TIHOME

tidings

VOLUME 2 , NUMBER 3 MORDEN SURREY JUNE 30th 1982

TIHOME is not affiliated with Texas Instruments in any way and is supported only by its subscribers.

dingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdinc
ngsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsTIdingsT

CONTENTS.

THIS ISSUE RECEIVED 6TH AUGUST 1982

EDITORIAL.

This is the last Tidings you will see in the present format. The membership is now over 260 and rising and the problem of photocopying this newsletter has finally got out of hand. So it is out to the printer to get the job done with a good deal more professionalism. I have investigated the possibility of reduction, double-sided copying, covering, stapling, etc, etc, and I am sure you will be pleased with Volume 2 Number 4.

Apologies for the late delivery of Volume 2 Number 2 which was due to a number of reasons, not all of them under my control. I understand that some of the copying was not up to the usual standard. If you would like to return any page that you cannot read I will replace it for you. While I am on the subject of bad copying, it is useful to remember that the copy can only be as good as the manuscript. Budding authors should, therefore, check the quality of their typewriter ribbons.

I have been inundated with requests about the 99'er magazine. The only certain way seems to be to contact Emerald Valley Publishing Co, 2715 Terrace View Drive, PO Box 5537, Eugene, Oregon 97405; Telephone:- (503) 485-8796.

I am pleased to see the letter from T. I. in this issue of Tidings. I hope it is the first of a continuing series. I look forward to seeing it in every issue of Tidings. I consider it very important that T. I. should keep members informed of new developments, should answer letters & criticism and should notify us of known faults and problems together with their solutions.

ADVERTISEMENT/ NEWS ITEM:

SOFTWARE FOR 99/4A:

As you read this copy of TI HOME an additional supply of 99/4A programs becomes available.

To add to the programs by PRP COMPUTER GRAPHICS already available, programs by NORTON SOFTWARE of CANADA are now available from

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

Together with many UK written programs.

Full details of the Norton Software programs not available to meet the copy data of this issue, but the following titles may be among those offered:

** Graphics Package- Hi res sketching with saving to tape files
- line,circle & eclipse commands

Music Synthesiser-enter & play.

Lunar Lander- similar to the arcade game. This version has rotation of ship, inertia, and gravity. Not easy.

Tank - described as 'classic arcade game' vs computer or second player

Star Trek- difficult 3d version.

All above in TI BASIC or Extended Basic (different logic used in Extended Basic)

Attack Man- the monsters move across the screen as you wander the maze picking up points

Cross Country Car Rally- Very difficult this one. Drive from one side of the US to the other. 'Inspired by the cannonball run'.

Above two in Extended Basic only.

Fuller details in next issue if available. Should be able to supply when you read this.

Prices- same structure as PRP programs: £6 each, 2 @ 5.50 each.

** CORRECTION: Graphics Package is not now available in TI BASIC-sorry.

FOR SALE:-

Extended Basic	Speech Editor	Speech Synthesiser
Joy Sticks	Chess Module	Video Games 1

Any reasonable offer considered. Contact:-

D. J. FORTH
32 Hazel Way
St. Ives
Huntingdon
Cambs

Tel:- 04.806-8686 (Home)
02.233-50355 (Work)

Babbling Brooks

BITS & PIECES

Any well-organised article invariably finds itself being not well-organised, and such is the case this summer issue, as I write BB back to front and upside down. The result is that the middle was written first, the last section second, and the bits and introduction last, and all without the aid of a decent word processor.

You, the eagle-eyed reader, will of course have already spotted that the above excuse is just a cover for an admission of a silly mistake two issues ago, which appeared in the listings on the subject of Plotting, and will also have decided that as it is so simple, everybody else is BOUND to see it and write in, so you won't bother. Am I right ? The error lies in the listing on page III which has line 410 set to CALL KEY(0, K, T), and 420 testing for a key value which is not obtainable from keyboard 0. Either make the test for 49 instead of 19, or change the 0 to a 1. I don't know how that could have slipped past me, your honour.

On a slightly more serious note, we have a couple of casualties to report: member Reggie Rowe, from Oxford, 'Blockhead' to his friends, and addicted to the 99, was in collision with a motor vehicle in mid-May and has sustained some rather nasty injuries to leg, hand and head, but he is on the mend. Jill Lush, mother of member Dickon, has just recently come out of hospital after an operation, and I'm sure you all join with me in wishing them both a speedy recovery.

Member Bill Henderson, a chemistry teacher from Glasgow, has been in touch with me, and wonders if anyone is thinking of writing a program to produce bar-charts or histograms. I have plans to do so, but I thought I'd throw it open to anyone. The simplest form would make the bars horizontal; mine will be vertical when I can get around to it (no comments please), and unless anyone else beats me to it, will most likely form the basis for an article later in the year.

This issue's BB contains more moaning from me in Soapbox, a pair of items briefly covered in Science Watch, a short book review - but it's not a short book, - and the usual incomprehensible main topic.

There is a little glint of intelligence coming from TI Stateside: rumour has it that TI are coming out with a video disk controller, which shows that someone at least has a lot of smarts, although Apple did show them the way; now all we need is for someone to market it CHEAPLY. If the prices over here don't come down a great deal, I for one (and possibly two) will not be buying anything in the UK, but direct from the States. Even in Holland it seems that TI's pricing policy leaves a great deal to be desired...

Now all we need is for someone to develop something akin to the mini-memory module, but providing FORTH, which is a very interesting and useful language in that it is shaped by you the user, so YOU decide what functions you want. PASCAL who ?

DESIGNS FOR FUN

This issue's main topic involves the use of the 99's graphics, taking advantage of its particular method of creation to produce some (I think) startling patterns. I have already commented in a previous issue on my fascination with the fact that an apparently random collection of dots and lines can become highly patterned when vertically- and horizontally- symmetrical images are added. Throngh the short example routine given here you will be able to see noughts and crosses (no, this isn't the much-touted 3D Noughts & Crosses program!), faces, etc., etc., depending upon how much you've had to drink first. The program should work on all systems, although colour, - especially on NTSC systems (American colour set-up; I am given to understand that NTSC stands for Never The Same Colour!), where Black on Grey becomes Red, Blue, and Green on Grey, - will increase the pastel-coloured 'wallpaper' effect.

This particular version is an early prototype, which makes use of 121 of the 128 ASCII (pronounced 'Askey') characters available for redefinition, and simply sets up a pattern area in response to your selection, and continuously generates random patterns. The larger the pattern area used, the longer the generation time.(Had the designers of the 99's seen fit to provide the equivalents of PEEK, POKE, and USR - standardly available on every other machine - instead of making them HIGHLY expensive additional items, then this program could have executed many times faster. There are factions for and against fast execution of programs; I look at it this way: firstly, the whole raison d'etre of computers is their speed. Secondly, programs can ALWAYS be slowed down if they are executing too quickly (!), but they can't always be sped up, unless you go from a high level language like Basic, to a low level language like Assembler; see Soapbox on TI's pricing policy; or unless you use a compiler, which comes close to achieving the same result.)

By the time you get to read this (being written in late May, and I still haven't seen the April issue of Tidings, but copy date approaches, and I can't leave it any longer) there should be an advanced version of Designs For Fun available from the User Library, offering a number of facilities including changing colours, changing pattern areas, creating source strings, and so forth. Extended Basic owners will be able to reduce the listing size by using multi-statement lines, functions like RPT$ and so on.

If you manage to follow the explanation which appears later, you might like to add your own ideas by sending in details of source strings which you have found useful, or even programs which go further; don't forget, if you find this all too much, or heaven forbid, all too simple, then PLEASE do write in. I can't possibly hope to explain things at every member's level, so that I - we all - need your comments, helpful or otherwise. I keep giving my address, so that you have at least two targets to aim at: me for writing this garbage, and Paul Dicks for letting me. My work phone number is likely to change shortly, all being well, so I won't give that yet, and I'm not currently on the phone (I moved), so you'll have to put pen to paper if you want to air your views, either by writing to Paul, who will pine away unless you keep up a steady stream of friendly correspondence, or to me, who will pine away unless I keep up a steady stream of friendly correspondence (well, moans, grumbles, grouching and general abuse, actually, but you get the picture.).

The address: 68, Kelburne Road, Cowley, Oxford OX4 3SH. If you have something you'd like to include in Babbling Brooks, or something you'd like to see included, then write away, and we'll see what can be done.

How It's All Done

Take an image composed of dots and lines. Place a mirror on the right edge of the image, so that it is reflected. This gives you the Left - Right symmetry (LR). Place another mirror along the bottom edge, so that both images above are reflected, giving you Up - Down symmetry (UD). Now transfer the whole pattern to the screen and duplicate it over the entire display area.

In essence, that is the algorithm (or set of algorithms) which defines the principle of operation of the 'Designs For Fun' program...fanfare, pause for cheering, general celebrations, etc...Not an original title, I agree, but then, 'Wallpaper patterns' doesn't make you want to sit up and beg, either, does it ?

It's difficult to know where to begin explaining, and to 'begin at the beginning' is not helpful, as I'm not sure just where that is. (This is my tenth attempt to explain this particular program, and as you can see, I'm not doing very well!)

Let's start here: unless you've only recently bought your machine and joined TIHOME, (hello there), you probably have already covered the section in the manual on how to define your own shapes using CALL CHAR. (What do you mean, you haven't got that far; read it this instant!) To begin with, let's look at how the algorithm could be applied to the 8 x 8 dot matrix of a single character. In order for the finished image to fit within that matrix, the original shape will have to be confined to a quarter of that area: 4 x 4 dots in say the upper left hand corner. In order to show as clearly as possible what is going on, look for the page marked Figure 1. Ignore all the pretty shapes, and look down at the bottom of column 3, where you will see an 8 x 8 grid with an 'F' shape occupying the upper left hand corner. I've chosen to use such a shape because it will illustrate best the points I'm going to make in the burble to come. What I want you to do, dear reader, is to imagine that you are holding the afore-mentioned mirrors in place around that shape. Either work on the grid in column 3, or the clear grid in column 4, or turn to Figure 2 and work on the grid there. Alternatively, draw out your own set of squares on some scrap paper, or, if you're a budding MENSA member, do the whole thing in your head!

Working with either '1's or '0's, or black or white squares, fill in the mirror images, working preferentially row by row. Look at the top line of the 'F'. It is 4 black squares, or '1111' in binary (F in hex, and you can read THAT any way you want to!!!). The LR image of that line is also 4 black squares, or '1111', or F in hex. Draw them in, then. Next line: is one black square, three white ones, or '1000', or 8 in hex. The LR image of that is three white and one black square, or '0001', or 1 in hex. Draw that one in. (I hope you are drawing these on the right of the image, that boy at the back!). Next line is '1110' (in binary from now on), and the LR of that is '0111'. Finally, '1000', and we have already done this one, so we don't have to think twice about it do we, yes, of course, it is '0001'. OK so far ? Good. (If not, altogether now, Write In!)

Now, gentle reader who has been two steps ahead of me all the way, how do we do the UD image ? Yes, you're absolutely right, we just copy out the lines above, putting a copy of the top line at the bottom, a copy of the second at the seventh, a copy of the third at the sixth, and a copy of the fourth at the fifth. Now for a test of your tenacity. Translate it all into hex! Figure 2 shows how the image is split down the middle (metaphorically speaking), and also gives a list of the range of patterns and the hex digits which stand for them. If you've managed everything OK, the string you should end up with is: FF81E78181E781FF. Now, you may have begun to see a sort of pattern here, in the way that the string looks: an easy way to generate the UD image is to take the first half of the string and reverse it in pairs. So if the first half of the string is A5BDC3DB, then reversing in pairs gives DB, C3, BD, and A5, so the whole string would be A5BDC3DBDBC3BDA5. I hope you can follow that.

But how to obtain the mirror image for LR symmetry ? Again, you may be able to see another pattern emerging. From the string we worked out first in binary, you should be able to see that for every binary number lying between '0000' to '1111' there is a unique LR version. For '0000' it is still '0000', but for '0001' it is '1000', for '0010' it is '0100' and so on. Try calculating all the possible reflections in binary, and translating them into hex. For the numbers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F you should have come up with 0, 8, 4, C, 2, A, 6, E, 1, 9, 5, D, 3, B, 7, and F. Now look at listing 1, and scan down it until you reach the lines which assign strings to H$ and I$. H$ contains the numbers 0 to F, while I$ contains the corresponding LR digits, such that for a given position in H$, the corresponding position in I$ is occupied by the LR version of that digit.

What about a little teaser ? Go back to Figure 1 again, and look at the shapes at the bottom of columns 1 and 2. The left-hand image, which this time occupies an entire 8 x 8 dot matrix, has an LR image in the matrix next to it. The left matrix is a representation of the string '0123456789ABCDEF'. Try and work out, from the right matrix, what its definition string is. Why is it not the same as the string in I$??

We have, as you might have guessed, been working on what is really only the tip of the iceberg. Suppose that our 'F' shape was enlarged so that it occupied an entire 8 x 8 dot matrix ? Each of the LR and UD images would also occupy an equivalent area, so that if we wanted to put the entire pattern on the screen, we'd need four characters to be redefined. That enlarged 'F' shape is in position r1c1 in Figure 1.

In order to show the different images clearly, I have spaced out the characters, but of course there will be no such spaces on the screen.

The LR image of r1c1 is r1c2; the UD image of r1c1 is r2c1, and of r1c2 is r2c2. If we were to decide to allocate the characters with ASCII codes 33 to 36 the strings to enable the shapes in r1c1, r1c2, r2c1, r2c2 to be defined, such that r1c1 would be represented by CHR$(33), r1c2 by 34, r2c1 by 35, and r2c2 by 36, then the hex strings, which would need to be stored in an array in a way which allowed easy access, would be represented by Table 1; in the examples given here, the 'space', CHR$(32), has been left undefined so that you might later experiment with inserting spaces into the formatting of the display array, of which more later. The D$ array, used to hold the definition strings ready for use in CALL CHAR statements to redefine the relevent characters in order to produce the patterns, holds the necessary pairs of hex digits (with the LR images in the appropriate positions) for each line of the total pattern: a pattern area of 1 requires only 1 character and 1 pair of hex digits per line, a pattern area of 4 (2 x 2) requires 4 characters and 2 pairs of hex digits per line, a pattern area of 9(3 x 3) requires 9 characters and 3 pairs of hex digits per line, etc. It becomes easier to think of pattern areas in terms of the square root of the area - i.e., the length in characters of one side, - so that pattern area of 1 needs 1 character, 1 hex pair; 2 needs '2' characters (in reality 2 x 2) and 2 hex pairs; 3 needs '3' characters and 3 hex pairs, all the way up to a pattern area of '11' (i.e. 11 x 11 = 121) needs '11' characters and 11 hex pairs.

When it comes to creating the hex pairs (i.e., original random digit and its LR mirror), I have found that the easiest way to create the overall image was to work from the inside out. That means that starting with a blank array, the program will select at random a digit from S$. It will obtain the LR version of this digit by using H$, I$, and the SEG$ and POS functions. The initial random digit is then 'concatenated' - stuck onto - with the existing digits in the array, and the LR version is then concatenated onto that. Let me run through an example, and see if you are any the wiser afterwards (ho, ho, ho!).

Before I can actually run through such an example, there is one item which still needs to be explained, and that is the 'source string', which is assigned to S$ in the listing. In order to exert some small control over the types of design which are produced by the program, I decided to make the program select the hex digits at 'random' not from H$ but from another string (S$), so that instead of being able to compose or create designs using the full range of 'shapes' possible using the hex digits 0 to F inclusive, the choice could be restricted to a chosen set of particular hex digits. In this way, different types of pattern could be produced, still with some semblance of being random, but falling into what might be termed 'categories'. For example, by assigning the digits F, E, C, and 8 to S$, the program can only select the shapes which are equivalent in binary to '1111', '1110', '1100', and '1000'. This will tend to produce patterns which are heavy and dark, and on the whole rather angular. However, using 0, 1, 2, 4, and possibly 8, the images will tend to be light and ornate, perhaps too detailed. In addition, there is the factor of pattern area: the larger the pattern area, the more dispersed the shapes tend to be, depending on the source string. Using a root area of 1 does not do full justice to strings like '12480'. The frequency and position of the chosen digits within the source string does also have a large part to play, but there the effects are complicated by the 'pseudo-random' number generator used by the 99; in fact, under certain circumstances, the PRNG (pseudo-random number generator) will generate the same series of numbers over and over again, which means that the patterns will also become cyclic; the use of RANDOMIZE without a 'seed' should obviate this.

So, now for a run through. For this example, let us assign '1248FO' to S$. Our PRNG is going to be required to produce an integer between 1 and the length of S$, which in this case is 6, for SEG$ to use, to select one digit at a time from the source string. This method of digit selection is also known as 'biassing'. Let us also decide on a simple root pattern area: 1. This means that we will have to produce 8 pairs of hex digits, with each digit in a pair being the LR image of its partner.

Using our PRNG, let us suppose that the first number produced is a '5'. The 5th digit in S$ is selected, which is an 'F'. We then use POS on H$ to find the position in H$ of the 'F', which is 16. Then we use SEG$ on I$, the LR image digit string, to 'cut out' the 16th digit from the string, giving us another 'F'. We concatenate these two 'F's together to give us 'FF', and the first line is completed. Once it has been completed, it can also be copied into both the first and the last elements of the definition array (to give UD symmetry), and we can begin compiling the next line. Let us suppose that this time our PRNG produces 1, so we select the 1st digit in S$, which is a '1'; this lies in position 2 in H$, and SEG$ on I$ at position 2 yields '8'. Concatenate the two digits together, and you have '18', which is another line completed, ready to be assigned to the second and seventh elements of the definition array (again, for UD symmetry). If our PRNG next produces a 2, the second digit in S$ is a '2', after processing, the LR is '4', and the concatenate is '24', which is assigned to the third and sixth elements of the array. And finally, if PRNG produces a 3, the third digit in S$ is a '4', and its LR is '2', and the concatenate is '42', which goes into the fourth and fifth elements of the array, giving us 8 pairs of digits. We can now assign the relevent pairs in the right sequence to a character using CALL CHAR. The definition string, which must be compiled from the definition array, is FF182442422418FF; to see the effect of that without running the program, in the Immediate Mode assign CHR$(159) that string: e.g. CALL CHAR(159,"FF1824424224 18FF") and ENTER, then CALL HCHAR(1,1,159,768) and ENTER, which will place the shape over most of the screen, and you should begin to get an idea of the possibilities as far as design creation is concerned.

Now we come onto the next factor in this extremely awkward explanation: formatting the display itself. Using HCHAR above is really only suitable for a root pattern area of 1: anything greater than that and HCHAR is not practicable. I have chosen to try and overcome the difficulty by creating another array containing the necessary characters in the correct sequence as strings, which are then PRINTed out.

In order to make the whole thing work, we will need a separate character to be redefined for each 8 x 8 dot section of the total pattern area, LR and UD symmetrical characters included. Thus a root pattern area of 1 requires only 1 character, but one of 11 will require 11 x 11 = 121 characters. We will need a standard way of setting these characters out, so that we can evolve a general method for producing the strings which will contain them. If you go back to Figure 1 you may be able to envisage the problem. Beginning the count at CHR$(33) - leaving the space, 32, undefined for future experiments, - a 2 x 2 pattern area (root = 2) requires 4 characters: in this case 33, 34, 35, and 36. I chose to make the system produce 33 and 34 on one row, and 35 and 36 on the next row. Similarly, with a root pattern area of 3 we need 3 x 3 = 9 characters: 33, 34, 35 on the top row; 36, 37, 38 on the middle row; and 39, 40, 41 on the bottom row. And so on, up to and including a root pattern area of 11.

You may disagree with the exact method that I am about to describe: if you think that you have a faster or better solution, please write in with it, and every serious suggestion will be carefully studied, as usual.

I decided to allocate each row in the pattern, in terms of characters, to an element in a string array, and to assign as many elements as there are rows of 8 dots in the pattern - in other words, a 5 x 5 pattern would use initially 5 elements with each element containing 5 different characters. In this fashion, the position of the characters in the array, and eventually on screen, would correspond to the 8 x 8 dot areas in the pattern. (There was recently an article in one of the popular micro monthlies which covered a related topic about addressing elements of an array, and I would say it was recommended reading. If I remember correctly the magazine was E & CM - Electronics & Computing Monthly, but the exact issue is not presently to hand. If there is sufficient interest, perhaps one of the editorial team (!) could be prevailed upon to write a short piece explaining some of the details).

The equation that I eventually came up with involved the use of two loops: one to index the elements of the array, and one to index the characters by ASCII code. It is possible to use an alternative technique without resorting to the use of the equation which I have developed - anybody any ideas on how it might be achieved ? You know what to do if you have!

However, this is only a part of the process: we have to extend this character pattern so that it covers the entire screen, by duplicating it in two directions, both laterally (sideways, i.e., by concatenating duplicates of the strings created so far) and vertically (by duplicating the contents of the elements from element 1 to the root-pattern-area-value element into the succeeding elements. If that sounds confusing, don't worry: as usual, all will be not-quite-clearly explained shortly!)

The swiftest way to duplicate laterally (unless you have XB - Extended Basic - in which case RPT$ is the swiftest way!) is to concatenate successive concatenates: i.e., take an original string a$, concatenate it with itself to produce a new string which is double the length of the original. Do it again, and you have a string which is quadruple the length of the original. Again, and you have, what, an octuple ? This is where L$ (see the listing) is used. Each of the root pattern areas produce formatted display strings which need to be duplicated or doubled a minimum number of times in order to make their length equal or exceed the 28 - character line length of a PRINT line. A root pattern area of 1 requires 5 such doublings, of 2 requires 4, and so does 3, areas of 4, 5, and 6 require 3 doublings, and so on. The loop control value for doubling is therefore obtained using VAL on SEG$ on L$, where the specific digit accessed is determined by the root pattern area. (I'm getting tired of typing root pattern area all the time; call it RPA for short.) So far, so good. However, the criterion for using L$ was based upon increasing, by doubling, the length of each formatted display string, until the string length equalled or exceeded 28 characters. Once the doubling loop is exited, we need to cut down the string just processed until

its length is exactly 28 characters, and that is very easily accomplished by using SEG$ again, specifying a start position of 1, and a length of 28, and assigning the 'cut' string back to the relevent array element. Once this has been done for all the strings in the array which have been created so far, we have produced only part of the array. It consists at present of a series of duplicated strings, numbering RPA in all. That is, if RPA is 3, the number of strings so far is 3; if RPA is 7, the number of strings is 7. The aim of the program originally was to fill the entire display area: this can only be achieved if we extend the display created so far, until it reaches the maximum of 24 lines: i.e., 24 elements. So all that is left is to copy the first element's contents into element 'RPA + 1', the second into 'RPA + 2' etc.

After this the entire array can be printed out, but because of the PRINT function, a pending PRINT condition has to be used, otherwise the top line is shunted off the display, leaving a gap at the 24th line. The pending PRINT condition is imposed by using the semi-colon (;) after each PRINT. The program can now loop round creating shapes at random as already described in outline.

If I can digress a little here (when do I do anything else ?), go back to Figure 1. You may have noticed that there are a number of those dratted 'F's spread over the page. The set in r1c1, r1c2, r2c1, and r2c2 have already been discussed. The others are primarily to show you what the effects are of placing the original 'F' in differing initial positions. In the group r1c3, r1c4, r2c3, r2c4, the original 'F' is in the top right hand corner, and in the next two groups the 'F' origin is in the lower left, and lower right, corners. Now rotate the page through 90^{0} - in either direction - and notice that the same 'F' shape now provides another 4 distinct pattern possibilities. You could also have the inverse (white on black) versions of all 8. What other ways can you think of manipulating just this one shape to give different patterns ? And this, by the way, is only a fraction of the total number of patterns possible from each RPA - and these are not duplicates, the rotation and transposition suggestions merely save me from drawing out all of the possible patterns. Actually, the range of patterns possible from each RPA is quite staggering:for an RPA of N, the total range of possible patterns is obtained from examining the binary equivalents of the range of hex definition strings possible, which is in turn equivalent to calculating 2 to the power of the total dot area of the origin, which is one quarter of the total pattern area in dots. Bear in mind though that these values are possible ranges only; the bias introduced by S$ and by the PRNG alters the range which you will see.

RPA	Orig. Area	2 ^ that:
1	4 x 4 = 16	65536
2	8 x 8 = 64	1.845×10^{19}
3	12 x 12 = 144	2.23×10^{43}
4	16 x 16 = 256	1.158×10^{77}
5	20 x 20 = 400	2.582×10^{120}
6	24 x 24 = 576	2.473×10^{173}
7	28 x 28 = 784	1.017×10^{236}
8	32 x 32 = 1024	1.798×10^{308}
9	36 x 36 = 1296	1.364×10^{390}
10	40 x 40 = 1600	4.446×10^{481}
11	44 x 44 = 1936	6.224×10^{582}

Of course, these ranges also include the extreme values which result in an all-black or all-white image; but if you feel a little suspicious about the enormity of the numbers shown here, try working them out. Start with an RPA of 1. The quarter area which is the origin occupies a 4 x 4 dot area = 16 dots. Those dots can range from being all-off, i.e., binary 0000000000000000 to being all on, i.e., binary 11111111 11111111. If you include those two extremes, it adds up to 65,536 intermediate values: 0000000000000001, 0000000000000010, 0000000000000011, etc., etc. An RPA of

2 has an original pattern area of 8 x 8 dots, (= 64 bit number), so the range, in binary, is: 0000000000000000000000000000000000000000000000000000000000000000 to 1111111111111111111111111111111111111111111111111111111111111111 puff..puff..puff. Need I go on ? In the case of RPA = 2, with an original pattern area equal to the size of one character matrix, you may be able to visualise the enormous range of patterns in these terms: if you were to write down all of the intermediate 64 bit numbers at the rate of one number every 60 seconds, it would take you around 6×10^{11} years of continuous writing. If you find that kind of notation rather confusing, it means that it would take you six hundred thousand million years!! That's about, oh, 10 thousand million lifetimes, assuming the average life expectancy is 60 years. What about an RPA of 11.....??

As is my wont, a reasonably detailed account of how the listing is composed :

Listing 1:

100: Clear the screen.

110: Set the screen colour to grey. The default colours of the entire character set are: Black foreground, Transparent background; so no other settings need to be made to obtain a black on grey colour scheme.

120: Use Option Base to set the minimum array element address.

130: Dimension the arrays to be used for formatting characters for display, and for holding the definition strings for the patterns as they are created.

140: Assign the standard reference string of hexadecimal digits 0 to F to the string variable H$.

150: Assign the LR version of the digits contained in H$ to another string variable: I$.

160: Assign the lateral duplication pointers as string digits to the string variable L$.

170: Assign the bias or source data to S$.

180: Assign the length of S$ to numeric variable S.

190: Define the pseudo-random number generator so that it will generate a positive integer between 1 and S, where S contains the length of S$.

200: Input (with a user prompt) the root pattern area to numeric variable Z.

210: Clear the screen.

220: Input validation: ensure that the content of Z is an integer number.

230: Input validation: ensure that the content of Z lies between 1 and 11 inclusive; the format of the statement is equivalent to INOR or Inclusive Or. If the test is failed (i.e., an invalid entry was made) return to line 200 for another entry.

240:- 290:

Initially assign a 'tile' shape in quarters to the first 121 re-defineable characters in blocks of four. This in itself is worthy of further study, as you will see when the initial printout is made after RPA is entered.

300 - 340:
Create the initial block of characters which will form the basis of the formatted display array.

350 - 400:
Duplicate the contents of the elements, and 'level off' each string to a standard length of 28 characters.

410 - 430:
Duplicate vertically down through the elements of the array.

440 - 460:
Print the contents of the formatted display array, using a 'pending print' condition to ensure that the printout fills the entire 24 x 28 display area.

470 - 490:
Re-entry point for the main routine: reset the contents of the elements of the definition string array to 'null'.

500 - 560:
Create randomly-selected definition strings and their LR symmetries and assign them to successive elements of the arry;as each line is completed, produce the UD symmetrical images and assign them to the inverse positions in the array.

570 - 670:
Having created the definition strings, they now have to be assigned using CALL CHAR to the relevent characters.

680: Re-enter the main routine, and repeat the process.

Additional experiments that you can perform involve removing the CALL CHAR in line 640 and placing it in 625, so that shapes change during the process of compilation, rather than after it. Also the sequence of compilation can be slightly changed: by concatenating the contents of A$ in line 620 to SEG$(D$(N), M, 2) thus,

```
620  A$ = SEG$(D$(N), M, 2) & A$
```

the shapes can be made to appear as though they are 'dropping' into place, albeit very slowly. It doesn't make any difference to the pattern being produced. Why ??

There are many improvements which you can make to extend the scope of this program, not least of which are the deletions which could be made in order to make the routine run a little faster, or occupy less space. Does anyone fancy trying their hand ?

What about a routine to SAVE a particularly pleasing pattern, or to print out the definition strings which went to produce it ? How about photographing the screen ? Unfortunately, because we are currently having Tidings Xeroxed, I think that colour reproduction is out (although I know that there are colour copiers), but I for one would be interested, and it would be nice to be able to produce colour printout from a CHEAP (funny word that, TI) Texas peripheral: it can be done. It's about time that TI stopped dumping the useless thermal printer on us, and produced a decent one, one capable of at least reproducing the 8 x 8 dots which go to make up the on-board character shapes!

I hope to have some follow-up to the program in the coming months, and I also hope that most of it will come from you, the reader!

Figure 1. Designs For Fun.

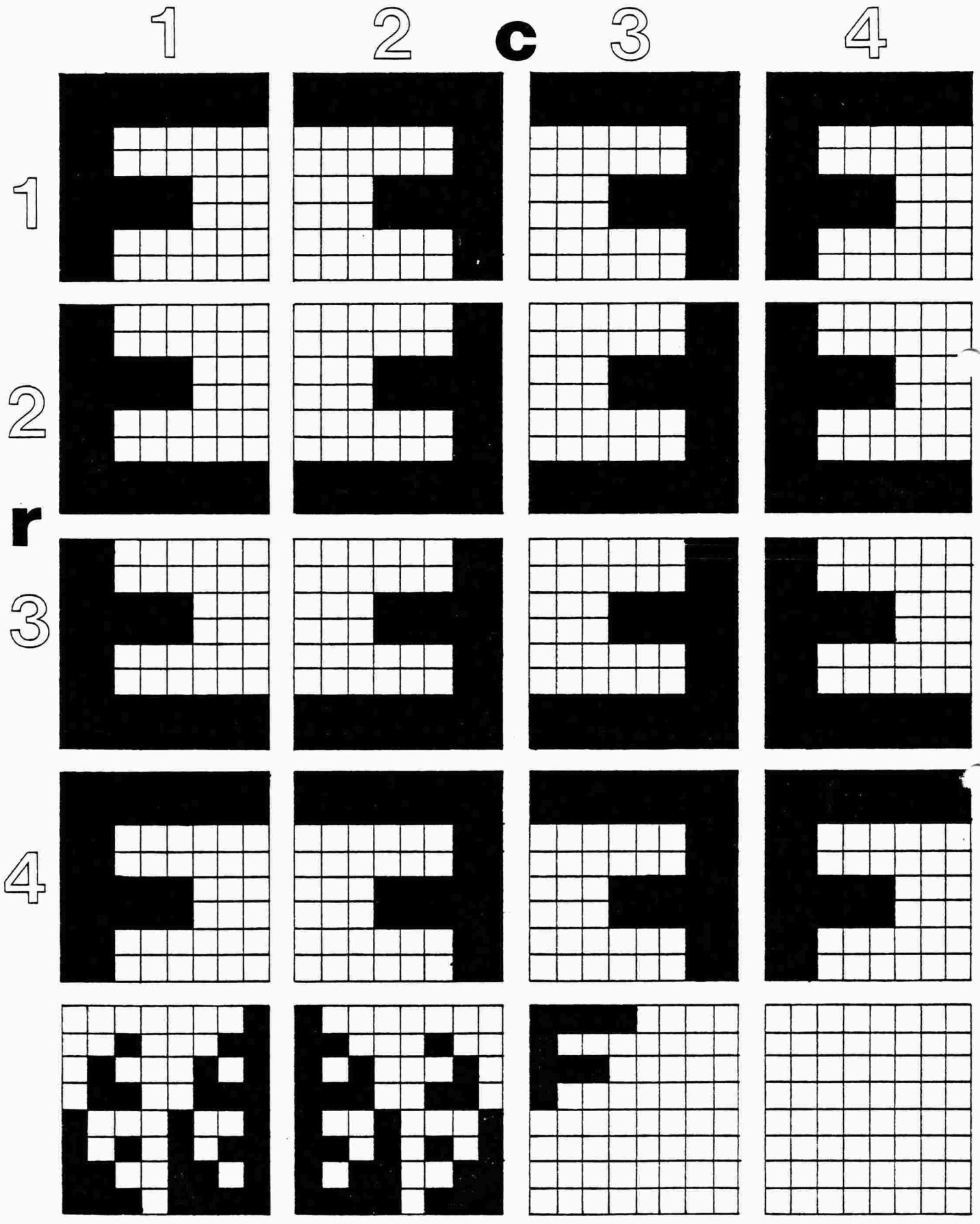
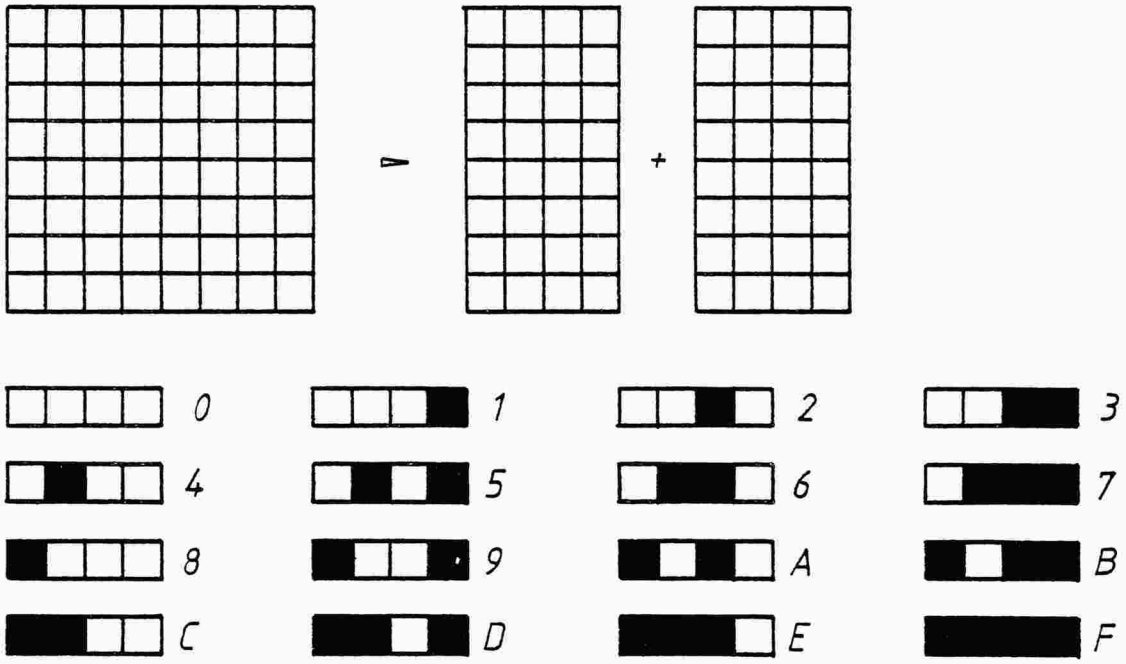


Figure 2. Designs For Fun



Listing 1

```
100  CALL CLEAR
110  CALL SCREEN(15)
120  OPTION BASE 1
130  DIM C$(24), D$(64)
140  H$ = "0123456789ABCDEF"
150  I$ = "084C2A6E195D3B7F"
160  L$ = "54433322222"
170  S$ = "01234789CEF"
180  S = LEN(S$)
190  DEF R = INT(RND * S + 1)
200  INPUT "ENTER ROOT PATTERN AREA:":Z
210  CALL CLEAR
220  Z = INT(Z)
230  IF (Z<1) + (Z>11) THEN 200
240  FOR L = 33 TO 153 STEP 4
250  CALL CHAR(L, "0103070F1F3F7FFF")
260  CALL CHAR(L+1, "80C0E0F0F8FCFEFF")
270  CALL CHAR(L+2, "FF7F3F1F0F070301")
280  CALL CHAR(L+3, "FFFEFCF8F0E0C080")
290  NEXT L
300  FOR L = 1 TO Z
310  FOR M = 1 TO Z
320  C$(L) = C$(L) & CHR$(32 - Z + L * Z
     + M)
330  NEXT M
340  NEXT L
350  FOR L = 1 TO Z
360  FOR M = 1 TO VAL(SEG$(L$, Z, 1))
370  C$(L) = C$(L) & C$(L)
380  NEXT M
390  C$(L) = SEG$(C$(L), 1, 28)
400  NEXT L
410  FOR L = Z + 1 TO 24
420  C$(L) = C$(L - Z)
430  NEXT L
440  FOR L = 1 TO 24
450  PRINT C$(L);
460  NEXT L
470  FOR L = 1 TO Z * 8
480  D$(L) = ""        i.e. null string
490  NEXT L
500  FOR L = 1 TO Z * 4
510  FOR M = 1 TO Z
520  A$ = SEG$(S$, R, 1)
530  D$(L) = A$ & D$(L) & SEG$(I$, POS(H$, A$,
     1), 1)
540  NEXT M
550  D$(Z * 8 - L + 1) = D$(L)
560  NEXT L
570  Q = 33
580  FOR L = 1 TO Z * 8 STEP 8
590  FOR M = 1 TO 2 * Z - 1 STEP 2
600  A$ = ""          i.e. null string
610  FOR N = L TO L + 7
620  A$ = A$ & SEG$(D$(N), M, 2)
630  NEXT N
640  CALL CHAR(Q, A$)
650  Q = Q + 1
660  NEXT M
670  NEXT L
680  GOTO 470
```

Example layout of D$ array and the hexadecimal definition strings for 2 x 2 block

ELEMENT	CHR LINE	CHR CODE	HEX 1	HEX 2	CHR CODE	HEX 1	HEX 2
1	1	33	F	F	34	F	F
2	2	33	F	F	34	F	F
3	3	33	C	0	34	0	3
4	4	33	C	0	34	0	3
5	5	33	F	8	34	1	F
6	6	33	F	8	34	1	F
7	7	33	C	0	34	0	3
8	8	33	C	0	34	0	3
9	1	35	C	0	36	0	3
10	2	35	C	0	36	0	3
11	3	35	F	8	36	1	F
12	4	35	F	8	36	1	F
13	5	35	C	0	36	0	3
14	6	35	C	0	36	0	3
15	7	35	F	F	36	F	F
16	8	35	F	F	36	F	F

Definition strings are selected from D$ so that chr 33 is defined by FFFFC0C0F8F8C0C0, chr 34 by FFFF03031F1F0303, 35 by C0C0F8F8C0C0FFFF, and 36 by 03031F1F0303FFFF.

D$ array contents are:

```
D$(1): FFFF          D$(9): C003
  (2): FFFF            (10): C003
  (3): C003            (11): F81F
  (4): C003            (12): F81F
  (5): F81F            (13): C003
  (6): F81F            (14): C003
  (7): C003            (15): FFFF
  (8): C003            (16): FFFF
```

This configuration is shown in Figure 1, consisting of images r1c1, r1c2, r2c1, and r2c2.

Table 1.

SOAPBOX

There are a number of items in this issue's Soapbox. The most pressing is caused by the recent arrival of a new machine on the market, a little dazzler from good old Uncle Clive Sinclair, called the Spectrum. Now what, you might ask, has that to do with TI ? Well, I'll tell you. This latest, British, product will be making many headlines over the coming months, and you don't need second sight to see that. Not the least of its many charms is the 16K + colour which you get for £125, and the 48K for £175, VAT inclusive. There are many things that I don't know, and one of them is why a 32K add-on for the Spectrum should cost £60, but for the TI costs nearly £300. The price may have fallen by the time you read this, but TI's pricing policy is one of my seven wonders of the world, not least because it has yet to be seen. In addition, the Spectrum offers loads of facilities, many of which ARE available for the 99's, but at an unacceptable price. For example, GROMs are now becoming available to permit the use of machine code routines without the necessity of forking out some £400 for Extended Basic and the 32K add-on: I'm thinking in particular of the Mini-memory and the Assembler/Editor. In the States these cost $99.95 or thereabouts, and apart from the fact that they should have been supplied as standard, the price is reasonable. In the States, TI are actually in the forefront of cheap computing. Not so over here: that acceptable $99.95 becomes a whopping £115 by the time it has crossed the strip of water separating the mighty US of A and us. Come on TI, we can buy a whole bloody computer for that price over here! Wake up whoever decided to charge us gullible Brits such a swingeing amount of our precious lolly and boot them round the office. If you haven't learnt by now that we don't have as much lolly to burn as the Yanks, and that we like things cheaper than we can get them, then your days are numbered. And the final nail in the coffin is the proposed cost of peripherals for this new Sinclair wonder machine: an RS232 + communications interface will cost a measly £20, compared with the £140 which TI are asking for their RS232; and to top it all, the microdrive disk system will require a forking out of the massive sum of £50, without the need for a separate controller. Put that in your pipe and smoke it!

Whatever the reasons for Uncle Clive's low prices, no-one is going to convince me that a) TI don't possess an Uncle Clive of their own, and b)TI can't sell their equipment as competitively - 99/4 NTSC owners are only too familiar with the way that prices can drop to suit the market! I'm not suggesting that TI can drop their disk drive prices to £50 a throw, but they sure as hell can drop them a damned sight lower than they are now. You have only to cast your minds back to the days of the good old British motorcycle: anybody selling a bike with ALL the accessories for a very cheap price was considered a cowboy by the established Brit producers - and where is the British bike industry now ?

TI's pricing has left a sour taste in the mouths of quite a few of our members, and I know of three who are not only going to cease spending any money on their machines, but who, sadder and wiser (and broker) are actively looking at other machines, amongst them Apples and Spectrums (Spectra ?). TI's approach, it would seem, is the old one of mini-computer manufacturers: hook your customer, and then force him to buy from you alone. It might make more money than any other method, but it won't make you many friends.

Another point is that one of the selling points for the 99's is the existence of the User Group; but I for one am not going to try and help sell a 99 to any prospective customer unless he/she has already made up his/her mind that no other will do, in which case the sale has already been made. I'm not the type who can say 'Yes, it is a good system' when I know damned well what problems are likely to lie ahead. Now some members will wonder what the fuss is all about, and others will doubtless say 'Who wants machine code anyway' and it is true that a small proportion of you will be more than happy with what TI provide as standard, even if your eyes are subsequently opened by events. But most of us will eventually contract the insidious programming disease, and like any addict, will be pushing for more and faster; a little like

youngsters when they first learn the rudiments of reading. Whereas before the simpler skills were learnt you would have been hard pushed to interest them in a book which contained anything but pictures, after they have been learnt everything has to be read: jam labels, garment washing instructions, the works. What TI have done is to provide the machine which is an excellent first computer, but failed to allow for growth in a sensible manner. It is a lot like teaching people to read, and then pricing all future books at £50 each, regardless of content. Perhaps some of you are quite content to fork out huge sums for items which everybody else gets as standard; the rest of us are not. The price of the 99's and their peripherals makes it an expensive mistake, and it will certainly alienate some customers from TI with respect to any future products that that company might produce, no matter how excellent the product.

Changing the subject somewhat, it has permeated down to me that Babbling Brooks' main item might be considered a trifle too long-winded for newcomers to follow, and I would certainly agree to that being a very strong possibility. However, I am faced with two problems: one is that most of the items explained here could be put over with much waving of arms and repetition in a very short time indeed, except that I can't be personally present to explain things to each and every one of you, and if I reduced the waffle to a manageable chunk each issue, not only would Tidings look very sparse (because of the second problem), but new members would need 10 backcopies in order to find out what had been going on. The second problem is that feed-back is a vital part of any group and specifically of any newsletter. So far only two people have got in touch with me with regard to my poor explanations, and I think, I hope, that I have been able to explain things satisfactorily to them, at least to clear up temporarily any difficulties which had arisen. If the rest of you aren't interested in what BB presents, (and I find it difficult to believe that ALL of you understand every word I say), then say so, and say what you'd like to see, or at least what you don't like. Even a postcard will do, as long as you include your details so that I can help out in any way. Would you like, for example, to see more on the Music applications, or on Speech, or what ? If you have any hardware experience, would you consider writing articles on interfacing other bits of equipment to the 99's ? Are you in the process of writing the most devastating Alien Mothers-In-Law program, and have you got stuck ?

The biggest enemy of any organisation like ours is apathy; Tidings is not like other popular computing magazines, whose readership need not contribute anything; we need you more than you need us: we need your opinions, your criticism, your experience, good or bad. And if you object to being described as apathetic, do something about it! Or are you too apathetic ?

If that doesn't put the cat amongst the pigeons, nothing will. The Babbling Brooks series can't last for ever, and I would dearly like to write less so that you can write more. Yes, it takes a lot of time to put together an article as an article, but on the whole it is just as beneficial to the author as it is to the reader: you tend to have to understand what you are writing about, and you also tend to discover that your programs improve and your general approach improves, as you are forced to organise what you say and do. (Except in my case, of course, where things just seem to get more complicated).

Bear in mind that Soapbox is NOT a substitute for the Quotes From Your Letters section: it is in addition to that item. Paul doesn't have space or time to reprint your letters in their entirety, but as long as space in BB is not at a premium, I am happy to let someone else, once an issue, sound off about their particular pet hate. Or love, of course. And anyone writing in to say how BB gives them a reason to carry on living, and makes their day for them, will naturally not be given any special treatment whatsoever. Well, not much, anyway....

S C I E N C E W A T C H

Josephson Junction and The Microrefrigerator

You could be forgiven for thinking that the title is a reference to some Fifties' skiffle group. You could be, but you won't. The Josephson Junction is perhaps a milestone in electronics, which is another way of saying that it is too complex a subject to do justice to in a few lines, but here goes.

Josephson Junctions are probably best described as superconducting switches. They make use of two phenomena: superconductivity and electron tunneling. Their speed of operation is such that they represent a significant advance on machines like the Cray 1, about which you might have read or heard something. The Cray 1 is probably the world's fastest computer (unless the Cray 2 is) with a 'cycle time' of around 12 nanoseconds - a nanosecond is a one thousand millionth of a second, so it is a fairly short period of time, although GPO counter staff are rumoured to be able to close a position in a shorter time than this, - while other high performance computers have a cycle time of around 30 to 50 nanoseconds. (Don't ask me what the 99 cycle time is; measure it with a calendar.) Devices incorporating the Josephson junction, however, are expected to have cycle times of around 1 nanosecond, although the actual switching speed of the junction is around six picoseconds - a picosecond being a one million millionth or one billionth of a second - and a factor of ten as a measure of increased speed can make quite a difference to some programs.

Like most components in computers, Josephson junctions (JJs) are based on semi-conducting technology, which means that it acts in a similar fashion to a transistor in that it can switch (in the sense of channeling, not in the sense of light switches) information. The major difference lies in the environment which is required to permit this high-speed operation, which is where we come across superconductivity.

Superconductivity is a subject which is outside my field, but I will do my best to explain it as succinctly as possible. It is a quality which is associated with very low temperatures - around minus 260 degrees Centigrade (or Celsius as it should be), where the electrical resistance to the flow of current approaches zero. I have seen quite impressive demonstrations where electric motors have been set in motion, the circuits have been cooled, and the power source disconnected, and the motors have carried on running. Resistance is not totally removed, but it is severely reduced, and it is this resistance which generates the heat produced by integrated circuits, requiring them to have 'heat sinks' in order to dissipate some of the energy.

Jospehson junctions can also store information, which makes them candidates for both central processor and memory components.

The second phenomena is rather more difficult to explain simply, as it involves the weird logic of Quantum Mechanics. An example which might help to outline the problem: a marble rolling along a flat surface meets a hill. As the incline increases, so the marble will need more energy in order to climb and travel over to the other side of that hill. However, in Quantum Mechanics, there is a finite possibility that the marble will travel THROUGH the hill, by 'tunneling'. Now, at the size level of a marble, that possibility is virtually nil. However, at the size level of say electrons this phenomenon begins to occur, and it is the basis of the semi-conductor operation in the JJ.

The only drawback is that although the computer based on JJs would be very small, about 5 cm^3 (in fact it couldn't be very much larger, as the distance for the signals from central processor to other devices would be so great that the system would be

slowed down, waiting for data to cover the distance), it would be dwarfed by the unit required to keep everything at the very low temperatures required, which also correspondingly increases the price both of purchase and of maintaining the system.

Enter the second item, the Microrefrigerator. It was invented as a cooling unit for instrumentation, not for beer, and is not much larger than a microscope slide, if you happen to know how big that is (2.5 x 7.5 cm). It is produced in a manner similar to that in which chips are manufactured (not the crinkle-cut kind), using photolithography. The layout of the gas lines, boiler, etc., is produced and a photolithographic plate is made. A photosensitive coating on a thin piece of glass is then exposed to this 'negative', creating a 'print' on the glass. A fine-grit microlathe is then used to cut away the channels through which the gas will flow, etching into the glass. Then another piece of glass is bonded to the first, creating a seal for the channels, and the etched grooves then become tubes. The unit can then function like any other refrigerator, it would seem.

At present, the mini-'fridge is capable of going down to minus 190 degrees C, near the temperature of liquid nitrogen, which substance is often used for keeping lab specimens frozen in the hospital where I currently work. In fact the temperature can get so low that the microrefrigerator has to be kept sealed within a vacuum container, otherwise any moisture would condense upon it in chunks of ice. The unit is expected to be able to be improved to bring its operating range down to tha required by JJ - based devices later this year, so that the possibility of having a domestic micro-mainframe comes a large step closer.

References: Scientific American, May 1980: 'The Superconducting Computer'.
Science Digest, February 1982: 'The Littlest (sic) Refrigerator'.

Book Review

16 BIT MICROPROCESSORS, mentioned last issue, has now given up most of its secrets to me, and although it will be a useful reference work with respect to the details given about a number of 16 bit mpu's, don't waste £10.45 buying it unless your interest extends quite some way beyond the TMS 9900 chip family. The chapter on the 9900 is badly put together, hence the delay in understanding on my part, and there is no logical, sequential list of the commands which the system can accept and execute. It is a good primer, but an expensive one, as far as explaining what the general format of the types of instruction is. Fortunately, thanks to Paul Dicks, I had the benefit of a Texas manual which helped a great deal, although I have to say that one without the other, with hindsight, would have been less than helpful. The manual was Paul's own copy, so don't rush out to order one from him; there are no more; although if I decide to pursue this particular machine any further I will certainly run a series of articles on machine code programming.

Useful items of information arising from 16 Bit Microprocessors are that the 9900, although a 16 bit chip, and according to the pundits, 16 bitters can address directly around a megabyte (1024K) of memory, cannot directly address more than 64K, which is the limit for 8 bit systems. The important word here is 'directly', as there are other methods of memory addressing which allow a much greater addressing capacity, and perhaps, with time, TI will offer say 128K or so, to match the competition. Unlike many other processors, the 9900 has few 'built-in' registers, relying instead on RAM for register space, which increases processing capability, and which might also explain why the 99's are so prone to hiccups. Confusingly, TI number their bits in the opposite direction from everybody else, although to be honest, I think their way is a little more logical: other systems are not really numbering bits, but using their 'significance', or the number that that position represents. All in all, not a recommended buy; there must be more on the market, surely ?

GUIDELINES FOR SOFTWARE CONTRIBUTORS : PROPOSALS

The guidelines are divided into three sections:

A. General; personal details, pre-loading requirements, equipment details.
B. Program; structure, use of REMs, function, content, etc.
C. Documentation; listings, flowcharts, algorithm(s), modification points, etc.

The guidelines are not intended as categorical DOs and DON'Ts, but are presented, together with the rationale behind them, as a rough guide for you to follow when submitting programs or subroutines so that alteration, in the event of debugging being necessary, should be made as easy as possible. There is only one definite, categorical requirement: the program should work! Some of the suggestions made here may be at odds with your own practices or beliefs: it is not intended that you should alter your own programming style, only that you try and follow this set of simple proposals to make life a little easier for the librarian.

A. General:

Personal details.

Please include your name(s) and full address. Include also the date of completion/submission of your program, and also state whether it is a copy of someone else's work, or an original item. It would be advantageous to have this information both in a REM at the beginning of the program listing, and in any documentation.

Pre-loading requirements.

If your routine requires the presence of additional items in either the GROM port (e.g., PRK or Stats modules, Extended Basic, Assembler/Editor, Mini-memory, Logo, Pascal, etc.) or the peripheral port (e.g., Speech Synthesiser, Thermal printer, etc.) BEFORE any program is loaded, then this should be stated somewhere on the cassette or disk on for example a label. This is particularly important as it can be very annoying spending five minutes loading a program only to discover that the Speech Editor should have been placed in the GROM port first, because of course to do such a thing after loading the program will cause a system reset, losing the program.

Equipment details.

Full details of the equipment which will be required to run the program should be given both in a REM and in any documentation. It can be very frustrating to load a program only to find that it will not accept keyboard entry and is designed solely for use with joysticks, which of course is the one item which you don't yet possess; or which requires a disk system to function although the original program was supplied on tape.

Additional information.

It would be particularly useful, and in the case of a submitted subroutine a virtual necessity, to be provided with a list of the variables used in the routine, and ideally the list would be divided into two sections, one dealing with variables which are used either in the main routine or in subroutines alone, and the other dealing with those which are assigned values by a routine/subroutine for use in a subroutine/routine. In the event that you wish to re-name some variables, such a list can save a great deal of time which would otherwise have been spent attempting to elicit that information from the listing itself.

B. Program:

Structure.

It would appear that the 99 operating system (which is involved in running your programs) will execute subroutines that much more quickly if the subroutines are placed as close to the beginning of the program as is possible. Upon encountering a GOSUB command, the system appears to begin searching for the designated line number (jump destination) from the first line of the program; for short programs the time taken to find this jump destination is not appreciably long, but for large programs there can be a quite significant saving in terms of execution time if the subroutines are placed very early in the listing. This practice necessitates two other structural changes: most importantly, a GOTO whose destination causes a jump over all the subroutines, and, of only slightly lesser importance, the occurrence of certain commands BEFORE that unconditional jump command. The certain commands include for example OPTION BASE, DIM, DEF, etc. If a subroutine contains a reference to a subscripted variable (array), and that variable is not DIMensioned until after the physical occurrence of it in the subroutine, it will be incorporated into the variable list with a default limit of 11 elements, during the first pass through the listing. When the DIM command is encountered, a NAME CONFLICT error will occur, and the program will not run. The same restriction applies to the use of DEF. In terms of program structure, the following outline will best show what is required:

```
100  OPTION BASE 1
110  DIM C$(12, 7, 5), N(180), V(20, 20)
120  DEF R4 = INT(RND * 4) + 1
130  GOTO 1000
140  subroutine 1
150  N(J) = V(T, K) * VAL(C$(B, C, D))
 -
 -
300  RETURN
310  subroutine 2
320  A = R4
 -
 -
780  RETURN
790  subroutine 3
 -
 -
 -
990  RETURN
1000 continuation of main routine
 -
 -
1100 GOSUB 140
1110 GOSUB 790
```

Structure (continued)

Generally, it is advantageous to place commonly-used segments of a program in a subroutine to reduce listing size only when the segment appears very frequently indeed, or is of such length that placing it in a subroutine will save considerable space. However, the 'modular' approach to programming favours the use of many subroutines, so that each 'module' may be tested and run independently of the others, making modification that much simpler.

Use of REMs.

The use of REMarks in program listings really dates back to the early days of programming when low-level languages and compilers were in almost exclusive use, and where it was possible to place enormous amounts of detail about the operation of the program within the source listing itself, for the compiler removed such comments while producing the object code. While not dismissing the function of REMs out of hand, their profligate use in programs which will run under an interpreter is undesireable, as such comments will take up remarkably large amounts of space. Although it is not suggested that your original listings should be devoid of REMs, a) your submission to the library should be accompanied by extensive documentation if a large number of REMs would otherwise be required, and b) the use of REMs should be limited to the first line, containing for example personal details, etc., and to special functions within the routine itself - for example, for the purpose of inactivating a line which may be required under certain circumstances but not under others. Such a use of REMs is demonstrated in the Babbling Brooks article on Plotting, Feb. 1982 issue of Tidings.

It is worth making the point that REMs should never be the destinations of conditional/unconditional jumps, as this can present severe problems if REMs are being deleted in order to create additional space for modifications, for example. Ideally, all REMs should appear on intermediate lines; this means that the program should be written without REMs, tested and debugged, then RESEQUENCED, and then any REMs for your own edification added on lines ending in 1, 2, ...9. This means also that entry of programs directly from the keyboard (from listings, for example) can use NUM without having to exit the mode in order to avoid REMs or to follow unevenly-spaced line numbers. This does not prevent your original program from consisting of unevenly-spaced line numbers, and twenty REMs to each program line; it is only suggested that you remove the REMs and tidy up the program for submission to the library.

Function.

The most obvious requirement for any program is that it should not only work, but that it should do what it is intended. In addition, it is very important that any instructions to the user (who may not know the first thing about computers) are clear and accurate. If your instruction is 'Press Any Key' then your program should respond on any key; the uninformed user may not realise that pressing 'Shift' was not part of your key range, and this will reflect badly (and rightly so) on the programmer. Bear in mind that using INPUT will require the user to understand a) that ENTER will need to be pressed after his entry, and b) that string characters cannot be entered when a numeric variable is to be assigned the input, nor can a comma be included in string input unless the entire string is placed within quotation marks. If your input requires two or more values to be entered with separating commas, make this clear. If scientific notation is likely to be used by the user, ensure that all the information necessary to permit error-free entry of data is given, especially with regard to the use of 'E'. The best test for any program is to give it to an individual who is totally computer-naïve; make notes about which areas cause problems, and bear these in mind when providing instructions.

Content.

Make the best use possible of the language in which you choose to write your program. Always RESEQUENCE the final version so that the listing is tidy and legible (before adding title REMs etc., see Use Of REMs). If you wish to scroll information on screen, don't use individual PRINT statements where a single statement with numerous colons will do. Use ON...GOTO and ON...GOSUB instead of multiple IF...THEN...ELSE statements where possible. Make use of functions like 4 * ATN(1) instead of 3.141.... for pi. (These suggestions apply almost exclusively to Basic; writers of Pascal, Logo, or other language programs may have different criteria to follow). It is also not a good idea to make alterations to your program without checking that no additional bugs have been created. It can be advantageous both for you and any user if, when awaiting entry of Y or N for example, use is made of CALL KEY instead of INPUT, especially in view of the number of programs which simply ask a question and expect the user to be psychic and know that either Y or N or YES or NO is the expected input. Make sure the response is correctly acted upon: one program which has been submitted asks the user if further attempts are wanted, and then ENDs when Y or YES is given in response.

Additional items.

It can sometimes be helpful if programs making use of peripherals like the thermal printer, speech synthesiser, RS232 interface etc., offer their use as an option; i.e., permit the user to decide whether speech output is wanted, or hard copy as opposed to screen printout, etc. This flexibility can enable 99 owners who do not possess Speech to use programs which have an optional speech capability, for example.

C. Documentation:

Listings.

If at all possible, provide a listing of the program which you are submitting. In the event that a disk or cassette will not load, a listing can, if short enough, enable the program to be entered from the keyboard. Listings also enable debugging to be carried out a little more easily, and they can highlight repetitive elements in your program which you might perhaps not otherwise have noticed. Some members are willing, for the cost of postage and materials, to provide printed listings for those members who do not possess a printer. Listings can also contain as many REMs as you wish, as they do not affect the size of the original REMless program; however, the line numbers of program statements must obviously correspond to the line numbers which appear in the printed listing. Handwritten listings are not really advisable unless your handwriting is particularly legible; typed or printed listings are best.

Flowcharts.

Ideally your program will be accompanied by extensive documentation, including flowcharts of the operation of the routines. Flowchart templates are available, albeit rather expensively, but these can ease the chore of flowcharting.

Algorithm(s).

Any algorithms which you have used in order to create your routine(s) should be described in detail, indicating how each has been implemented.

Modification points.

Some programs may be designed in such a way that the user could make specific alterations in order to customise sections to his own particular requirements. Details should be given with regard to how this is to be achieved, together with the limitations.

Additional information.

It can be helpful if contributors assume that the recipients of their work are 'computer-illiterate', and must therefore explain every detail of operation of their program/subroutine. In this way, authors can gain a deeper understanding of their work, and can also educate less-experienced programmers, providing them with numerous ideas and examples which they may find of benefit in their own programs.

It should be emphasised that submissions which do not provide any of these basic items of information will still be accepted, but the lack of information in the event of faults will reflect upon the author. Equally, the lack of faults, or the ease with which faults can be rectified as a result of information supplied, will also reflect upon the author!

A Final Word

Since the production and submission of this article, TI have been snipping away at their prices. It seems likely that by the time you get to read this, the 4A console will be at an RRP of around £250, with sufficient leeway for some dealers to be able to offer it at around £200. Some module prices have also come in for a little pruning: the TI Invaders module drops from almost £40 to under £20, or so I am led to believe. It will be interesting, for me at least, to see if the Mini-memory module too comes in for a little scalpel-work - around £55 - £60 would be nice, if you can manage it TI.

What I want to know is: why weren't they sold at those prices in the first place ? I've heard the one about 'They won't sell if you make them too cheap'; it just makes me wonder why everyone doesn't flock to Harrods and ignore the cut-price superstores...

Something else has also been brought to my attention, only it isn't as pleasing as the above. It seems that when blind or partially-sighted prospective TI owners have asked TI whether the Terminal Emulator II and the Speech Synthesiser will enable listings to be spoken aloud, somebody at TI has been saying 'Yes'. The answer should have been an honest 'No', unless TI have been keeping secret another set of commands not listed in the manuals. I don't know about the feasibility, but I could envisage a program running under Extended Basic, employing the RAM Expansion and the Speech Synthesiser and using machine code, which would enable a feeble attempt to be made (the speech facility on the TEII is superior to that on either Extended Basic or the Speech Editor) to read listings, but otherwise the answer must be 'NO'. If TI know otherwise I sincerely hope they will tell all as soon as possible. One former TIHOME member bought his machine almost entirely on this basis, and that is why he is now a former member. He is also a poorer, wiser, former member.

From time to time, odd things are pointed out to me. One odd thing is that, until just recently at least, one mail order firm were offering a TI for sale, apparently a PAL 99/4, for around the £450 mark, (although the photo rather indistinctly suggested it was a 4A), which just goes to show that there's hope of finding a Dodo yet.

Good programming,

Pete Brooks.

2021 update: To have the computer LIST a program verbally, with TE2 and speech synthesiser, type LIST "SPEECH".

QUOTATIONS FROM THE QUESTIONAIRE.

Question 1.

No.

Question 2.

Yes, with certain qualifications.

Question 3.

Yes.

Question 4.

Updates on software available.
What about the 99'er magazine.
Translations from other dialects of Basic.
Update Software catalogue.

Question 5.

Price of peripherals & modules.
Total lack of technical knowledge.
T. I. are bad at marketing.
Lack of software.

Question 6.

??
Haven't tried yet.

Question 7.

Yes.

Question 8.

A small program each week.
Regular articles on assembler programming.
Improve reproduction quality of Tidings.
Ask BB to use a better ribbon in his typewriter.
How about a cheap RS232C interface.
How about a 32K D.I.Y. memory expansion.
Short programs in general.

Question 9.

Only availability.
Rotten joysticks !!!!
Price.
How about a ROM disassembler.
Failure to read cassettes.

Question 10.

Education for children.
I am very pleased.
I thought it would provide entertainment.....
To teach myself basic programming.

Just a few comments! All your comments will be processed and the results Will greatly influence the format of the next issue of Tidings. For those who lost the last page of Tidings because it was printed on the back of the questionaire, I enclose another one. Also another Questionaire for those that didn't have the opportunity of completing one.

Paul Dicks MIDPM

Random dots

THE GAMES SCENE

With the announcement of new games modules nearly every week, we are now being spoilt for choice on the games side. Our favourites at the moment are Attack, TI Invaders, and (still) good old Pinball (From Video Games I). By the way, my copy of VG I is not even recognised by the 99/4A - if you plug it in you don't even get it listed on the initial menu! So 99/4A Owners beware - it may be that VG I is not compatible, although it is still on sale.

Top scorer on Attack is my wife, with 960250 on Stardate 97!! Can anyone beat that, and how far do the Stardates go?

Everyone who plays TI Invaders agrees that it's the tops. The fast action and true high-definition graphics on this module (due, I am told, to it being written in Assembler rather than the usual GPL) set that standards high for future games.

Another point which applies to anyone using a black-and-white monitor or TV is that some of the games modules (Pre-School Learning Fun, for instance) have colour combinations which are unreadable in black-and-white. Again it seems to be a case of trying out any modules with YOUR set-up before you buy.

By the way, Mr KA Wilson of Doncaster, about the price of software, the signs are that it is already starting to fall.

If you shop around, you can get quite significant differences in TI software at the moment - For example the Attack at Dixons is £39.95 while at Boots it's £29.95. Since the arrival of the Sinclair Spectrum TI have reduced the price of the 99/4A console to less that £200, but, so far there is no corresponding drop in the price of other hardware.

COLOUR COMBOS

If you have tried any of the more unusual colour combinations on your screen you will by now have noticed that the recommended combos in the Handbook can be taken with a pinch of salt and that some of the fore and backgrounds are useless especially with text.

Combinations which are good on MY set-up (NTSC 99/4 and Skantic Monitor) are no good on a 99/4A and PAL colour TV.

If you are in any doubt and wish to check which are the best colour combinations for YOUR outfit, then run the following short program with a notepad in your hand

```
100 CALL CLEAR
110 FOR FOREGROUND=2 TO 16
120 FOR BACKGROUND=2 TO 16
130 FOR GROUPS=1 TO 9
140 CALL COLOR(GROUPS,FOREGROUND
,BACKGROUND)
150 NEXT GROUPS
160 CALL SCREEN(BACKGROUND)
170 PRINT "THIS IS COLOUR NUMBER
 ";FOREGROUND
180 PRINT "    ON COLOUR NUMBER "
;BACKGROUND: : : : : :
190 PRINT "WRITTEN AS: ";
200 PRINT "CALL COLOR("&STR$(FOR
EGROUND)&","&STR$(BACKGROUND)&")
"
210 FOR PAUSE=1 TO 600
220 NEXT PAUSE
230 CALL CLEAR
240 NEXT BACKGROUND
250 FOR PAUSE=1 TO 600
260 NEXT PAUSE
270 NEXT FOREGROUND
```

I have written the variables out in full to make them understandable, but if any beginners are still a bit baffled by this little routine, especially Line 200, then drop me a line. This applies if you take issue with any of the other drivel in "Random Dots". My address is:

130 StapleFord Lane
Beeston
Nottingham NG9 6GB

I will print any interesting queries and replies in the next issue of TIdings

NEW 99/4 APPLICATIONS

My BLIND friend, LES BALL has just acquired a 99/4A (for under £200!!!) along with a Terminal Emulator II (for the Text-to-Speech facility) and we are both now busy writing programs for his use in both his job (he's a Social Worker) and his hobby of ELECTRONICS!! I will make any of the resulting programs available to anyone interested, later. I know that TIhome has at least one other blind member. I find the TE II Module fascinating as it will speak ANY English words, and quite a few not-so-English!! It even LISTs programs as speech.

(Notes for New Members. "Random Dots" is written on the 99/4 with a TI Thermal Printer attached and using a Word Processor program and Character Set of my own design. The program is available from the TIHOME Software Library - Serial No 403)

MORE ON EXBAS

In the beginning there was the TI 99/4 and TI Basic. Then along came the 99/4A and Extended Basic, closely followed by Paul Karis's revelations on the PRK Module and subsequently TI's rather reluctant publication of the full story on what is now being called Enhanced Basic (involving the PRK or Statistics Module). For a simple soul like me the picture is getting very complicated, and I have been in on it almost from the beginning (being one of the "Founder members" of TIhome.

Now I have my Exbas module, and I am getting used to the superior range of facilities, I am tempted to re-write all my programs in Exbas. The problem is - how many other members have Extended Basic, as I don't just write programs for myself. Indeed, we don't even know how many own the PRK/Statistics Modules, so the same thing applies to programs written in Enhanced Basic.

Anyway, for the few? who either own or are trying to get hold of Exbas, here are a few snippets of info. which you won't find in the manuals.

Exbas recognises AND and OR as Logic Operators and also the TI Basic equivalents - "*" and "+" Used with the Speech Synthesizer, Exbas does not say "UHOH" to words which are not in the Vocab. - it SPELLS THEM OUT!

The Thermal Printer quirk, which "garbles" line numbers occasionally into unrecognisable random graphics does NOT occur when programs are TP-listed under Exbas, so the moral is to use Exbas for printed listings of ALL programs.
A TI Basic program which contains double-colons used to print blank lines will cause Exbas to halt execution, although MORE than two colons used for the same purpose (in TI Basic) will be automatically spaced out by Exbas.
I am writing these pages using a new version of my Word Processor in Exbas which is much faster and more flexible than the former version in TIbas. Perhaps the biggest change is being able to use the LINPUT Statement instead of INPUT. This means that the lines of data can include commas, quotes, and even spaces as valid input making the former character re-allocations (for the comma and quotes) and the "indent" control character - > - redundent.

WHAT'S IN A NAME?

When TIHOME first started I suggested that we call the newsletter "TIdings". I am now pleased to hear that the GEBRUIKERS-GROEP TI 99/4 NEDERLAND (our Dutch counterpart) have taken up the name "TIJDINGEN" for THEIR newsletter and it is even more relevant in Dutch than English.
TIJDINGEN means not only "tidings", but split-up into TI and Dingen (things) makes up "TI Things".
The Dutch User's Group have also not only translated my Word Processor into Dutch, but they ACTUALLY USE IT!!

MUSIC & the 99/4

As a semi-professional musician (organist) I have a continuing interest in the music facility of the 99/4. One limitation is the lack of a FOURTH voice (available on other micros), which restricts the harmony side of things quite a bit. I have recently obtained a Music Maker Module and I am at present working out the best way of using it. Again there are some pros and cons over the use of music in programs, For instance there is no facility for TRIPLETS, TRILLS, or time-signatures such as 12/8 or 5/4 (where the numerator>denominator I would have like to see the ability to add white noise in the Traditional Mode as well as in the Sound Graphs.

Writing music in TIbas is a lengthy and memory gobbling procedure, because of the need to constantly repeat CALL SOUND(D,F,V) for each note. Those of you who have tried to write decent three-part harmony will realize that it takes quite a bit of musical ability and that it is even more difficult to make some notes sustain whilst others (of shorter duration) play simultaneously, if you know what I mean. There are ways of overcoming this, but none is easy and all take up even more memory. I have therefore written a new (and much shorter) routine in Exbas. If anyone is interested, write to me at the above address.
It might even be a good idea to start a Music Page within TIdings!!

Mike O'Regan

```
**********************************************
**********************************************
***                                        ***
***   AN EPISTLE FROM TEXAS INSTRUMENTS    ***
***                                        ***
**********************************************
**********************************************
```

At long last I have, after many prompts from a certain Mr Paul Dicks, found the time to write my bit for this excellent TIHOME. Let me start by giving you Texas Instruments' feelings for TIHOME.

We think that TIHOME is an excellent organisation (we are members too!) that fulfills a valuable function for owners of TI-99/4 and TI-99/4A computers. Your chairman Paul Dicks does a superb job in organising and running TIHOME and I would like to take this opportunity of thanking him on behalf of Texas Instrument, and encouraging you the Members of TIHOME to give all the support you can to Paul.

Now I have finished buttering up Mr Dicks, lets get on with some news. Many of you may have seen adverts in the paper showing a new low price for the TI-99/4A, well I am pleased to announce that due to increased sales both in the UK and in Europe, we have been able to speed up our cost reduction programme and the computer is now being sold for around £200. This will cause an influx of members to TIHOME. (Now you can see why I have been saying nice things to Paul).

We have also been able to reduce the price of some software modules for instance TI-Invaders from £39.95 to £19.95, Tombstone City from £39.95 to £24.95, Chess from £49.95 to £39.95, Number Magic from £17.95 to £14.95 and Speech Editor from £34.95 to £19.95. There are many other price changes, so I suggest you consult your dealer for further information.

Becoming available in August will be the new Peripheral Expansion System. This will get away from the 8 foot long monster you would have with the old style peripherals. The unit is 14x15x20 inches and has eight slots for peripheral board-type cards, one of which is used to interface the system with the computer. The owner simply plugs in additional peripheral cards to add desired functions and there is also room in the unit to house one Disk Drive. Two further Disk Drives can be added externally but all this means is that you can end up with complete computer system in a neat attractive package.

Also coming shortly is Editor/Assembler, this kit contains a Command Module, a Diskette and a 400 page Manual, (all for £89.95), and allows you to program in TMS9900 Assembly language. You can now write your own version of Invaders and to give you an idea of how fast and capable this language is, a little story follows.

TI Invader was written by a college student during his 6 months in Industry, in our plant at Lubbock in Texas. He wrote Invaders in a couple of weeks, but unfortunately having selected the desired level of difficulty the game was over before anyone could pick up the joysticks! The poor student then had to spend another month slowing the program down. This neatly brings me to MINIMEMORY, Minimemory is a cut down version of Editor/Assembler requiring no peripheral other than a cassette recorder. Consequently Minimemory allows you to write line by line Assembly Language Programs and store them directly in the Command Module. You can do this because we have put 4K Bytes of User Memory (RAM) into the module. This RAM has battery back-up so programs are saved even when the module is removed. You can also use this 4K of RAM to store ordinary Basic programs and data, since the computer treats it as Disk Memory. The module costing around £89.95 and has more available RAM in it than a standard Commodore VIC 20.

Texas Instruments will be at the Personal Computer World Show at the Barbican Centre in London during the beginning of September. All the latest kit will be there and I would be pleased to meet any of the members of TIHOME. Our stand number is 220/224 in Hall A Upper.

We are just starting a campaign to increase the number of 3rd Party software packages. To achieve this we will be writing to all Software House's currently involved in the Home Computer business and if any member of TIHOME is interested in writing software and marketing and selling it, and would like further information, please contact me. I do stress that this is for the serious author since we will review any programs sent to us and those of an acceptable standard will be published, along with information of where to purchase them in our new brochure.

I am asked about suitable cassette recorders from time to time so here is a list of those that we have had no problems with. If you know of others please let TIHOME have their names.

SANYO	Slimline One
SANYO	Slimline Five
PYE	9110
LLOYD	U182
FERGUSON	3T15
HITACHI	TRQ 299
CROWN	Automatic CRO2

....Cont/d

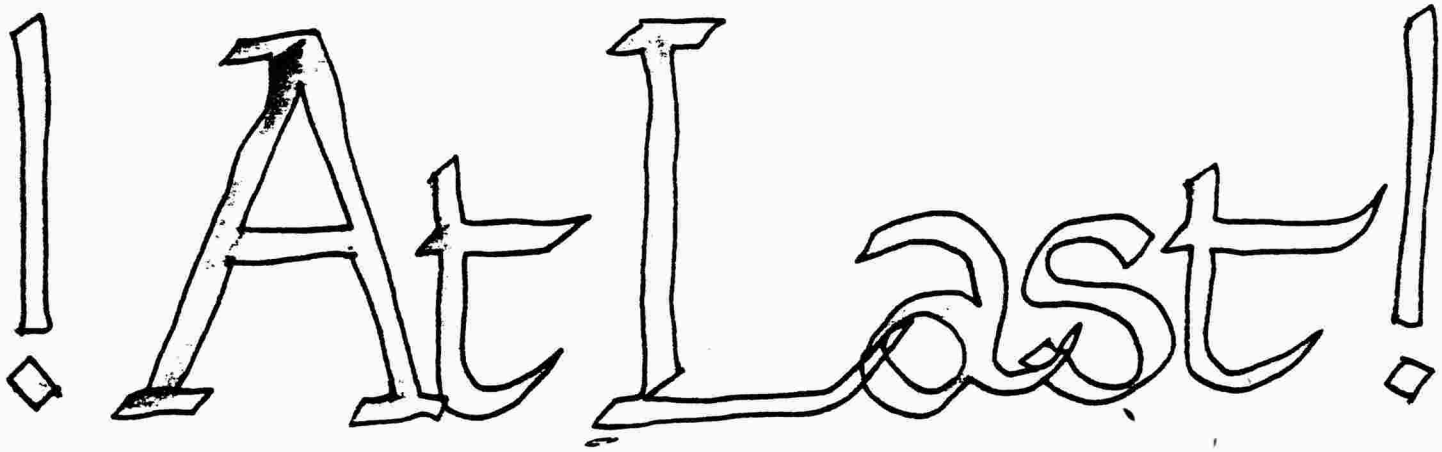
As you are aware, we made changes between the TI-99/4 and TI-99/4A. The major impact to the user was the keyboard and location of special functions, so for your information here are the differences.

NAME	TI-99/4	TI-99/4A
AID	Shift A	FCTN 7
CLEAR	Shift C	FCTN 4
DELETE	Shift F	FCTN 1
INSERT	Shift G	FCTN 2
QUIT	Shift Q	FCTN =
REDO	Shift R	FCTN 8
ERASE	Shift T	FCTN 3
PROC'D	Shift V	FCTN 6
BEGIN	Shift W	FCTN 5
BACK	Shift Z	FCTN 9
CURSOR CONTROLS	SHIFT (DIRECTION)	FCTN (DIRECTION)

I hope this has been of some interest to you and I will do my best to ensure you the members of TIHOME are kept up-to-date. If you have any queries or questions please send them to Paul who can publish them along with the answers from me.

Thanks for your attention and Happy Computing.

ROBIN FROWD
Product Manager
TEXAS INSTRUMENTS LTD



A

INTRODUCTION to April & May Rambles by Stephen Shaw

TI HOME - June 1982.
(or July or Aug—whatever! Vol2 Issue3)

Now that TI have launched what promises to be the final format of their Home Computer, with an impressive list of household names as distributors,we hope that this issue of TI HOME will be reaching a lot of new purchasers for the first time.

If you find the style/content of TI HOME unsatisfactory, please would you do two things:
1. Write and tell us.
2. Consider submitting your own article(s) - anything from a paragraph to a whole sheaf of paper!

My own section consists very largely of small items,written as they are discovered/occur. There is not the space to fully expand and explain every item! You are invited rather to consider these notes as a starter to your own deliberations and experimentations.

If there is anything which is causing you a problem in using your Home Computer, or if you feel you are not making the best use of it, drop me a line, and I will try to respond in TI HOME (press dates mean a likely delay of 3 months).

If you are really stuck- please send a large SAE. I dont have very much spare time but will try to assist.

I can offer a variety of services and software. The prices are as low as possible, but there are costs to cover, the pricing structure acts as a rationing system (of my time!), and my feeling is that our Computer is best served by encouraging true commercial involvement - establishing prices at too low a level will tend to drive this away,or encourage sloppy programming. I am not looking -nor expect! - a high income.

The very best way to learn about your computer is to use it and to experiment with it. Buy a few programs and LIST them- try to see how they work - can they be improved?

If there is any particular item in the following pages which you would like expanded or better explained - tell me! Subject to time - and space - permitting, I'll do what I can.

I strongly reccommend to you the magazine 99er - an airmail subscription direct from the States is US$43 for 6 issues at the time of writing. There is a lot to be learned from this magazine.

Stephen Shaw
10 Alstone Road STOCKPORT Cheshire SK4 5AH

---to reduce postage and help conserve trees,one or two of my pages this issue are reduced & printed sideways. Hope this does not give you eyestrain.

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

ERROR in CORRECTIONS TO EXTENDED BASIC MANUAL (Version 110):

Page 11- quote:
"Any file name listed in part of in whole by lower*case letters is not accessible by the TI99/4 Computer. Only the TI-99/4A Computer can access a program named or called in lower case letters".

I have an old 99/4 computer.

Here is a catalog of one of my disks. Note the last two items. Saved by me.

I can use them as well.....

((Put the string together using concatenation and CHR$(n)))

(* The 99/4 did not have lower case)

NB: DIS/FIX 0 means the program with this against it is flawed, either due to a fault while saving, or lack of space on the disk. The program cannot be read from disk.

CATALOG DISK

DSK1 - DISKNAME= SPARE-DISK
AVAILABLE= 0 USED= 358

FILENAME	SIZE	TYPE
BIO	51	PROGRAM
BIO-B	52	INT/VAR254
BLKJK-TIB	24	DIS/FIX 0
BOXES	40	PROGRAM
LOAD	6	PROGRAM
RESCUE	36	PROGRAM
SCRIBBLE	50	PROGRAM
SCRIBBLE-B	50	PROGRAM
XWORD-XB	45	PROGRAM
test	2	DIS/FIX100
test_2	2	INT/FIX100

You cannot SAVE a PROGRAM using a lower case name with the 99/4.

I am now authorised to distribute programs from PRP Computer Graphics of the USA. The sales flier is enclosed, together with my reviews of the programs offered.

After four days of continually trying to contact TI EDINBURGH I have come to the conclusion they have given up answering the phone for Lent. TI SLOUGH answered VERY promptly and were extremely helpful. Is there anyone alive up there in Edinburgh????? You are giving TI a wonderful reputation.

TI's new launch of the 99/4A clearly emphasises their approach to this product, moving the home computer away from specialist shops to consumer shops- principally BOOTS, WILDINGS, & DIXONS. HMV was a surprise, and possibly London ARGOS is an experiment?
This really leaves TI HOME to assist purchasers. Please would you consider if YOU could write an article or two, perhaps aimed at new owners ?

2021- MOST TI listed dealers never had any stock and probably sold none.
Waste of adverts and many lost sales.

(1)

titititititititititititititititititititititititititititititititi

READER ENQUIRY---where can I buy PROGRAMMING BASIC WITH THE TI HOME COMPUTER by Herbert D Peckham ? (1979 p/b)
A: This book is published by MCGRAW HILL- you are unlikely to find it on open sale, but any good book shop should be able to supply to order. (In Stock in Manchester May 10th)
In case of difficulty the publishers have a UK address:::
Marketing Services Dept, McGraw-Hill Book Co (UK) Ltd., FREEPOST, Maidenhead, Berks, SL6 2QL. ABOUT £11-25

titititititititititititititititititititititititititititititititi

PEEK & POKE : You do not need to buy extended basic and the expansion memory. These are available (as PEEK & LOAD) in TI BASIC when you use the MINI MEMORY MODULE which also gives you 4k memory and the capability of writing and running Assembly progs.
Price in the States is about US$100 so expect a UK price (should be available when you get this) of £80 or so -same as Extended Basic. — Actually £115 !!
COMPUTER CONTACT (22 Birchall Rd Rushden Northants) offer 10% off modules. Ask them their price/stock position on 09334 55673/56894. £104?

Extended Basic and Expansion Memory are still useful if you can afford them.

+

Suggestion: An extra page in TI HOME listing recent additions to the program library and significantly amended programs. ???

@&@&@&@&@&@&@&@&@&---graphiks?@&@&@&@&@&@&@&@&@&@&@&@&@&@&

Mike O'Reagan & myself - both with 99/4's - are having a little difficulty in booting up Extended Basic - anyone else with problems ? If you have a 99/4A & have NO difficulties- please report in! Is this something for TI to deal with? or a case of so many contacts one is bound to be dirty??

"LE STIC"--- has any member managed to interface their 99/4(a) with this super 'joystick' yet? Please report in - are you prepared to sell to club members?

MPI

Back in the February issue I mentioned that I was having a problem obtaining a refund from the above company, who distribute 99er Magazine in this country.
In fairness to the Company, I wish to now report that they did in fact give me a refund . In response to a letter written 21.12.81, my credit card account was credited by them on 25.2.82. I received no advice from them, and have just received my credit card statement.
I therefore retract my original suggestion that they may have been dishonest, but I still cannot reccommend them.

×

(((co-incidende - my refund from MPI was made by them at about the time TI had the February issue of TI Home???)))

×

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

I was sorry to see the price in the UK of the Assembler/Editor module- from £104 to £115.

In the USA the same module retails for from $80 to $90.

Assuming an Exchange rate of 1.65, postage of $12, price $90, and VAT at 15%, DUTY at 17% - this comes to...em ... er...
£81.60

Bear in mind i) the US price quoted is RETAIL and already includes a mark up for the retailer
ii) Import more than one module and the post per module averages downwards.

So- why such a high UK price ?

To conform to UK guarantee logistlation? I think not- some US states have just as tough legislation.
To debug the programs? You mean TI INC sell lousy programs to their US customers...?
To give uk retailers a higher mark up? To pay for lower sales due to ~~their~~ higher prices....

Where is it going please?

BOOTS and DIXONS in Manchester now have the 99/4A- £315 and £330 resp.
BOOTS console available to the public but not turned on- the assistants appear to have had no training, and apparently frightened by the multiplicity of keys-
SHIFT/FUNCTION/ALPHA LOCK/CONTROL/ENTER
-were trying very hard to enter a short program into a VIC20. The book contained the gem FOR J=1 to 1000 ; NEXT N. Wouldn't work would it!

DIXONS have everything in a glass case behind a counter. Running -off & on- a new demo module which is better than the old one. Dixons also sell the VIC20- console available to the public.

I have a feeling TI could hope for better support from their retailers than this! No peripherals visible, but BOOTS had the games joysticks, and a box marked 'speech synthesiser'.

I dropped a line to the respective managers but as the stores are so big dont expect a reply.

My current gem is a program which rewrites about 70% of itself- the 'etch a sketch' routine from Feb TI HOME which saves the picture back into the program. Run again and your picture appears on screen VERY quickly. I'm offering the prog to 99er, and may be able to print it in the next issue of TI HOME. ((nb- it requires the expans memory)).

MODULE PROBLEMS ?

From time to time you may have problems getting one or more modules to 'boot' properly.

If this applies to ALL your modules, the problem is probably in the connector in the console - it may need replacing or repairing.

If some modules work OK, the problem is probably in the modules. The modules connect to the console by means of PCB tracks going to a PCB connector. The tracks are on BOTH sides on the PCB.
Inevitably, corrosion, grease, tar residues(if yousmoke) etc will build up on the PCB. Although the contacts in the console do carry out some 'cleaning' work on insertion, you may have a poor electrical contact on one or more tracks.
Look at a module - at the back is a thin slit, surrounded by a movable plastic mask. This can be pushed downwards into the module to reveal the PCB inside.
CARE DO NOT touch the PCB contacts. You will almost certainly destroy the module!!!

Push the shield down from the sides,well clear of the PCB. In a strong light look at the PCB contacts (both sides) - any signs of dirt or corrosion? Wear?

If they require cleaning probably the safest means is to use those cotton-swabs-on-a-stick together with tape recorder head cleaning fluid. NB:Do not be too enthusiastic with the fluid and try to avoid getting it on the module casing.

****

Do you have a RUBIK CUBE or cheap copy thereof?
DONT CHEW IT! Lead levels are ten times the maximum permitted in paint...
Have you solved all the Cube problems, and do you regularly solve it in under thirty seconds ?

OK. Heres a REALLY hard one for you - a cube with 4 cubelets on a side (4x4x4). The mechanism is courtesy David Collins and Daniel Smith, and IDEAL should have them on the market very soon...
IDEAL have again got their sums wrong. They state 3.63E50 combinations, forgetting different side orientations and lack of a 'middle' cubelet - there are 3.7E45 combinations.

Where does the 99/4(A) come into this ? You can buy the 4x4x4 as a 99/4(A) program! Imported from the USA and selling at about £9.Called 'Quadcube'. £9 from Work Force,140 Wilsden Ave,LUTON,Beds.

If you like manipulative puzzles, I can reccommend CUBIC CIRCULAR a mere 16pp published 4 times a year by David Singmaster for £2.50 pa
(David Singmaster Ltd 66 Mount View Road
LONDON N4 4JR)
It contains some interesting puzzles-mostly suitable for computer simulation.
((*copy sent to Paul Dicks for reference))

News from Radio Nederland (broadcast of 22.4.82) of a universal computer coding which would permit:
15 different computers to talk to each other
Program swops between 15 brands of computer without rewriting the program!
High baud rate for saving programs- 1200 baud
Lower rate available for short wave or telephone use.
A single byte lost in loading will not cause loading to halt- merely put a syntax error into the program.

The 'hobbyscoop code' has worked at 1200 baud in Holland via FM radio on the TRS80 and PET, but will never work on the ZX81.
The code may become available for use on the TI99/4(A) - some hardware MAY be required. Code is US$8, if hardware is required, the maximum is US$30 (plus UK Duty!).

As at transmission date they had not fully debugged the 300 baud version, and the English handbook had not been printed.
I have asked our Dutch colleagues for news when available.
→ IN DEVELOPMENT (11.5.82)
Presumably programs are restricted to ANSI BASIC but no news was given on this.

£££ £££ £££ £££ £££ £££ £££

TI SOFTWARE: MARKET A simulation of business competition. Available on cassette or disk.

If you have this program and find it difficult to find anyone to play with you, I have amended mine to run for one player, the computer taking the other part.
As supplies appear to be limited in this country I wont give the details here, but send me an SAE if you are interested.

& & & & & & & & & &

My friends at Xerox have told me to fit a new typewriter ribbon- hope this assists your eyesight!

Points to ponder:
1. It is very unlikely that sales of the 99/4A will ever come near the half million sinclairs - with a smaller base to spread development costs over, (better development ?), prices will always appear higher - especially on software with the high risk of rampant piracy.
2. Looking at the computer mags - lots of progs for the sinclair but not many for the others. The BBC micro had an unfair marketing advantage (free ads on BBCtv, endorsement by BBc) which guaranteed thousands of sales regardless of quality - I still think the TI is the best HOME computer.

"""""""""" """""""""" """""""""" """""""""

Have you seen the Saturday morning TV game- Powwww! ? A loud sound fires the gun in this video game. It could be done with the 99/4A, a sound triggering the fire button on the remote control socket- required- mike,op-amp,opto-isolator. Just an idea - if anyone is a deft hand with a soldering iron they may like to try it. The pin-outs for the remote socket have been published in TI HOME-send me an SAE for a copy.
Concentrating only on the joystick,and operating the fire control by sound could make for interesting play!
Let us know if you manage it.

I have just taken delivery of the Tombstome game- review herewith- and yet again with a new module, I had to insert/withdraw it a few times to clean the contacts before the game would load OK.
Attention New Owners: Do not be discouraged if a new module is a little hesitant - sitting on a shop shelf (or in a drawer or on a desk) without use, the contacts in the module gather a thin non-conducting film. This is removed by repeated insertion/withdrawl.

::::: ::::: ::::: ::::: ::::: :::::

An advert appearing widely in the States, placed by Commodore, compares the VIC 20 with other computers- this is what they say about the 99/4A:

Maximum RAM = 16k	Keyboard= Half typewriter size
No function keys programmable	
Only 64 displayable characters	
Microprocessor TI990	No accessible machine language
No upper/lower case characters	No microsoft basic

None of these claims is completely true. The add does dpecify the A model.
I hope TI takes action on this one- these ads are not fair or true, and could not appear in UK magazines.

Just as you can amass any score you wish at Space Invaders, if you follow a few simple procedures (they cant bomb you when they are on the bottom line!) so, in Tombstone City there is a simple way of scoring anything you want (at Level 1 anyway.Much harder at other levels) - but I wont spoil it for you. See if you can discover it- . if you do, write & tell TI HOME about it!

In using this module (Tombstone) which has the full 44 pin pin-out, with all the problems I've come to expect of this full usage (on the older 99/4 anyway), I've come across two new 'error messages' not mentioned in my manual - including one Pete Brooks found a while back.
When you switch on,you have a 'test card' -press a key & you have the TI logo, the words 'Texas Instruments Home Computer'
and then,usually, a menu to select from eg 1. TI BASIC

I came across an additional item- '4 FOR REVIEW MODULE LIBRARY' - this is a description of this very page(module library)- I pressed four,and the screen now had only two items,not four! The module contents were no longer available.

The other message I was given, below the logo & title, but without the 'PRESS.... etc' was "INSERT CARTRIDGE" (note word- not module).

A quick bit of thinking suggests that these two error messages occur when the computer
a) has identified the cartridge but cannot use it, and the other message:
b) knows there's a cartridge there but cannot identify it-

eg due to poor insertion, dirty contacts &c. Sometimes this does not just cause a lock out, but the friendly CPU tells you its troubles!

T.I. HOME JUNE 82

TOKENISATION

or How the 99/4(A) Stores Programs....

When you enter or load a program into your Home Computer you do not need to know where in memory it is stored, or how it is stored. It is sufficient if your program RUNs OK!

However, if you do know, you can use your knowledge to write programs which use less memory - eg get more out of your '16k'.

If you have Extended Basic and the Expansion Memory, it is fairly easy to obtain these details (use call peek from 0 to -24k in -1 steps) - but if you have only the console?

With only the console, only one area of memory is used for everything - string and numeric variables, character definitions, and your program -etc etc.

Your program is stored in memory in two parts- the line numbers, and in a separate section the line contents.

Thus the Computer needs an index -and has one- to see where the line contents are! This is why a program line takes up 6 bytes before you put anything into it.

The index uses four bytes- two bytes for the line number and two bytes for the memory location of its contents.

The length of the program line (in bytes) takes up one byte, and an end of line separator takes up another byte.

Thats 6 bytes and nothing there yet!

If you have a line ie 100 REM

How many bytes is that going to take up?

The 99/4(A) uses a SINGLE BYTE for BASIC commands-such as REM. So this sample line will take up 7 bytes of memory.

If the sample is 100 GOTO 120
How many bytes?
GOTO as we have seen uses ONE byte
(but if you use GO TO which is two command words, that uses two bytes)
The LINE NUMBER 120 takes up 3 bytes- the first a marker to say 'this is a line number' and the next two the actual number (if the first byte is A, the 2nd is B, the line number will be A*256+B)
So this line uses up 10 bytes.

If the sample is 100 B=C
then B, =, and C each take up one byte, making a line total of 9 bytes.

However 100 B=3456
will take up 14 bytes - B & = use 1 byte,
the number 3456 takes up 6 bytes-four for 3456, and one to indicate 'number', the other byte says how many characters are in the number.

HINT: If memory is tight, replace all number)sed a few times by numeric variables.

String variables are dealt with in a similar manner- one byte says 'this is a string', another 'this is string length' if you use a quoted string - if you use a string variable, only the characters in your listing take up space-
eg "TEST" is 6 bytes .
TST$ is 4 bytes.

The CALL routines use what are called UNQUOTED STRINGS- that is the word following CALL is a string, but without quotes. You cannot therefore use CALL A$ - 'cos A$ has quotes.

CALL COLOR uses one byte for CALL and 7 bytes for COLOR- two bytes say 'unquoted string' and 'length of string'. Using a lot of CALLs can eat into your memory- well worth using GOSUBS if you can.
(100 GOSUB 140 = 10bytes -remember!)
(150 RETURN = 7 bytes.)

The 'tokens' which take up one byte of memory to indicate these commands and so on, use numeric values 129 to 254- but not all are used. The ones used -are listed on the next page- note the same codes are used for Extended Basic as for TI BASIC, but some codes cannot be translated by TI BASIC, and will appear on your screen as funny characters.

PRINT : takes up two bytes in both languages, but
PRINT :: also takes up two bytes in Extended Basic, as :: is covered by a one byte token.

The codes for !@P- and !@P+ are not known at present (used in Extended Basic Version 110).

When you use a NUMERIC variable, the value resides in its own memory location - and uses 8 bytes . If you keep a careful watch on your numeric variables, and dont use more than you really need, you can save worthwhile memory space.

Similarly, if you use DIM, 8 bytes are set aside for each numeric value eg DIM B(7,8) sets aside 7*8*8 bytes if you use option base 1,
or 8*9*8 bytes if you dont.
If you use a two dimensioned subscript without using DIM, because neither of the values exceeds ten, in option base 0 the computer will set aside 11*11*8 bytes.
That is a lot of memory gone west if you are not using it!

HINT: Always DIM numeric arrays, use OPTION BASE 1 if you are not going to use 0 in the subscript.

String arrays are not so heavy on memory, a more two bytes being set aside for each possible subscript. (replace 8 with 2 in above examples).

Incidentally- using the same block of memory for everything CAN cause a system lock out if you RUN-EDIT-RUN-EDIT - the lockout occurring much earlier if your program is a long one.

Have you noticed your computer pausing for a second every now & then? When you redefine a variable, the old value remains in memory, taking up space. When you have defined enough times, there is no more memory- and the 'rubbish' is kicked out. The 99/4A is remarkably efficient at this process, but it still takes a little time. The old definitions are not kicked out immediately, to give a faster running program.

Any que)s - send them in!

LIST OF NUMERIC VALUES used to denote BASIC commands &c..

These are returned with CALL PEEK when you peek the memory location which holds your program.

129	ELSE	130	::	131	! (tail rem)			132	IF		
133	GO	134	GOTO	135	GOSUB	136	RETURN	137	DEF		
138	DIM	139	END	140	FOR	141	LET	142	BREAK		
143	UNBREAK	144	TRACE	145	UNTRACE	146	INPUT	147	DATA		
148	RESTORE	149	RANDOMIZE			150	NEXT	151	READ		
152	STOP	153	DELETE			154	REM	155	ON		
156	PRINT	157	CALL	158	OPTION	159	OPEN	160	CLOSE		
161	SUB	162	DISPLAY			163	IMAGE	164	ACCEPT		
165	ERROR	166	WARNING			167	SUBEXIT	168	SUBEND		
169	RUN	170	LINPUT	171	-175 not used						
176	THEN	177	TO	178	STEP	179	,	180	;		
181	:	182	)	183	(	184	&	185	not used		
186	OR	187	AND	188	XOR	189	NOT	190	=		
191	<	192	>	193	+	194	-	195	*		
196	/	197	^	198	not used						
199	quoted string			200	unquoted string (or number)			201	Line number		
202	EOF	203	ABS	204	ATN	205	COS	206	EXP		
207	INT	208	LOG	209	SGN	210	SIN	211	SQR		
212	TAN	213	LEN	214	CHR$	215	RND	216	SEG$		
217	POS	218	VAL	219	STR$	220	ASC	221	PI		
222	REC	223	MAX	224	MIN	225	RPT$	226	unused		
227-231 not used				232	NUMERIC	233	DIGIT	234	UALPHA		
235	SIZE	236	ALL	237	USING	238	BEEP	239	ERASE		
240	AT	241	BASE	242	unused	243	VARIABLE	244	RELATIVE		
245	INTERNAL	246	SEQUENTIAL			247	OUTPUT	248	UPDATE		
249	APPEND	250	FIXED	251	PERMANENT	252	TAB	253	# (files)		
254	VALIDATE										

Using tokens like this slightly eases program writing- no names mentioned, but on some computers you have to tokenise yourself to save memory-
eg use ? instead of PRINT or g instead of GOTO
- which can make a listing difficult to read and input!

You can use this information to write programs that write programs - or to alter your program as it is running. This is very advanced programming requiring Extended Basic and Expansion Memory (or Mini Memory & TI. BASIC). Suffice to say that you use CALL LOAD to load the code number into the 'appropriate' memory location.

Drop me a line if you want more info & if the demand is there- maybe a followup article!

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

HOW DO YOU FEEL ABOUT

PIRACY

Personal View:

Microcomputer Software is very easy to copy on cassette without authority - a lot cheaper than buying it - but apart from the legal considerations, what is the effect going to be if only a small number of program sales are properly authorised???

A really good program requires lots of time - and especially lots of debugging time. Time of course costs money.

When an author has a program published, he receives ROYALTIES on each sale of the program - but NOTHING on pirate copies.

If ten copies are sold and then the purchasers each make four copies for their friends , the author will receive royalties on ten copies, but there will be fifty copies knocking around!

Royalties are not very high on software- typically an author will receive 8-12% of the net price - to encourage an author to spend lots of time writing additional programs - and investing the time to write GOOD programs - he has to receive royalties on a large number of sales.

Are you still with me?

Most of the price of software goes to the retailer - who has to pay for a possibly unsalable stock, and usually at least a slow-stock-turnover. Profit margins of 100% are neither unusual nor excessive. Some retailers will go down to 40% if they can keep stocks low or turning over quickly. Not unfair?

NOW - QED and all that - if you as a computer owner want a large range of really good quality software to be readily available -

ENCOURAGE the writing and supply of software by purchasing your programs from reliable sources - give the authors a break. You are quite capable of writing that unique classic program yourself!

Buy- or deal in - pirate copies - and you are driving good software off the market.

Memo all members: 99er Magazine are selling their programs on cassette - more than any other source they do deserve your financial support!! Programs published in other magazines will require translation and considerable work to make them 'good' - so copyright problems are not so severe.

Well thats the way I feel. What about you?

regards Stephen

stephen shaw

PROGRAM POSTMORTEM

When you have read every book on Basic you can lay your hands on you will know all the commands - and perhaps be a little confused. Whatever you want to do in your program - there are at least a dozen ways of tackling it in Basic! Some ways are 'better' on all computers - some are 'better' just on ours.

To understand basic, list all your programs - try to follow what is happening, and even if you get lost, try to see what each line is there for .
NB: No program is perfect! Watch for flaws - see if you can improve on the original.

I reccommend to you Peter Brooks articles - there are some real gems of programming to be found, and they are not that difficult to follow if you concentrate a little!

Reread Pete's article in Issue 4 on the use of POS & CALL KEY. I have read in one mass-circulation magazine that POS is of no use and never used. After reading Issue 4 you will use it in every other program! (Back issues are available from Paul.50p).

ISSUE 1 of Vol 2- TI HOME:
Listings by Peter Brooks:

Page 1: Nested Ellipses listing. Line 220. See
.....COS(N/15*4.....
Better simplified to N/60

Page 1: 3d plot. Line 210: SS=x1^ 2.
This could have been written: SS=X1*x1

What is the difference? SS=X1^2 uses 9 bytes
SS=X1*X1 uses 8 bytes

SS=X1^ 2 takes 12 ms. (1ms = 1/1000 Second) timings in
SS=X1*X1 takes 4 ms.) TI BASIC

Now- which do you think is 'better'. ?

((NB: I am NOT criticising Pete for his coding! This feature is to help you improve your programs - remember - every program can be improved!!!)))

Page 3: User-Directed Plotter.
Did you note lines 250 and 260 ? Look at them again. .
These lines can be used anywhere you use the arrow keys to move something! (Pete's article in Issue 4 helps to explain the coding). NB: If you have model A, change (K=0) to (K+1=1).

99er Magazine: Issue 4:

Interplanetary Rescue- p.35:
I have enclosed a separate review which lists some changes I have found to be needed to this program. One thing is clear- it was not fully tested!
There are also signs of a 'rush-job' in the program. Remembering that as was written in EXTENDED BASIC we shall take a closer look on the following page....

99er issue 4 interplanetary rescue p35

Lines 230,250,260,270:
```
CALL CHAR(33," FFFFFFFFFFFFFFFF")
CALL CHAR(95," FFFFFFFFFFFFFFFF")
CALL CHAR(42," FFFFFFFFFFFFFFFF")
CALL CHAR(98," FFFFFFFFFFFFFFFF")
```

That takes a lot of typing and a lot of memory.
Ignoring the memory used for the actual lines, the contents use up 128 bytes.
Now,in Extended Basic we can use instead:

```
A$=RPT$("F",16) :: CALL CHAR(33,A$)
CALL CHAR(95,A$)
CALL CHAR(42,A$)
CALL CHAR(98,A$)
```

That uses up just 75 bytes!
We have used two 'tricks' - the long string has been replaced by A$ - A$ uses two bytes but the actual string uses 18 bytes, saving 16 bytes each time it is used.
A$ has been defined using the RPT$ function. RPT$ uses one byte, () use two bytes, "F" uses three bytes,and 16 uses 4 bytes- total 10 bytes- quite a saving on using A$="FFFFFFFFFFFFFFFF" !

(note:- :: uses one byte).

LINE 330:
```
V=0 :: V1=0 :: V2=0 :: S=5000 :: F=0 :: H=0 :: D=0 :: T=0
```

wow! what a lot of memory THAT uses up - without need.
The memory used here is52 bytes.

In extended basic we can define several variables at once. The following line does not have EXACTLY the same effect as the above, but only really expert programmers need worry about the difference which has to do with allocation of memory. In practical use you can key in:
V,V1,V2,F,H,D,T=0 :: S=5000 which uses only 28 bytes of memory.
As part of LINE 370 we find: VOL=ABS((600000-F)/20000)
this portion uses up 27 bytes, but can be written:
VOL=ABS((6E5-F)/2E4) uses 22 bytes- not a big saving perhaps but useful if memory space is tight! Here for example , 6E5 has taken up 5 bytes.

As part of the same line is another lengthy segment:
IF VOL >30 THEN VOL=30 ELSE IF VOL<0 THEN VOL=0
In Extended Basic,this can be written: using 35 bytes
VOL= MAX(MIN(VOL,30),0) using 22 bytes.

This is a bit complex but if you work it through with different values of VOL you will see it works.
MIN (VOL, 30) for instance will return the value of VOL, unless it exceeds 30 in which case it will return 30.

For this example,an indication of timing is useful-
in Extended Basic then, the first way takes 15 milliseconds.
But using the MIN/MAX way uses 30 milliseconds!

Here saving memory slows down execution time! & as in this program memory is not tight,we would use the original way for speed,in preference.

You saw at the end of page 2 an excellent example which illustrates your need to know YOUR version of basic! Although a program may be better written in one way on one computer, a different computer may be faster using another way!
(I will draw your attention here to any significant timing differences when different methods are suggested)

Anagram Cracker - Microcomputer Printout - May 1982
Written for PET.

Looking at the problems associated with translating this program to the 99/4A will tell you something about the 99/4A-

I've already dealt (in an earlier issue) with translating LEFT$, RIGHT$, and MID$.

Problems come across:
DIM(LM)....In the PET program, the DIM statement is used with numeric variables - the 99/4A insists on NUMBERS.
...in the PET program,the DIM statements towards the end of the program are repeated several times in the course of the program. In the 99/4A this results in
NAME CONFLICT (TI Basic) or
IMPROPERLY USED NAME (Extended Basic)
You may only DIM an array ONCE.
....In the 99/4A, when the computer looks down a program, and sees for instance A(2)=5, if it has not seen a DIM statement, it automatically sets DIM A(10). Any subsequent DIM will cause the above error messages.
In the PET program,the DIMs come at the end- in a converted program they must be at the beginning of the program.
.....In the PET program we come across,for instance:
N1(N1)=5. This is also a name conflict on the 99/4A, as the variable N1 may be a simple numeric variable, or it may be an array variable -it cannot be both.
My suggestion here is that you prefix the array variable with A(for Array!) eg AN1(N1)=5 -remembering to DIM AN1 and not N1 !!

There are also problems with FOR...NEXT loops-
for instance,in the PET program we find:

```
760...For J=1 TO B1 : FOR JJ=1 TO B3
765...FOR JK=1 TO L1-1
770...IF A$(JK) <> L$ THEN NEXT JK,JJ,J : GOTO 795
780...JK=L1 : NEXT JK,JJ,J : GOTO 795
```

This causes all sorts of problems. First,translate 770 (To Extended Basic)
to: IF A$(JK) <> L$ THEN NEXT JK :: NEXT JJ :: NEXT J etc

Syntax Error- You CANNOT place NEXT... after a THEN clause,on the 99/4A.

Cure by amending 770 and adding a line 771:

```
770 IF A$(JK) = L$ THEN 780
771 NEXT JK :: NEXT JJ :: NEXT J :: GOTO 795
```

Syntax Error: Next Without For!!! The NEXT's relating to the FORS at 760/765 are now to be found twice,leaving one unmatched set! We must remove the odd set -which we do by a 'goto' leaving a 99/4A listing looking like:

```
760...FOR J=1 to B1 :: FOR JJ=1 TO B3
765...FOR JK=1 TO L1-1
770...IF A$(JK)=L$ THEN 780
771...NEXT JK :: NEXT JJ :: NEXT J :: GOTO 795
780...JK= L1 :: GOTO 771
```


Computing Today- June 82- FORTH SIMULATOR for TRS 80:

This program produces the usual problems with (IF-THEN) and (FOR-NEXT), and with the same name for simple variables and arrays.
There are some new points of interest though-

In line 20:
CLEAR 2500- you can ignore this completely ,it reserves string space,& you dont have to do this with your 99/4A
DEFSTR A,B,S- You must take note of this line. Now,whenever the TRS80 meets a variable (simple or array) with a name which begins with A B or S, it considers the variable to be a string - the $ sign is not used.
Thus for DIM A(40) -same line- you would need to use DIM A$(40), and take care to add the $ sign to any relevant variable/array.
DEFINT C-R,T-Z- In most programs,including this one,you can ignore this command. You can instruct some micros to work only with integer arithmatic,which is faster than floating point decimal arithmat ic. You cannot do this on the 99/4A.
If there is some other reason for telling the computer to use the integer value of a variable you can of course use INT - for instance,
DEF A = INT(A). This way you use INT only once.

(In this example A,B,& S are not included in the instruction- they are strings!)

There is a bug in this line too- DIM N should be DIM N(20).

To conclude our look at this program, lets look at lines 40 to 60:

ORIGINAL: (converted to 99/4A):	REWRITTEN: (excluding FOR NEXT):
`30 FOR J=0 TO 9 :: S$(J)=" ." :: NEXT J`	`30 J=0` `40 S$(J)=" ." :: J=J+1 :: IF J<10 THEN 40`
TIME: 170 ms	270 ms
MEMORY: 30 bytes	47 bytes

The two versions do exactly the same, but in the second version we are using IF THEN instead of the FOR NEXT loop. You can see the difference in execution time and memory used.

SOFTWARE REVIEW

INTERPLANETARY RESCUE

99er Magazine Volume 1 Issue 4

Also available on cassette. Airmail to UK for US$19 inc 2 other progs.
99er Magazine P O Box 5537 Eugene Oregon USA 97405

Subs: US$13 for 6 issues, airmail.
Subscribers dealing DIRECT with USA address: Cassette US$ 13 inc air mail
(not available to MPI subscribers)

One of the first computer programs attempted is often a MOON LANDER. These programs abound.

This is a very good implementation, written in EXTENDED BASIC and using 3 sprites.
There are 12 skill levels- by 3 levels of gravity and 4 maps.

The surface is rep. by a map, with your craft moving horizontally -using the 4 arrow keys.
You have momentum, but no inertia or friction.
There is a graphic altimeter at the side with another pic. of your craft, going up & down.

You must move from station 1 to station 2 and return. Safely.

The original program attempts a scoring system, but this was found inadequate, so I have implemented my own, using variable CR-see right-- Once you learn to get there & back, go for a high score - 20% on time & 80% on fuel used.

There were a few bugs found- wrongly numbered GOTOs, dropping through subroutines. These are dealt with in the listing at right.

Having corrected the bugs and inserted a meaningful scoring system, this is a very good game indeed, with lots of playability. Strongly recommended. Skill required (real-time, so very fast graphics not needed).

Also displayed:
height, vertical & horizontal velocity, fuel left, power setting, time.

ALTERATIONS I
FOUND TO BE REQUIRED:

```
1120 IF TRIP=1 AND TIME<250 THEN
 CR=TIME/2000 ELSE CR=.15

1220 DISPLAY AT(10,3):"YOUR SCOR
E IS:";INT(((2000-2*TIME)+E/G+(O
PT1*OPT2*500))*CR)

1490 CALL CHARSET :: IF V<=-10 T
HEN 1290 ELSE CALL HCHAR(22,1,32
,96)

1495 IF CO<>62 THEN 1270

420 IF H=0 AND TRIP=1 THEN 1270

1300 DISPLAY AT(23,1):"CRATER A
MILE WIDE" :: CR=.1

1340 DISPLAY AT(23,1)BEEP:"ARE D
EAD,AND YOU ARE HURT" :: CR=.15

1360 DISPLAY AT(23,1):"THIS IS Y
OUR LAST FLIGHT" :: CR=.25

1420 DISPLAY AT(23,1):"LOST HALF
 YOUR FUEL." :: E=E/2 :: CR=.7 :
: GOTO 1450

1440 DISPLAY AT(23,1):"IN GOOD S
HAPE TO RETURN." :: CR=1.00

1520 CR=CR*.750 :: GOTO 1550
```

Sorry about this - some bugs in my debugging!

```
1280 IF XC<>137 AND YC<>185 THEN
 GOSUB 1160 :: CR=.08 :: DISPLAY
 AT(22,1):"YOU MISSED THE PAD." :
"YOUR SHIP HAS CRASHED" :: GOTO
1190

1460 NEXT TO :: F=0 :: TRIP=2 ::
 GOSUB 190 :: GOSUB 1655 :: GOTO
 370

1655 IF TRIP>1 AND OPT1=3 THEN T
OFF=250000
```

** EXTRA alteration for 99/4A:
820 IF kl+1=1 THEN D1=D1+1 :: E=E-50 :: GOTO 860
(different value of zero using call key.)

Stephen Shaw nl82

42

SOFTWARE REVIEW TI COMMAND MODULE

TOMBSTONE CITY (21st CENTURY)

Unlike the other new game modules, this is an ORIGINAL TI creation - a NEW video game.

Joysticks are optional, but I reccommend them- my fingers ached using the keyboard, after only a few games. The action is FAST.

Basic theme: You command a fighting ship, which has a 'safe' area in the centre of the screen. You can be blocked into this area (you lose then).
Outside, there is tumbleweed getting in the way, and pairs of cacti (saguaro). From time to time, from behind a pair of cacti a Nasty emerges- this is called a MORG and it is after you. You may shoot it -hit it, and it turns into- a cactus! Wherever two cacti gather together is yet another hiding place for morgs- so you have to watch where you shoot them.
If you shoot a Morg when it is adjacent to one cactus of a pair, the pair dissappear and are replaced by one or two more Morgs (this is the only way to remove cacti).
Your aim is to destroy all paired cacti, and all live Morgs. Any tumbleweed you destroy also adds to your score - destroy it all and another lot appear.
When all pairs and Morgs have gone, you start another day, but this time there are additional paired cacti to start you off!
You have 10 ships (this aint an easy game!), and nothing will move when your new ship appears, until you press a key- so you can study the screen a while.
The maximum number of pairs at the start is 30 -you would have to be very good to get this far. You select from 3 sets of rules at the start of each complete game -3 levels of difficulty, ending with 'insane'.
The Manual suggests a Morg will appear every ten seconds, but warns- if there are less than 10 tumbleweeds, and no living Morgs-provided a pair of cacti remain, a morg will appear instantly!
Helps: Touch the space bar to move by warp drive to the safe area (costs you 1000 points). Before a Morg emerges from behind a pair of cacti, one of them is surrounded by a white energy screen -it gives you a little warning.

The Title Music is fascinating.

The characters are not too well defined perhaps, but adequate for the game. Sprites have NOT been used, and movement is therefore not smooth- but it is so fast you wont mind too much.

This is the first Module Program to appear on Disk, for running in Extended Basic (expansion memory also required)- it comes as a 'sample' with the EDITOR/ASSEMBLER package, in Assembly Code.

I dont think I'll be testing the capacity of the maximum score on this game (No high score to date indicated-pity) for a while- it is hard- but it is a fast action game with plenty of strategy and little luck. I may even buy joysticks....

Highly recommended.

Stephen Shaw

H

This is for Mr Freeman of Dorset & all you new members struggling with TI BASIC's lack of PRINT AT/ACCEPT AT/ restricted scrolling:

If you have the PRK module, ACCEPT AT/DISPLAY AT is available as CALL A and CALL D (works with STATS module too).

This was well covered in an article in TI HOME Issue 5 - send Paul 50p for a copy.

If you do not have those modules and also cannot afford Extended Basic, all hope is not lost - you can do a very creditable job with CALL KEY and CALL HCHAR.

To display a message at a particular position, you must set up a loop to look at each character in the message, and place it on screen with CALL HCHAR. The subroutines are part of TI's Programming Aids I.

eg to display "TEST MESSAGE" on Row 5, starting at column 4:

(using the HCHAR columns- 32 instead of PRINT's 28):

```
 8.  COL=4
10. M$="TEST MESSAGE"
12. FOR T=1 TO LEN(M$)
14. CALL HCHAR(5, COL-1+T,ASC(SEG$(M$,T,1)))
16. COL=COL+1
18. NEXT T
```

\- - - *** TRY IT !

To input a message or value without scrolling the screen calls for a little more skill, using CALL KEY.
Briefly, to input a message to string M$ (which can be printed using a routine as above -following doesnt print):

```
 8. M$=""
10. CALL KEY(0,A,B)
12. IF B = 0 THEN 10          (no key pressed)
14. IF A= 13 THEN 20          ("enter"key pressed)
16. M$=M$&CHR$(A)
18. GOTO 10
                              (line 8 is M$="" -ensures no carry
                                                  overs)
```

Within this short routine you can test for various keys being pressed- eg ensure a digit has been entered.

There was an excellant article on this in TI HOME ISSUE 4- another 50p to Paul.

Brief example (Paul doesn't like repeats):

Add to above routine,to restrict key pushes to digits:

```
15. IF (A < 48)+(A>57) THEN 10
```

TRY IT

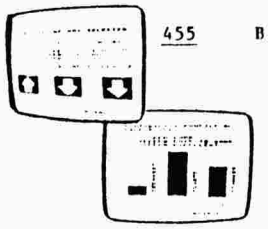
To change M$ to number form, use N=VAL(M$) -you now have a numeric variable you can use in your program!

Stephen

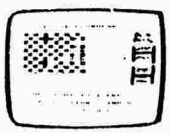
PrP computergraphics
901 Cherrydale Street
Lake Charles LA 70605

10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

PROGRAM NO. DESCRIPTION

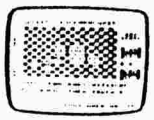


455 BIORYTHYM... (T. I. 99/4 Basic)
Outstanding use of Color Graphics show computer analysis of Bio Rythyms based on any date since 1801. Displays mental, physical and emotional status for any given day. Graphics dramatically change to show status of succeeding days. Same program contains a Compatibility analysis for any two people. Results shown in spectacular color graphics, with sound. Also contains humorous comments and notification of error inputs. Exciting program.

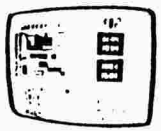


470 CROSSWORD CHALLENGE... (T. I. 99/4 Extended Basic)
After instructions, screen displays a large, easy-to-read crossword design. The object is to fill in the crossword squares. There are only 6 words needed to fully complete the puzzle, but doing so can be difficult. Two players compete in this word game. Bonus letters offer extra incentives when used in constructed words. Computer keeps and displays continous score in personalized color Box Scores. Program is easy to learn, winning a bit more difficult. Beautiful use of Color Graphics add enjoyment to this fun game.

457 SCRIBBLE... (T. I. 99/4 Basic)
Fascinating Computer Word Game. Players form crosswords from 8 random letters offered by the computer. Game is displayed in a colorful graphic playing board. Game rules are similar to Scrabble. Scores are calculated and displayed at all times in personalized color Box Scores. Extra Bonus Color Squares are included in each game-board setup, always different in each game. This one is exciting and easy to play.



471 BOX LINES... (T. I. 99/4 Basic)
This is a computer version of the old children's game of connecting dots to form Boxes. The object is to complete more boxes than your opponent. Whenever a box is completed, the computer will color that box with the player's assigned color, calculate and continously display score in personalized color Box Scores. Exciting graphic screen playing board provides for 100 boxes to be completed. Program routine prevents error inputs. Strategy and planning ahead helps the contestant to win. Kids will love it! Two players can enjoy this fun game.



These, and all, PrP COMPUTERGRAPHICS cassette programs for the T. I. 99/4 Home Computer are priced at £6 each, any two for only £11, and three or more at £5— each. Send check or money order, including your name and mailing address, and the names and numbers of the programs you desire to:

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

Authorised U.K Distributor for
PrP COMPUTERGRAPHICS
901 Cherrydale Street
Lake Charles, LA 70605

Cheques and "Stephen Shaw" please.

44

SOFTWARE REVIEW: PRP COMPUTER GRAPHICS programs.

Available from Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH
at £6 each inc p&p.
Discounts available.

SCRIBBLE

Loosely based on Scrabble. Start with a checkered board. At random put 6 bonus squares. (fixed bonus). Add one red starting square.
At each turn, each (of two) players must spell a word selected from random letters presented to him (unused letters are NOT carried over- a fresh selection is made each time). Each letter scores a certain number of points.
The words are placed on the board as in Scrabble-eg interlocking.
This program (which fully fills memory) does NOT check to see if the word you are placing is possible with your letters, OR if the interlock is correct -players must referee the game themselves. The computer displays the board, presents the letters, places the words, and keeps score. Max word length=8 letters.

An interesting program. Solo play is possible- aim for the highest score sum possible, using the letters presented.

CROSS WORD:

Very similar to the above, but a different game- take a 5x5 board and block in the four corners of the medium square, forming three lines horizontally and three vertically - this is easier with a diagram:

1	2	3	4	5
6	▓	8	▓	9
0	1	2	3	4
5	▓	7	▓	9
0	1	2	3	4

So- you have a crossword with just six words.
You may use any letters, but some have bonus points (the bonus letters & their bonus are different in each turn).
Each player in turn places a 5 letter word. Where words interlock the letters must be the same- but this (very large) program does not check- players must referee themselves.
The computer keeps the scores.
Solo play is possible -aim for the highest JOINT scores. (Because different letters have different bonus points, in solo play you ARE presented with a different problem each time).

BIORHYTHMS:

You believe in them or you dont. This program indicates whether you are + - or 'critical' in each of the three categories, for any day, and can also indicate 'compatability' with another person.
No explanation is given of + - or 'critical'. There are some nice touches hidden away in this program.
Take it with a good dose of salt and this is an amusing little program!

JUMBLE:

This is an exceedingly simple program. Most of you could write it. You key in the length of your group of letters, key in the letters, and the computer lists every possible combination. Thats all.

EXTRA PROGRAM

AX

BENCHMARKS

No, not graffiti on your desk!

When you see reviews of microcomputers in the magazines, you will frequently see reference to BENCHMARK TIMINGS. These are rarely specified or analysed - except that you will frequently get the impression the 99/4A is much slower than anything else around!

The time a computer takes to do anything depends on a lot of variables - the 'clock rate' of its processor, which sets the number of times those tiny electronic switches can switch on and off in any time period. -the language used & the way it has been written. -the SIZE of the language (if the computer has only 6 commands to look up in one language and 600 in another, obviously the 600th command will take longer to perform than the 6th!).

The TI99/4A is particularly slow on certain things, most of which are a major part of the benchmark tests! As the tests are cumulative programs, if you do badly on Test 1, you will do equally badly on Test 2.
The timing of a particular test is not the final thing to look at- look at the time difference between adjacent tests to see how long the EXTRA items have taken!

Let's have some BenchMark timings for the 99/4:

LISTING OF BENCHMARK 'S' (non-standard)- you can see here the general routine to be used. Note the use of IF-THEN which is used instead of a FOR-NEXT loop. See next page....

2021 explanation: Line 30 below is for TI Extended Basic Version 100 and speeds up execution by turning off sprites. It is NOT required for Version 110.

```
30 CALL INIT :: CALL LOAD(-31878
,0)
100 CALL KEY(0,A,B)
110 IF B=0 THEN 100
120 K=K+1 :: CALL CLEAR :: CALL
HCHAR(4,4,66):: B=ABS(-K):: RAND
OMIZE :: C=INT(RND*K):: D$=CHR$(
K+40):: PRINT "+" :: IF K<100 TH
EN 120
210 PRINT "//---//"
220 STOP
```

SUPERFAST EXTENDED BASIC
TEST "S"

```
100 CALL KEY(0,A,B)
110 IF B=0 THEN 100
120 K=K+1
130 CALL CLEAR
140 CALL HCHAR(4,4,66)
150 B=ABS(-K)
160 RANDOMIZE
170 C=INT(RND*K)
180 D$=CHR$(K+40)
190 PRINT "+"
200 IF K<100 THEN 120
210 PRINT "//---//"
220 STOP
```

TI BASIC
TEST "S"

NB: Using one program line in Extended Basic (120 above) instead of several lines saves you about 1 millisecond per loop - not a lot!

TEST NO:	TI BASIC:	EXTENDED BASIC:	FAST EXTENDED BASIC (with 32k EXTRA RAM)	
1.	2·9	6	3·7	This column is for XB v100 with sprites turned off with a CALL LOAD.
2.	9	18	11	
3.	23	46	28	
4.	24 (!)	46	28	
5.	26	50	31	
6.	61	94	60	
7.	86	148	92	
8.	380	240 (!)	210	
'S'	270	370	240	

What are these tests & what do they mean?

1. Is a simple FOR-NEXT loop. (All these tests are 1000 times, time in seconds)
2. Instead of FOR-NEXT this uses IF-THEN,and includes one addition per loop. You will see that creating your own for-next by using an increment and checking it is much slower.!
   eg K=K+1 ,.... (contents),.., IF K<5 THEN (back to increment)
3. Adds to 2 four arithmatic operations,using numeric variables.
4. Instead of numeric variables this one uses numbers,and on many computers this runs faster than 3,'cos the computer does not have to look up variables. Note that in TI BASIC this is slower than 3! possibly because of the way memory is used.
5. Adds a dummy GOSUB routine
6. Sets up an array and dummy entries to enable you to compare it with 7:
7. Which is 6 but this time the array (5 items) is filled 1000 times. In TI BASIC, 7-6=25. In Fast XB 7-6=32
8. This drops the dummy gosub and the arithmetical operations. Here we are looking only at ^,LOG & SIN. It is particularly interesting as for this benchmark Extended Basic is FASTER than TI BASIC, even before you speed it up.!

'S' -see listings- this was added by me to look at some particular commands and functions.

Now do your own time tests on individual commands & functions, and the different ways of doing things. See also the sheets headed 'Program Postmortem' which have some timing details.

IMPORTANT: Benchmark timings only relate to the short benchmark programs,not to typical programs.
In particular,I have found fast extended basic to always run faster than TI Basic, usually by 30%, largely due to faster screen displays and faster line transfers -the latter only apparent in a program of reasonable size.

There is now an additional TI dealer in Manchester - Micro C (a division of CURRY's) - selling it for £200, with a console switched on and running (Zero Zap) - keyboard available for use, and staff knowledgeable and helpful. Only peripheral available was a speech synthesiser but there was a good range of modules.

There have been a lot of new computer mags recently - I've duly bought each one to have a look at it, and am amazed at the originality and quality of the programs published (pretty awful) and the thin quality of the editorial material. There is clearly a market - but our own magazine 99er is head & shoulders above those offered here. The better ones- in my estimation - would appear to be Computing Today and the American 'Creative Computing' , with Computer & Video Games Magazine coming a very close third and Personal Computer World a distant fourth. The remainder a long way behind.

In the February TI HOME I quoted Creative Computing correspondence dealing with the TRS80 - the May issue has further-

"To eliminate the freeze-ups, you must eliminate static electricity. My model I is so sensitive that it would freeze up if I even came near it and did not touch it."

The writer has gone to the extreme length of fastening an earthed band around his wrist and reports that it works.

"Second I have installed gold-plated edge connectors. They have totally eliminated the re-boot problem: no more oxidised connectors"

(Richard A Press. MD)

Sadly, the TI MODULES do NOT have gold plated contacts - and this causes our problems. The peripheral contacts are also subject to oxidation.

WARNING: UNETHICAL TRADING: Electronics & Computing Monthly:

Having sent off my cheque for a classified ad I thought no more about it - until this am when I received what had the appearance of a request for money for my ad -at four times the published rate.

A response has been received from the magazine, saying the 'offer' (?) was an error. The tone of the letter is most unsatisfactory, the author apparently thinking it of no great concern if his staff apparently request money from customers without need.

I shall not be advertising in nor purchasing this mag. again.

AD:-

99/ER MAGAZINE
P. O. Box 5537
Eugene, OR 97405

99er magazine™ SUBSCRIPTION

☐ YES–Please sign me up as a subscriber. Enclosed is my payment

Term	U.S.A.	Canada & Mexico	Foreign Surface	Foreign Air
1-yr (6 issues)	☐ $18	☐ $22	☐ $28	☐ $43

Address shown is: ☐ Business ☐ Home ☐ Check enclosed MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

NAME _____ PLEASE PRINT

ADDRESS _____

CITY _____ STATE _____ ZIP _____

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

N

PRESS CUTTINGS

(My own ads are excluded - ~~56~~ 58 progs available at present- + services - SAE please!)

MICROCOMPUTER PRINTOUT JUNE 82

MICROCOMPUTER PRINTOUT JUNE 82

Texan digits

Further to Mikes Muses (March 82) and the letter from Mr Bobler, I would like to point out that both of them base their generalised statements about 'Computer Accuracy' on old-fashioned eight-bit micros.

My TI99/4A gives me the right answers to all their examples and this brings me to my next point – why did your reviewer in the May issue not touch upon the numeric capabilities of the TI? Far from "keeping the enthusiast at bay", the highly accurate number crunching of the T.I. must be an additional attraction to the numerate computer enthusiast. After all, if a machine cannot perform basic maths functions accurately, it's not much of a computer, is it?

P.J. Haydon,
London N.W.1.

Well done, Texas! The accuracy of the TI99/4A's mathematics probably owes a great deal to that company's previous experience in pocket calculators, which are generally a great deal more precise than desktop computers.

For the majority of users, for whom processing is more important than calculation, however, most computers are accurate enough. Good programming always involves avoiding statements of the form IF A=B anyway. (Huh? – Ed.)

Incidentally, Mr Haydon, accuracy has nothing whatever to do with having 8 or 16 bits; it is the Floating Point routines inside the BASIC interpreter which count.

.1 divided by ten equals .001 – at least it does according to the IBM Personal Computer. "It's a bug in the BASIC interpreter ROM" admitted a spokesman for the jolly grey giant. So you'll be replacing the ROM? "Er, No...we are recommending customers not to divide .1 by ten..."

COMPUTING TODAY JUNE 82

TEXAS T199/4A (SLARI) A fast action game. Try to trap your opponent with your snakes trail before he does. Colour graphics and sound. £4.50 inc. postage, Michael Yates, 42 Rose Lane, Chadwell Heath, Romford, Essex.

ALSO AVAILABLE FROM ME £5-00.

WOULD YOU BUY A CAR FROM THIS MAN?

How many versions are there of this photo ? (You hadnt noticed they're NOT all the same?) Seriously- lovely photos, they ARE eye-catching, and friendly. Nice touch TI.

Mr. & Mrs. S. Shaw
10 Alstone Road
STOCKPORT
Cheshire SK4 5AH

END SHEET RAMBLES

At last,last sheet....

In program postmortem I have covered also some specific program conversion problems - did you find these useful? If you would like more of this sort of thing,please drop me a line. Only if there is a clear response will I continue this feature.

If you have forgotten my request in sheet One - a while back -

If you find the style and content of TI HOME unsatisfactory PLEASE:

1. Write and tell us
2. Consider submitting a sentence,paragraph,page,pages - anything! News reviews comment. Programs even.

If there has been anything in the last few pages which you would like expanded or better explained, please- write and tell me!

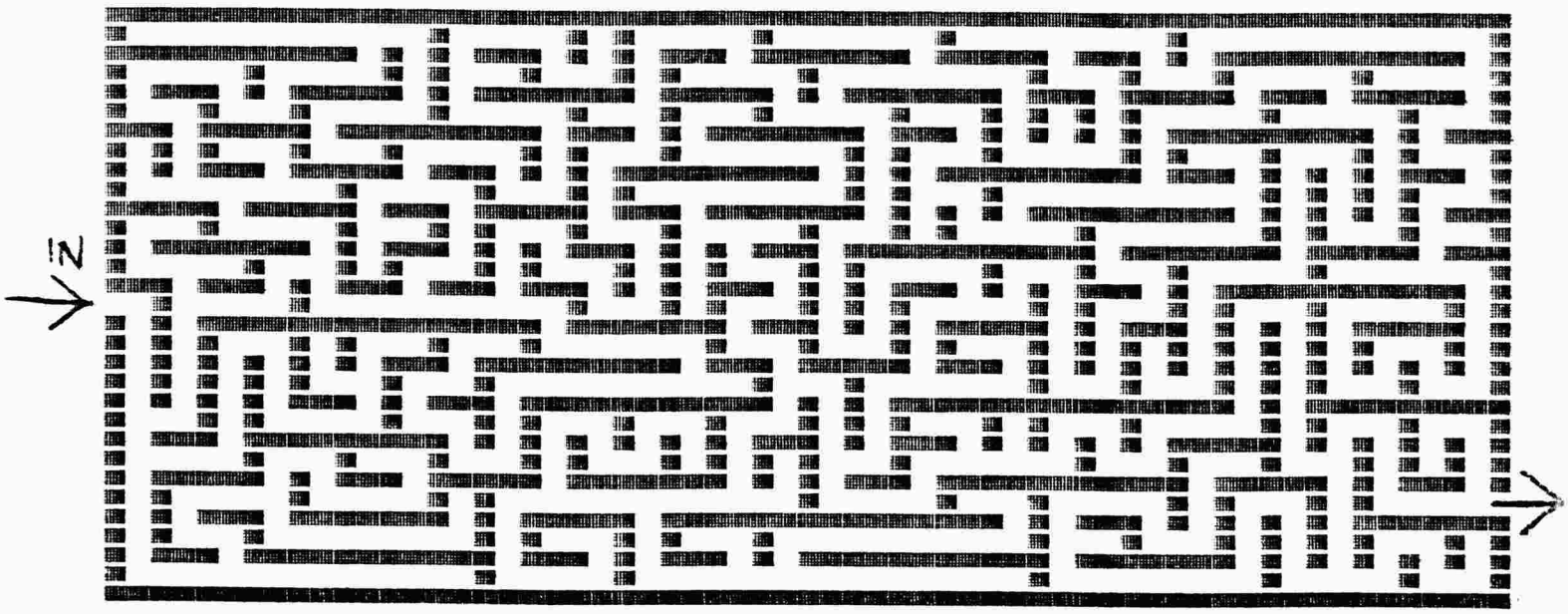
NB: Such are the press dates that when you receive this copy, my article for next TIHOME will probably have been written and sent off!
Please bear with me if response time is a little slow.

Drop me a large SAE for details of software & services I can offer.

IF any dealer pleases or displeases you, let us know. No-one is perfect and mistakes are made- but perhaps we can communally discover anyone who is prone to error!

Best wishes & happy computing.

Stephen



QUESTIONNAIRE.

1. Do you object to your name and address being published?

2. Do you think a Children's Page would be of use in Tidings?

3. Do you think a Beginner's Page would be of use in Tidings?

4. Are there any services that TIHOME can give you that it does not already do?

5. What is your biggest complaint against TI?

6. Are you successful in contacting TI in Bedford.

7. Are you prepared to use TIHOME as your go-between in dealing with TI in Bedford?

8. Do you have any suggestions for articles in Tidings?

9. Do you have any particular problems with hardware?

10. What did you think the 99/4(A) could do for you when you bought it? Has it lived up to your expectations?

Please fill in this questionnaire and return as soon as possible to:-

TIHOME
Paul Michael Dicks MIDPM
157 Bishopsford Road
Morden
Surrey

Name :-
Address :-

POSTSCRIPT

Well, there you are, yet another newletter has come to an end. Think about it, and enter your opinions on the questionnaire and send it back to me. You must realise that I started TIHOME over 18 months ago and have been running it very much in isolation ever since. The first year was really difficult as I was running it out of my own pocket, however, things have improved since then, so come on, let me know what you want and need and I will attempt to supply the requisites.

Do not be worried or disturbed about any of your problems. When in doubt ring TIHOME. I will always try to give you a sensible answer. If I can't give you a sensible answer I will include your moan in my next nag to TI thamselves.

TI have a great respect for TIHOME. Use this respect, if you have a problem get in touch with me and I will pass it on. Be assured, I always get an answer. If the problem is really tragic I will put you in touch with TI and you can talk to the TOP people.

I want to start a Computer Pen-Friend spot in the newsletter. If you want a pen-friend, let me know and I will publish your name & address in a dedicated section. Anyone answering a request for a pen-friend should accept that they may or may not be answered.

Another thing I want to do is to organise meetings for members in different districts, so be sure to fill in the questionnaire and give me some idea of who I can contact to organise district meetings.

I am always willing to use offers of help with the running of TIHOME. If you are interested, please let me know with a note of what you think you can do. Please realise that TIHOME is a non-profit making organisation and so cannot afford to pay anyone, including me, as 100% of the subs go to running the organisation. In the future, perhaps, TIHOME can pay wages and fees, but the membership will have to rise a lot above 180. Anyway, think about it!!!

A last tip for you beginners with a 4A, please remember that OLD CS1 and SAVE CS1 must be input with the ALPHA CAPS key locked down. The cassette interface does not recognise Old or Save Csl. You understand?

Tidings has as yet not taken adverts, however, I am now prepared to take adverts at the rate of £5 per A4 sheet per one issue, or proportionally less per area used per issue. Personal adverts will be accepted at the rate of 20p per line of 90 characters per issue. Send any adverts in with the correct money detailing the issues required, remembering that there are 6 issues of each volume each year, and each volume begins in February of each year.

If you send programs for inclusion in the Library, please remember that they will probably be distributed world-wide. Do not include your name & address or any request for money unless you are prepared to deal with the problems of international exchange, or are prepared to receive letters from Australia or Hongkong, etc.

God bless, and the best of computing luck,

Paul

Published by:- TIHOME, Paul Dicks, 157 Bishopsford, Morden, Surrey, England.
Registered No. 2778359

POSTSCRIPT.

I have included a new copy of the questionaire for those of you that had no chance of completing it. I have also included another copy of Postscript for those who completed the questionaire and lost another page of Tidings.

A little problem:-

Type in:- A=1.00000000001
PRINT A
the answer will be:- 1.
where has the rest of the variable gone???
If you have ExBas type in:- PRINT USING "£.£££££££££": A
you will get the answer:- 1.00000000001
so the complete variable still exists somwhere, as you will find if you do any form of arithmetic on the variable.

2021 note: Use # not a £ sign!!!

Does anybody notice the deliberate mistake???

TIHOME will be on holiday for the first two weeks of August, so time your requests for tapes and listings carefully. I don't really want to arrive home to a massive pile of letters. However, if that's the way it will be.

In future issues of Tidings I shall include recent new additions to the software library. However, to start things of right I am rewriting the software library incorporating lots of information. I will issue a copy of this new software catalogue with the next issue of Tidings. I would like to thank Pete Brooks for doing the job of checking all the programs and providing the information I will include in the catalogue.

Don't forget, if you have any problems, please ring TIHOME. If we can help you we will. For the record the phone number is 01.640-7503 between the hours of 7p.m. and 11p.m. If you have a really pressing problem, you can contact me between the hours 9a.m. to 12a.m. and 2p.m. to 4p.m. on 01.648-7090 Ex 302. I would request that you do not ring me at work unless you really have to, for I am frequently out of my office and quite often out of the building.

Those of you that are kind enough to send s.a.e.'s really help the club, however, if you want to send return postage just send stamps. Quite often when you send envelopes that are not big enough so stamps are a much better bet.

Best of luck and good computing.

Paul

Published by TIHOME, Registered No. 2778359.
Prop:- Paul Dicks, 157 Bishopsford Road, Morden, Surrey.

YOUR COMPUTER
(Australian Version)
MAY 1985

Also for beginners, I have two unusual books. I say unusual not because the subject matter or the treatment is unusual, but because both authors were once heavily involved in TISHUG's sister group in the UK, TIHome.

Peter Brooks, the author of *Mastering the* TI99 (Micro Press through ANZ, $21.95) was a pioneering member of TIHome in early 1981. Brooks established himself as a knowledgeable writer and programmer for *Tidings* the UK group's newsletter.

Mastering the TI99 is a valuable purchase, if only for the chapter on translating BASIC code between machines. Other chapters discuss file handling, graphics plotting, printing errors, hints and tips – all in Brooks' typically concise style. There are routines for checking memory, using CALL FILES(), and a short program for visually checking tape loading. There are another 28 tips included, and with its highly functional index I would recommend this book for the more 'advanced' beginners among you.

Stephen Shaw continues his involvement with the 4A in his book *Getting Started with the Texas* TI99/4A (Phoenix through ANZ, $19.95). Shaw wrote for *Tidings* as well as conducting a successful import business from his home in Stockport, Cheshire. He was bringing the best American software into England at a time when the majority of British computer magazines, with the exception of perhaps *Computer and Video Games*, was giving little or no coverage to the 4A machine. Texas Instruments' attempts to launch the early model 4 there were abortive, causing many journalists to write it off as a bad joke, and even with the introduction of the more versatile 4A no-one wanted to know it!

With the first-time user in mind, *Getting Started* is a guided tour from setting up and understanding your computer to programming in BASIC and Extended BASIC. (Did you know there are two versions of Extended BASIC? One is the Vn100 and the other, which is more common, is the Vn110. Apparently the latter version is significantly faster, but there is some degree of incompatibility between the two versions with respect to sprite handling and user sub-routines.)

This book, with its encyclopedia-like entries could almost qualify as a dictionary for the 4A. All the listings included have been fully developed and represent excellent examples of the Shaw approach to programming. They equal anything he has published in earlier editions of *Computer and Video Games* magazine, which, if you haven't guessed, is my favourite English magazine.