient and in a classroom situation means a great deal of disc handling — which can prove time-consuming and expensive.

The program has three operating modes — Write (used to enter text), Edit (for changing, deleting or moving blocks of text) and Transfer (for saving/retrieving work and printing).

You are automatically placed in Write mode after booting the program disc.

Text appears in the box (see Figure I) as it is typed, but unlike some word processing programs there is no wraparound at the end of a line. This is helpful for the beginner.

Remember though that the text on the screen will on most occasions have a different format when it is printed on paper.

Error correction is simple using the backspace key. But if you have an Apple II+ or earlier, capital letters are rather awkward to obtain.

It takes some time to get used to pressing the SHIFT and CTRL N keys to capitalise.

There is no such problem with the Apple IIe as the program recognises the SHIFT and shift lock keys for capital letters.

There isn't room for much text inside the screen box, although as you can scroll backwards and forwards this isn't too severe a problem. Beginners would find this "box" concept useful. It is just like typing on a piece of paper, with easily recognisable borders.

To edit (or alter) your work you must change from Write to Edit mode by pressing the ESC key. Then choose which option you require by pressing the left

Replacing chalk with cursor?

PAT BAKER looks at a system designed with schools in mind

or right arrows key if you have an early model Apple (see Figure II), or using the "Open and "Closed" Apple keys on a IIe.

This is somewhat tedious and complicated compared to other word processing programs such as Applewriter.

Cursor movement is simple using the I, M, J and K keys on older Apples and the up, down, left and right arrow keys on the IIe to ERASE unwanted words and phrases. In case of error it is a useful feature to have UNERASE (if you can tolerate the word) and MOVEBACK.

It is more difficult to MOVE blocks of text around and this will take practice to master.

The FIND command enables you to search easily through the text for a word or short phrase, and this can be used with REPLACE when appropriate — for example, to correct mis-

spellings.

When your text is ready to save or print you must enter TRANSFER mode by highlighting the TRANSFER menu at the top of your screen and pressing the RETURN key (see Figure III).

Transfer operations include preparing new discs for use, saving or retrieving files on to a data disc, renaming and deleting files, and producing a printout of your work.

It is very simple to save and retrieve files, though you must remember to remove your program disc from the drive and replace it with a data disc.

It is not clear from the manual which is the precise moment to do this, but a little practice makes it an easy operation.

When it comes to printing a file (from the Transfer mode), there is a useful menu item called PRINT-DRAFT.

Product: The Bank Street Writer

Description: Simple word processing program designed for schools

Authors: Scholastic Inc, 720 Broadway, New York NY 10003

Distributors: Broderbund Software (available from most Apple dealers)

Price: £44.95

This gives a quick printout of your file without the need to set detailed printing parameters.

It is a printout of exactly what appears on the screen (38 characters per line) with wide margins and triple spacing between lines.

It is useful for proof reading as it is easy to locate on the screen places where corrections need to be made.

When the document or file is ready for final printing, select PRINT-FINAL.

It is then necessary to answer seven or eight questions — about line spacing, page numbering, etc — before printing begins.

This can be very time-consuming and tedious, as all the questions must be answered each time you wish to get a printout. However auto repeat on the Return key takes you through the options quickly.

It is worth noting that at this stage you cannot change the left hand margin nor the top or bottom margins on the page.

In order to change them, you must use the Utility program on the disc — but that is quite complicated.

You can enter page numbers and page headings, but for some reason when I did this the program did not print the headings entered!

In general though my efforts at printing were successful, and I produced some impressive looking letters and reports.

It is also possible to merge documents before saving or printing them.

One of the most attractive features of this package is the User's Handbook.

It comprises a reference

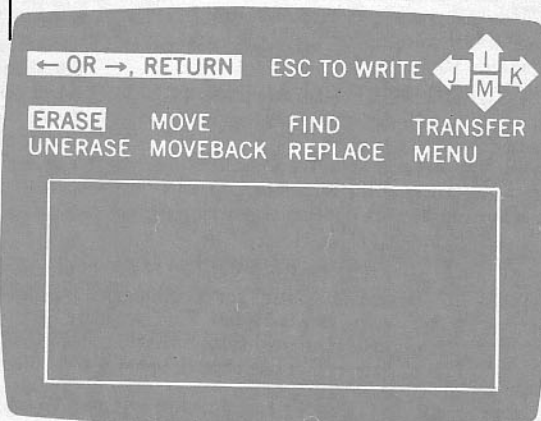


Figure I: Write mode

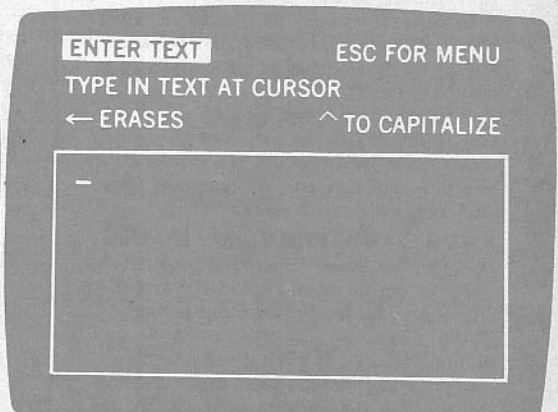


Figure II: Edit mode

guide, which is a comprehensive set of instructions for using the BSW program, together with a clear and simple student guide which is handy as a quick reference.

There is a question and answer section to help with beginners' problems, and also some student activity sheets to help the teacher.

On the reverse side of the program disc there is a tutorial program for the first time user. The Bank Street Writer is an attractive package which gives quick and satisfying results.

Menu selection is simple to master using the ESC key for a change of mode and the arrow keys for items within a mode.

A major disadvantage once you have learnt how to use the program properly is the small amount of text displayed at any one time.

I also found that, with only one disc drive, I was constantly changing from program disc to data disc – and vice-versa.

The PRINT-DRAFT facility

was very useful for quick correction of errors and omissions, but in PRINT-FINAL mode I would have preferred fewer questions about the printing format.

The manual is clearly laid out and easy to read.

Bank Street Writer is not a suitable tool for the modern hi-tech office but it wasn't designed for that role anyway.

It is a very good, simple-to-use package for teaching elementary word processing and text-handling skills.

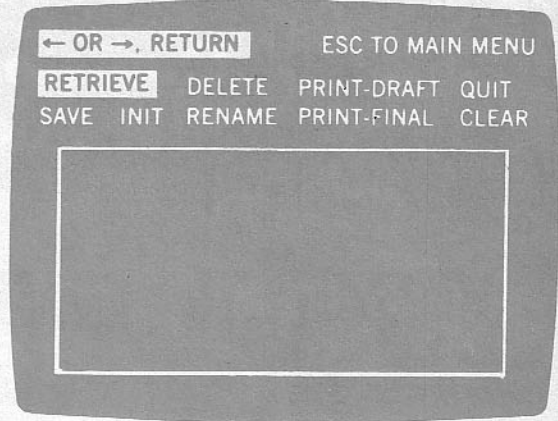


Figure III: Transfer mode

appletip

t When reading text file names into Applesoft programs (see Windfall, October 1983) it is possible to restrict a CATALOG listing to one type of file and display only the filename (not the sector count, lock asterisk, etc).

Use the following short patch to DOS:

```
ADAE:4C CA AD
ADDB:BD C8 B4 29 7F C9 04 D0 41 4C
13 AE
```

The number underlined should be set to indicate the file type to be listed, that is (in hexadecimal) Text = 00, Integer = 01, Applesoft = 02, Binary = 04, S-type = 08, R-type = 10. Both locked and unlocked versions are listed.

Another useful location to patch is \$B203. This holds the number of characters in the filename.

It is normally \$22, but can be reduced if you wish to store details such as date at the end of a filename but do not wish this data to form an "active" part of the filename.

R.A. Royal

NEW FROM CIRTECH CACHECARD16 AND CACHECARD64

THE ULTIMATE IN PRINTER CARDS AT UNBELIEVABLE PRICES!

A full function parallel printer interface with unmatched features AND a MASSIVE printer buffer on one card at amazing prices.
ONLY £90.00 for CACHECARD16 and £130.00 for CACHECARD64

Why wait for your SLOW... printer when you can buy a UK designed CACHECARD for less than most unbuffered printer cards.

FEATURES:

- * 16,000 or 64,000 character buffer controlled by on-card Microprocessor.
- * Full CP/M, PASCAL and APPLESOFT compatible.
- * Screen text echo (can be turned off).
- * 40 or 80 column instant screen dump (80 col. on IIe with 80 column text card and II+ with Videoterm type card).
- * Full function bit-image screen dumps: – Pages 1 and/or 2, double width, inverse, rotate, AND, OR, EX-OR, IIe double-density and full bit density support for Epson FX and Star Gemini printers.
- * Bit-image pictures can be positioned anywhere on the page.
- * Can send ASCII 128 to 255 codes.
- * All features controlled by one interpreter control code giving maximum printer and software compatibility.
- * Control function interpreter can be turned off allowing straight-through text printing guaranteeing compatibility with any software or printer.

Place your order NOW with your dealer or direct with
CIRTECH on 0383 729770


A stand-alone buffer will be available soon as an add-on if you already have a printer card.

Cirtech also manufacture a range of cards for your APPLE IIe or II+, for example a printer card with the same specification as above but without the buffer for only £32.00. Contact your dealer or CIRTECH for more information.

TWO WAYS TO ENSURE YOU GET

 **apple user**
EVERY MONTH

1. Complete and mail subscription form on Page 77
2. Hand this form to your newsagent.

Please reserve me a copy of  **apple user** magazine every month until further notice.

- I will collect
 I would like it delivered to my home.

Name _____
Address _____

Note to newsagent: Apple User should be obtainable from your local wholesaler, or contact the distributor – Walls, Gardner, Darton & Co Ltd Tel: Faygate 444

THE KeyStar is a device that attaches to the Apple by means of a serial card and provides an easy way to enter WordStar commands.

It is a small keyboard, about 9in by 6in with 52 pretty coloured keys and it has the relevant hidden WordStar command printed by the side of each key.

The keys are laid out symmetrically, with the function groups in different colours.

There is a small, easily read manual that details, briefly, all you need to know about the device.

When I received the review copy of KeyStar, I thought the hardware was pleasing to the eye, convenient, and a good idea.

I use WordStar regularly, and yet KeyStar made readily available a lot of commands I had forgotten even existed in WordStar.

For a person who only occasionally uses this excellent word processor, you can use 22 screen lines instead of the 14 when 10 are occupied by the help screen display.

To get the commands, all you have to do is simply look at the new keyboard, instead of the screen, and push one key instead of up to three.

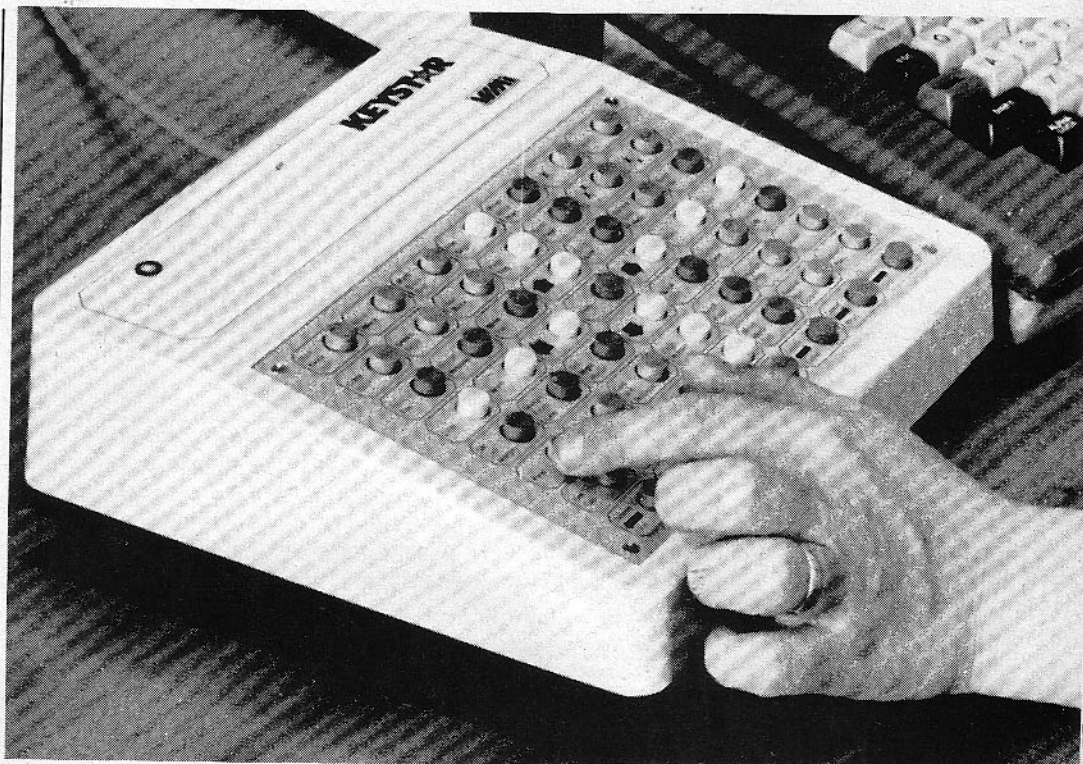
The first impression is of a large cursor control pad, laid out with one colour in a cross shape, quartering the board.

The quarters are then filled with keys of different colours that perform a related group of commands.

The cross contains all the cursor keys, with the steps taken getting progressively greater the further away from the centre you get.

In order to get KeyStar running, the user must patch WordStar so it will recognise the additional keyboard.

In order to use WordStar on your Apple system you require two additional interface cards — a Z80 card which allows the computer to run the CP/M operating system, and an 80 column card which enables the system to display 80 characters across the screen.



KEYSTAR

TERRY THOMPSON looks over a device that makes the WordStar word processing package even easier to use — but at quite a price, both monetary and ergonomical.

While the small KeyStar manual gives detailed instructions as to how this is achieved, these only apply to standard computer layouts.

If you have something out of the ordinary inside your Apple, such as a Vision 80 or UltraTerm, you have to work out the installation yourself.

Having installed WordStar dozens of times, I had no problem, but the new, non-technical user could find himself entering a minefield.

The detailed patches in the manual include those for other machines, including the IBM PC.

It's nice to see new hardware manufacturers making allowances for future upgrades in user

systems. Or is this foresight a spinoff from the recent university cash cuts?

As I used KeyStar some drawbacks began to become evident.

Although it is laid out as a cursor control block, it is much too big for the average sized hand.

That means you have to hold your hand above the new keyboard, with nowhere to rest when operating the keys.

It can become very uncomfortable.

The only alternative is to rest your hand on other keys, but this can give unexpected results.

It appears to me that no thought was given to the ergonomics of the board.

If there was, I apologise to those giants who tested it, but I'm a mere mortal and could have done with smaller, better laid out key areas.

Personally I would like to see the cursor block just that — a close coupled block positioned in the lower edge of the board, so that I could rest my hand while moving the cursor.

With that done, the more heavily used keys, such as block moves and storage commands, could be next to the cursor keys, and the least used keys out on the top edge.

The other drawback relates to the kind of keys used. They appear to be of the ordinary push button type, rather than being properly sculptured.

The buttons lie in an area larger than a normal key, in which the key's definition is written.

I also have reservations about the possible longevity of the keys.

I did take the opportunity of dismantling the hardware but could find no justification for its price at a shade under £200.

Obviously, there are the development and software costs but I feel that its price, in addition to the basic cost of WordStar, is somewhat over the top.



Turning your Apple into an organ

SYNTAURI claim to have pioneered a new era for music education and performance with the release of Simply Music, a keyboard learning and performance system for the Apple II.

"It is our response to the consumer's demand for a powerful yet simple to use musical instrument," commented Robin J. Jigour of Syntauri.

Simply Music consists of software, recorded music and custom music courseware which are used with the Simply Music instrument. The total system, including keyboard, software and Apple II computer, is said to be priced below many popular home organs.

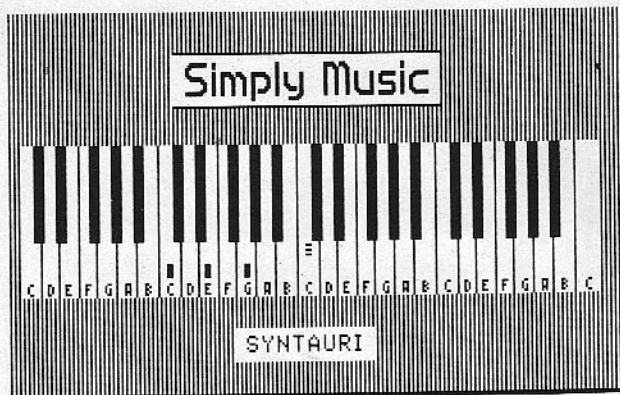
The major features which are used for learning, teaching and performing are:

- **MUSIC DISPLAYS.** Players have a choice of three video displays. During live play and playback, you can see the musical notes being played on a TV screen representation of the musical keyboard, or in musical notation of grand staff, or in a set of coloured bars corresponding to the notes played in each octave.

- **MATCH MODE.** For learning keyboard playing skills and to accurately follow a pre-recorded piece, the match feature causes the musical playback to pause and wait until you play the correct key on the musical keyboard and then continue.

- **MUSIC RECORDING.** For listening as well as learning, multi-part music pre-recorded with the Simply Music 16 part recorder is available. Students may play along with simple melody lines, or choose more challenging parts to follow with the screen display, and compose and record their own original music.

● Syntauri Corporation, 4962 El Camino Real, Suite 112, Los Altos, California 94022 USA.



Still not The Last One

THE Last One program generator has been enhanced in Version 3.0, now available.

Extra facilities include:

- The ability to produce programs in any national language.
- A Screensaver option which allows screen designs to be saved and re-used.

Retail price remains the same at £330 and existing users can upgrade to Version 3.0 for £39.

An optional extra for version 3.0 is Ghostwriter. As you create a program with TLO, Ghostwriter records every key stroke to a file.

This file can be edited by a word processor or text editor if required and played back using Ghostwriter.

You can recreate the same or modified program on any computer using a Ghostwriter TLO without touching the keyboard at all.

Retail price is £65 which includes upgrade to Version 3.0 for earlier users.

● D.J. 'AI' Systems, Station Road, Ilminster, Somerset TA19 9BQ. Tel: 04605-4117.

RAM drive emulator

A 310 sector RAM Drive Emulator for the Apple IIe has

been developed by Software Bank of Wisconsin, and is being marketed in the UK by Atlanta Data Systems.

It requires 128k memory (extended 80 column text card), is claimed to be easy to use, copyable and fast, and comes with a full set of utilities.

It can turn extra memory into a third disc drive and the manufacturers say you can load a 50 sector program in four seconds.

Drive 3 works with any program that uses DOS, and it allows you to FID files without stopping the drive.

● Atlanta Data Systems, 350/356 Old Street, London ECTV 9DT. Tel: 01-739 5889.

Tutorial chat show

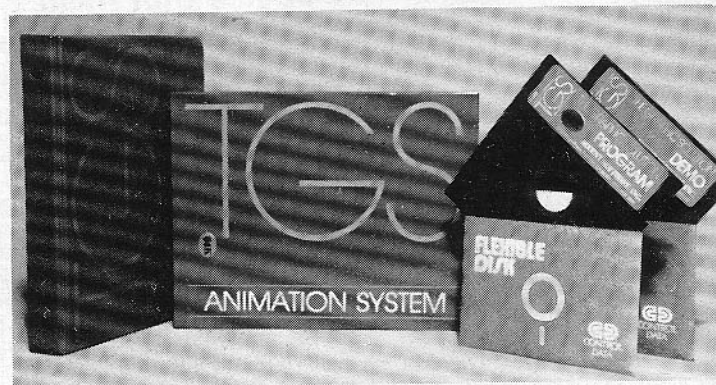
A CONVERSATIONAL tutorial on "How to Use Apple Writer", has been published by FlipTrack Learning Systems.

It is claimed that this interactive audiocassette course can train first-time users of Apple Writer to create, edit, correct and reorganise files, and also save documents and print them in a variety of formats.

"The learn-by-doing format allows users to work with their own documents, which is the most effective way to master Apple Writer", says Lee McFadden of FlipTrack.

"Learners also use Apple's flexible word-processing language (WPL) to merge their files and automate form letters and mailing lists".

With FlipTrack's patented cassette format a flip of the tape at various stages of a lesson allows the learner to pursue optional special interest topics, hear instructions for equipment



'Movies' animation

A GRAPHICS editor and animation system for combining graphics and text in a manner similar to making a movie film is The Graphic Solution.

It is claimed to be an invaluable tool for developing anything from training aids to 3D graphs and charts.

Features include speedy editing capabilities, and dual lo-res/hi-res operation to magnify work in detail for fast accurate

editing.

Multiple backgrounds and hidden-line animation can show the proper overlay of objects when one crosses another.

TGS costs £99 and runs on a 64k Apple II or IIe, ROM Applesoft, DOS 3.3 with one disc drive, a colour TV or monitor.

● Pete and Pam Computers, New Hall Hey Road, Rawtenstall, Rossendale, Lancashire. Tel: 0706-227011.

variations or try a quiz.

Simple instructions return the learner to the previous position on the front side of the tape, and the lesson continues.

The tutorial provides a complete hands-on lesson in only three spoken-voice sessions of about two hours. It is supplied with an indexed operator's guide for £54.95.

● *Micro Software International, Goddard Road, Whitehouse Industrial Estate, Ipswich, Suffolk IP1 5NP. Tel: 0473-462721.*

Printer interface

A FULL function printer interface incorporating a large printer buffer on one card is Cachecard from Cirtech.

The 16k card is priced at £90 and the 64k at £130.

It is fully CP/M, Pascal and Applesoft compatible and has 40 or 80 column instant screen dump (80 column on Apple IIe with 80 column text card and Apple II+ with Videoterm type card).

Bit image pictures can be positioned anywhere on the page and it can send Ascii 128 to 255 codes.

All features are controlled by one interpreter control code giving maximum printer and software compatibility.

● *Cirtech, P.O. Box 29, Dunfermline, Fife. Tel: 0383-729770.*

Estimates come easier

A SUITE of programs for the building and construction industry that claims to reduce the repetitive office work involved in preparing cost estimates and schedules of work for jobs that use standard elements, is Microspec.

The architect, designer, surveyor or contractor selects the elements or 'Tasks' for the job in hand, from a standard library built up over previous jobs.

The quantity, length, area,

time or alternatively a prime cost or provisional sum is then entered. When the tasks for that job have been selected Microspec produces a cost estimate.

If this is within the cost limitations for the job then the tender documents are produced with no additional input. When work is due to begin a schedule of works can be produced listing all the tasks by trade classification, with or without the quantities shown on the document.

Microspec offers a number of refinements for producing the total cost estimate. Money rates stored with the tasks can be tuned for inflation or location and by individual trade factors.

Valtec, the authors of the program, have a standard library for renovation and repair work. Containing 250 tasks, in 10 trade sections, the rates have been set by a quantity surveyor.

The standard library is said to be a useful starting point for anyone involved in this type of work.

The program, which is CP/M based and requires an 80 column card, costs £475 and the Valtec Library £200.

● *Valtec, 345 Gray's Inn Road, London, WC1X 8PB. Tel: 01-833 0105.*

Epson daisywheel

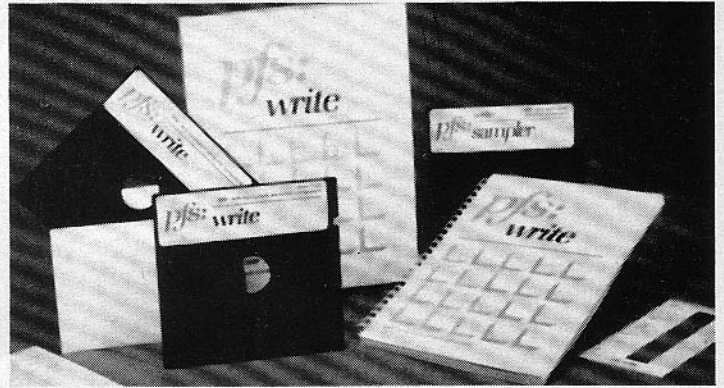
DUE out in the Spring is a new offering from Epson, the DX-100 daisywheel printer.

The DX-100 operates at 13 characters a second in bidirectional mode and provides a choice of printing using shadow printing, proportional spacing, automatic underlining, super and sub scripting as well as two-colour ribbons.

Double strike printing is also available for headings and additional emphasis within text.

There is a 3k buffer as standard (5k optional). This buffer may be used for repetitive printing of standard text and is accomplished by pressing the Copy key and the same text will be printed continuously.

Depressing the copy key a second time clears the buffer for



Aim is easier WP

AN addition to the PFS family of software, PFS:Write, is now available through distributors Pete and Pam Computers.

It is claimed to give the power of a word processor with the simplicity of a typewriter. All functions are in English and "Help" instructions appear at the touch of a key.

The screen is designed to resemble a blank piece of paper and the text reformats itself to

accommodate any changes. If revisions are needed the touch of a key makes it easy.

It gives documents a professional touch with features like boldfacing and underlining, centering and justification, page headings and footings and automatic page numbering.

PFS:Write is available for the Apple IIe and the IBM PC at a retail price of £98.

● *Pete and Pam Computers, 103/5 Blegborough Road, London SW16 6DL.*

new text.

Two-colour printing is available using separate ribbons instead of dual colour ribbons, thus extending the life of the lesser-used colour. Characters, words or phrases may be printed in a second colour for emphasis.

Super- and sub-scripting are available allowing fractions, exponents or other special characters to be printed in this way. This feature is ideal for scientific papers as well as trademark printing.

The printer will cost around £450.

● *Epson (UK), Dorland House, 388 High Road, Wembley, Middlesex HA9 6UH. Tel: 01-902 8892.*

Big tape back-up

A CARTRIDGE tape for the Apple III is the Sysgen-II-G which was developed originally for General Electric's Informa-

tion Service Company (Geisco).

Manufacturer Sysgen claims that its advanced cartridge tape technology enables its streaming tape to back up 5mbytes of data a minute.

"Our cartridge tape is able to back up information faster because of the way that we verify the data transfer", said Richard Newsome of Sysgen.

"We call this verification method 'verifying-on-the-fly'. The reading head is positioned just behind the writing head to verify that the information has been written correctly.

"If it hasn't, the incorrect data is rewritten and appended to the end of the tape. Other tape backups require a separate pass to verify the data thus increasing the time required to complete the backup", Mr Newsome explained.

The Sysgen-II-G is available with either 10 or 20mbytes of hard disc and streaming tape. The Sysgen-II-G-10 retails for \$3,295. The Sysgen-II-G-20 retails for \$3,995.

● *Sysgen, 47853 Warm Springs Blvd., Fremont, California 94539. Tel: (0101) 415-490 0900.*

into memory (for example about 30 pages or longer), the ALS card with CP/M Plus took about three times longer than the Softcard to move the cursor from the beginning to the end of the document when using CTRL-Q-C. When the document could all fit into memory, then the ALS card was faster.

Tom Derwent did not mention that CP/M Plus automatically logs-in the disc in drive B.; unlike CP/M 2.2 where you have to type CTRL-C if you wish to write to a disc. The error messages from CP/M Plus are far superior to the error message (BDOS ERR on B: BAD SECTOR) that you get with CP/M 2.2, and in some cases CP/M Plus will "recover" gracefully after an error has occurred, and you do not need to reboot the system.

I am surprised that Tom Derwent thinks CAT.COM is more useful than DIR.COM. There are so many options with DIR.COM on CP/M Plus that surely DIR is more flexible. For example, DIR [FULL] gives a large amount of useful information about the directory. No mention was made of the HELP.COM file either.

I suspect that Mr Derwent had very little time to do the review, and I think you really must give the reviewers plenty of time, otherwise reviews on complicated products such as this will confuse your readers rather than help them to make a decision.

An important point for a first-time user of the ALS CP/M Plus card is only to switch on the Apple after removing the disc drives from on top of the Apple. On my IIe, even with the increased shielding and earthing of the two disc drives, the system would not boot correctly until I moved the drives away from the Apple itself.

Two final points: Bob Ackerman (ALS director of sales) implies in your article that ALS's Revision B CP/M Plus supports the Grappler printer interface. Now it so happens that I have a Grappler, and it will not work with the version of CP/M Plus that I have. Since I registered with ALS quite some

time ago, you might have thought that they would have had the courtesy to inform me that (perhaps for a nominal charge) I could have my software updated. They didn't. Luckily, I also have an Apple parallel printer card, which works perfectly with CP/M Plus.

As for the adverts that tell you that the DR graphics extension is included - well you should know better now that you have read Mr Derwent's article. I think the "temporary" unavailability of GSX-80 should be pointed out in the adverts, since one of the reasons I purchased the card in the first place was because I thought that GSX-80 would be "in the box" when it arrived at my house! - **W.J. Hill, Brighton.**

Keep your printer happy

I HAVE just purchased Applewriter II which I am now using in place of Applewriter I. I am very happy with the many new features it contains, but am having some trouble in establishing compatibility with my printer.

I use the package on two systems, one at work and another at home. The computer at home is causing problems as it refuses to print out in anything in excess of 40 columns wide.

I have a Centronics printer and a cheap parallel interface card.

I would appreciate any help you may be able to give with this problem. - **Henry Stephenson, Canterbury.**

● Applewriter II may not work properly with some parallel interface cards. Often the interface requires a CTRL-I sequence if it is to print wider than 40 columns.

To achieve this in Applewriter II all you need to do is insert:

**CTRL-V CTRL-I CTRL-V
132N**

at the start of each file.

Use the CTRL-V before and after the CTRL-I to enter it into the file. Page 51 of the Apple-

writer manual gives more details.

Not all that random

I USE the RND function in Applesoft to produce random numbers. This would not appear to be an unreasonable requirement, as surely that is what it is there for.

I find however that the degree of randomness leaves something to be desired.

A measure of the reproducibility may be made by placing two Apples side by side and running the Kaleidoscope program from the System Master disc simultaneously on each machine.

This program makes use of the RND function to produce pretty pictures on the screen.

Even if the machines are left running for a considerable period of time, the so-called random patterns will be the same on each machine.

Is there any way I can improve this function so that it really does become random?

Incidentally, I like the new layout of the magazine. Keep up the good work. - **Keith Jacques, Sutton Coldfield.**

● The reason random numbers follow the same sequence is that neither DOS nor Applesoft initialises Applesoft's random number seed.

This seed is stored in locations \$C9 to \$CD. Whatever condition your memory comes up in will determine what the random number sequence will be.

The Applesoft random number generator, like all such routines, is only a pseudo-random generator, and non-random patterns will eventually occur.

The frequency of repetition of these patterns will vary from program to program.

Proper re-seeding of the random number generator during a program will help prevent the appearance of small repeating sequences.

Here are two suggestions from Apple:

□ Use the monitor's random

seed at \$4E and \$4F to initialise Applesoft's random number seed.

The monitor's seed is constantly being incremented while waiting for a key to be pressed. This will start you on one of 65536 different sequences.

10 X = RND (-PEEK (78) - PEEK (79) * 256)

□ The random sequence can be lengthened by re-seeding the generator occasionally within the application program.

Add the statement:

X = RND (-RND (1)).

Note that no method will completely eliminate identifiable patterns in the random numbers generated, but we can lengthen the sequences until they aren't recognisable.

Controller for ITT 2020

I HAVE an ITT 2020 computer with one disc drive. I would like to try to connect an 8in disc drive to the standard disc controller card.

But I am having difficulty in finding information about the controller card and wonder if you can help me.

I am especially interested in the pin connections to the disc drive from the card and I am also interested in the signal applied to the disc drive. - **Dipak Karadia, Bolton.**

● It is possible to modify the standard Apple 5¼in disc controller card so that it works with an 8in drive - but we do not recommend it.

The information you require can be found in the circuit diagrams in the DOS 3.2 manual. However, at the end of the day, you will have effectively rebuilt your controller card. It would be much better to buy an 8in card from an Apple dealer. A standard card costs between £200 and £250.

Our recommendation would be a Vista controller card which costs £350 but is worth it. It comes with the necessary boot software for Pascal, DOS and CP/M and is also "intelligent" - that is, it can cope with single-sided, double-sided, double density, etc.

Really bad software products aren't often featured in the editorial pages of *Apple User* — we prefer to give publicity to the better ones.

However, *Word Weaver III*, a word processing program for the Apple III, is a package to avoid and we wanted to warn unsuspecting users against parting with the £100 asking-price.

STEVE LASLETT'S review also makes some excellent points about how not to write and present software.

I HAD hoped that the attractively presented but brief documentation supplied with *Word Weaver III* was an indication of a straightforward, easy to use system.

The reality was very different!

Having experienced some initial problems loading the software (which can only be done after first loading *Business Basic*) it was clear that the screen displays were designed more for a computer enthusiast than a typist or casual user.

It was also clear that the screen help facilities were virtually non-existent, and that the first-time user had to rely on the documentation which itself was bad.

Instead of a clear step-by-step progress from loading the program to learning applications, the documentation invites the reader to choose from several alternative routes, without advising which is best.

This introduces immediate confusion, and requires reading ahead to several different sections before any progress can be made.

Nor is it made clear at this stage that the word processing software is in three separate modes — line, insert, and global.

The significance of these modes is not apparent, until attempting real applications. Then it becomes clear that the command codes, some mnemonic, others obscure, apply differently in different modes!

For example, on one mode command A means accept — that is, "write to file" — yet in another it means "show help display".

Oh, what a tangled web they weave...

(When it appears, the help display is a single line of characters, each clearly representing an option, but with no explanation of what they mean.)

Moreover, it is not clear from the screen whether using the command involves the control key, or typing just a capital A, or a capital A with inverted commas, as printed in the guide. In fact the inverts are not required.

The documentation suffers from many ambiguities, which serve only to further confuse the already confused reader. There are major omissions: no reference to word delete/line delete/para delete being typical.

More important fundamental weaknesses of *WW III* soon became apparent when creating a document.

There is no screen prompt to remind the user which mode he is in.

Nor is there any screen 'ruler' or other layout guide, or any status prompt. Worse, there are some essential requirements of WP what are not met:

- First, it is impossible to underline. (Either that, or the documentation is so bad that it does not cover this requirement.)

- Next, it is impossible to scroll back up screen to amend a line of text previously entered. In other words, errors detected at this stage have to stay there until the user moves from document creation mode to document editing mode.

This deficiency in itself makes the package totally unacceptable to this reviewer.

The fact that all format commands show on screen, and occupy a numbered screen line, is unfortunate.

Moreover, tab commands and indent commands are shown as a series of commands, and do not appear on screen in their correct position relative to printout.

This tab problem is one

example of the major weakness of *WW III* — that the screen display at document creation stage bears no resemblance to the final printed version.

It is reminiscent of the bad old days of magnetic tape word processing with no VDU.

How often the typist created a document, forgot to cancel tab commands at the end of a paragraph, but could not see the problem until after printout! It was then necessary to return to the format command line and try again.

More recent developments in word processing, both dedicated and with micro packages, have overcome these problems — *WW III* was a painful reminder of the bad old days.

Given the inability to scroll up screen when creating a document, it is impossible to change the format command sequence when creating a document.

This makes it even more likely that errors will remain undetected until print mode and it becomes a very complicated affair to create anything more than the most simple document.

Having created the document, and having left in the errors that could not be correc-

ted at draft stage due to the inability to scroll up screen, it is then necessary to call it back to screen in print mode to edit it.

With *WW III* the program actually has to be re-run, the document name entered again and recalled before any final editing can be done.

Moreover, the speed of text transfer from editing to print mode is painfully slow.

The package is very complicated and cumbersome compared with other packages. The first-time user would be thoroughly confused, and would probably not have been able to overcome the numerous problems created by the weakness of the program and bad documentation.


Being unable to underline, correct at edit stage, and no parity between screen display and printout are in themselves sufficient weaknesses to reject *WW III*.

Probable further problems that one would expect would arise from lengthy document preparation and editing — for example, block move-between pages, repagination, moving tables that span page breaks and so on.

Word Weaver III is not to be recommended. It takes word processing back to the stone age, and is guaranteed to put the first-time user off word processing for ever.

Title: *Word Weaver III*
Price: £100
Publisher: Synergistic Software

appletip

 When writing programs, first type the following routine on line numbers which will ensure it is at the end of the listing:

```
9000 CALL - 936
9010 VTAB 10; HTAB 15; PRINT
"GOODBYE !"
9020 VTAB 14; PRINT "SAVE DR
AW/SAVE/HGR2"
9030 POKE 37,12
20000 REM SAVE DRAW/SAVE/HG
R2
```

This will encourage the frequent saving of the program while writing, as it

economises on typing and will ensure the program is saved under the same name each time.

At any point in the program where an option is given include an Exit, either E or ESC, which takes it to the End routine.

When Run, and Exit option taken, line 9030 puts the cursor over the S of Save and all that's required is Right Arrow and REPT and Return.

On listing, all one has to do is bring the cursor up to the S of Save and do the same.

Frank H. Mallett