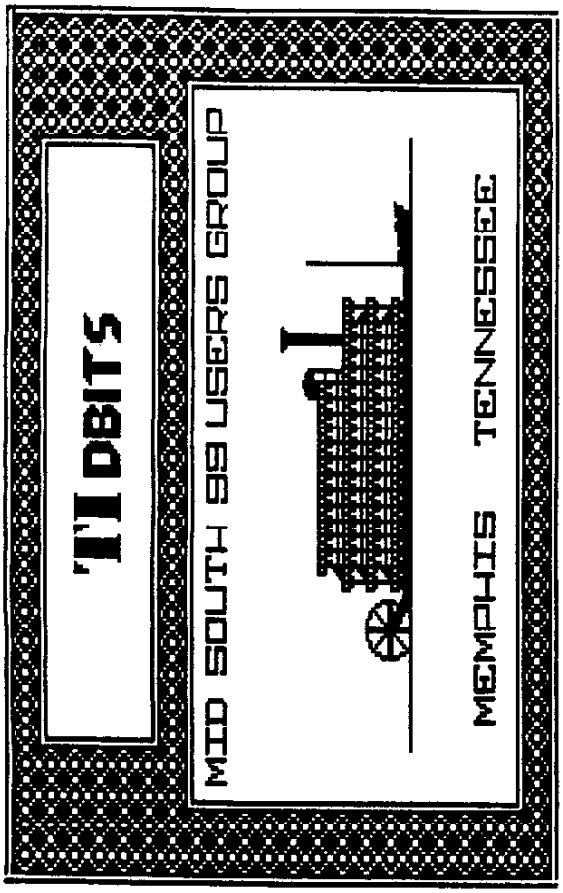
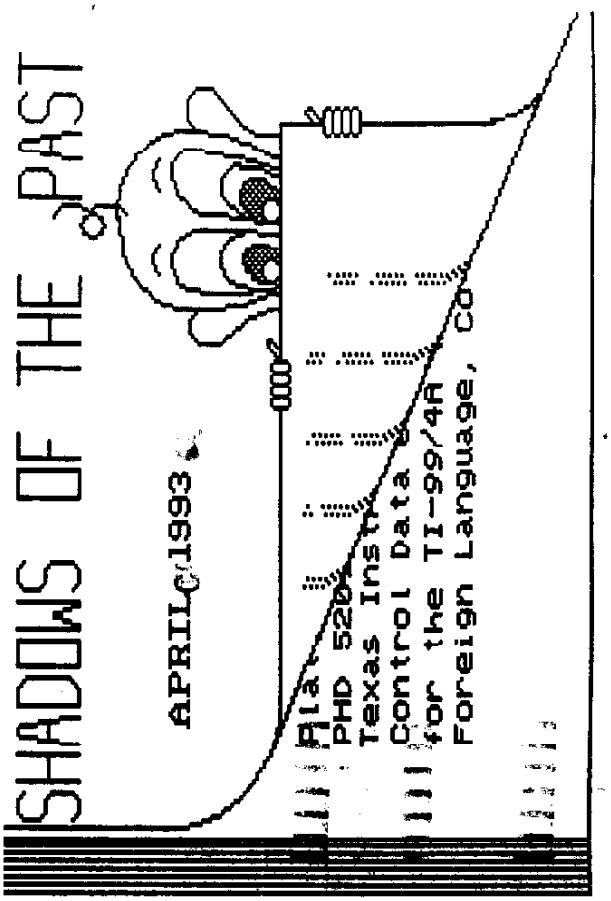




Mid-South 99 Users Group
P. O. Box 38522
Germantown, Tn. 3818

MEMPHIS, TN
PM
APR 11
1993
381

29 USA

FIRST CLASS

UG 2/86
DALLAS TI USER GROUP
P.O. BOX 29863
DALLAS, TX 75229

Newsletter of the MID-SOUTH 99 USERS GROUP-Vol 11, #4-APR 1993

TIDBITS

OFFICERS

Gary Cox	PRESIDENT	901-358-0667
Richard Hiller	VICE-PRESIDENT	901-794-9945
Richard Mann	SECRETARY	901-682-4195
Mac Swope	TREASURER	901-363-3880
Jim Saemenes	Technical Support	901-476-7011
Jim Saemenes	Disk Librarian	901-476-7011
Pierre Lamontagne	CO-Librarian	901-386-1513
Gary Cox	Program Chairman	901-358-0667
Mac Swope	Chairman - Equipment	901-363-3880
Marshal Ellis	Editor - TIDBITS Newsletter	901-327-2506
Marshal Ellis	Editor-Technical Interface	901-327-2506
Beery Miller	9640 NEWS BBS Sysop	901-368-0112

APR. 1993 INDEX

PRESIDENT'S BIT

By Gary W. Cox

We just had a donation of some TI equipment and software to the group of which we will selling at this months meeting but you must attend the meeting to buy!

As a reminder our second monthly newsletter publication (Technical Interface) is available only at the meetings while the one that you are reading now is always mailed out each month. If you would like to have Technical Interface mailed to you please add an additional $10 to your dues to pay for the

C ya at this months meeting...

RoundTable talk

from Berry Miller; MID-SOUTH 99 USERS GROUP

Welcome to the TI RoundTable from all the Genie Staff. We are presently supporting two assistant sysop (Tim Tesch and Barry Traver) and one programming guru (John Johnson) plus sysop (Beery Miller).

The TI RT is in a re-growth stage as the TI and Geneve enters a stage seeking compatibility to IBM standards. We already have standard IBM floppy drives achieving double density capability, our next step was achieving an interface with MFM hard drives. As the TI has continued to grow, new software was again written to use existing TI hardware to emulate IBM functions. First we saw MacFlix emulating MacIntosh and IBM graphics, followed by Picture Transfer bringing us the use of GIF's. Following Picture Transfer's release, we saw the release of GIF-Mania which brought GIF's to the TI-99/4A computer having only 16K of video memory which required 256K of memory on an IBM system.

The TI and Geneve have continued to grow. In the past two years, we have seen the TI play digitized sound files without the use of extra hardware costing hundreds of dollars. As a follow-up, the TI and Geneve saw the release of a cable expanding the capabilities of digitized sound to emulate the same IBM quality of sound.

In the past several years, we have seen two groups still grasping for additional IBM hardware compatibility. The first was with IDE interfacing which has not taken place. The second, SCSI, is nearing completion. Units were made and sold without software suppport. The software support is jumping the hurdles and is making progress.

Several other projects have been pursued in the TI Community. Video Digitizing was completed several years ago on two different approaches. The first approach again used IBM hardware through a serial interface to digitize. The second approach, though little known about it, achieved digitizing on a Geneve. The price was too high, $10,000 for the device and as a result, faded into history.

Other hardware interfaces have been built by users for both the TI and Geneve. The French have built an IEEE-488 interface (with software) for the Geneve and TI built (slightly unreleased) the IEEE-488 for the TI-99/4A.

On nearly all occassions, the TI has followed the footsteps of IBM co-existence. In one occassion, I believe TI users developed a product that has since been developed for IBM's. Around 1984, TI'ers saw the release of a TI-Ramdisk, called the Horizon Ramdisk that continues to be one of the best products available for the TI. TI'ers make more use of that single device probably than any other group in the US. Only in commercial industrial applications has the IBM been able to make use of the Ramdisk. TI'ers use the Ramdisk for general and home-owner use.

As we approach the year 1994, the TI is again making great strides towards IBM - co-existence. Mike Wright and company has released PC99, software that runs on a MS-DOS machine and emulates the TI-99/4A. Believe it or not, this is a MAJOR breakthrough for the TI-99/4A. Many folks have seen this as a further deviation from the TI, but this is not the case. It has a strong chance of bringing fresh programmers into the TI community. Many folks are not aware that the CPU of our machines are used in missile guidance systems. As a result, companies search for TMS9900 programmers. What is a better and more natural way to train to be a TMS9900 programmer when all you need to do is to run a single program (PC99) on your MS-DOS computer to test and debug routines. Further, as these folks train and test, these same folks will seek additional TI information and will release their own programs. This key point will carry the TI into the year 2000. The TI Family is getting to be old, just about time for them to become grandparents as new blood comes into the scene.

Folks, these are my thoughts and views. Feel free to discuss them among your usergroup. As an added incentive, this message may be freely reprinted in any non-commercial BBS or User Group Newsletter.

Beery Miller
GENIE TI and Orphans RT Sysop
9640 News Vendor

JOHN PHILLIPS

by Bill Gaskill

I would venture a guess that most people who have owned a TI-99 for more than a couple of years have run across the name John Phillips before. He is a near legend in the TI-99/4A cartridge and assembly language programming community and can claim authorship, co-authorship or significant involvement in over a dozen cartridge programs produced for the 99/4A, not to mention numerous articles written about the inner workings of the 4A's architecture.

Phillips will be 32 years old this year (1993) but he was only 21 when he was hired by Texas Instruments in 1982 right after graduating from Illinois State University. He started his career with TI in Dallas doing COBAL programming for business applications but it took him only 6 months to get a requested transfer to Lubbock where the "real" action was. John had purchased a 99/4 during his senior year in college and was already familiar with the Home Computer's architecture and he had wanted to program video games since purchasing his first cartridge, which was Munchman. Phillips didn't know TMS9900 assembly language but it didn't take him long to learn it.

His first project at Lubbock was Moonmine, followed by Hopper, which he co-authored with Michael Archulata. Hopper was followed by Word Radar, which he wrote in 2 weeks, for Developmental Learning Materials (DLM), the firm started by Bill Maxwell and Jerry Chaffin.

After completing Word Radar TI sent Phillips to Japan where he met with several companies who were being recruited to write software for the 99/4A. Following his return from Japan he became involved in almost every piece of software that was slated for production or that was actually produced for the 99/4A. When TI announced the end of the Home Computer Division Phillips was offered several incentives to stay at TI but turned them all down because none involved work with the 99/4A. Instead, he and fellow employee Michael Archuleta went to work for DLM, which had continued to work on products for the TI-99/4A even though it was no longer being produced.

In December 1983 John Phillips announced to the TI Community that he was available to any User Group for seminars, demonstrations and question and answer sessions related to the TI-99/4A. He would travel to virtually any location if the User Group would pay round trip airfare from Dallas, Texas plus lodging? While he could only make himself available on weekends, it was a pretty generous offer.

Both Phillips and Archuleta eventually left DLM (probably because the work there dried up too) and started their own firm in February 1984 called Video Magic. Video Magic also came to an end in too short a time, I suspect because it was becoming painfully obvious that one could not make a living trying to write software for the 99/4A.

At Texas Instruments Michael Archuleta was responsible for the 99/4A Technical Hotline and for 99/4A software quality assurance. Phillips was a third-party software development consultant and programmer in the education/entertainment section of the Consumer Products Division. Both men would get together again in 1986 to collaborate on the 4A FLYER game cartridge that was commissioned by Triton Products. To date, that is the last time we've heard from the John Phillips/Michael Archulata team.

Archuleta and Phillips were involved in, or responsible for such TI-99 favorites as:

ANGLER DANGLER - Phillips worked on this project as the debugger of the final code, but the project never reached completion before the bailout so Angler Dangler was never officially released. It does exist in GRAM file format however, so it probably was not too far from being a real product when someone at TI made the decision to pull the plug. If you look at the October 23, 1983 IUG price list you will see Angler Dangler listed as being available.

BEYOND PARSEC - This cartridge, which Bill Moseid's DataBiotics firm released for the 99/4A during the third quarter of 1988, started life in early 1984 as one of two game cartridges John Phillips was writing for CorComp's new CCI-99/64 (aka Phoenix) computer. The other game was Star Wars. Both efforts came to a screaching halt however, when TI objected to the use of the Parsec name, and George Lucas' company apparently objected to the use of the trademarked Star Wars name. The Star Wars code must have actually been finished at the time though, because I have the game on disk as a GPL file. It was ultimately renamed Star Trap and released in cartridge form by Exceltec in 1985 and then by DataBiotics during the third quarter of 1988.

BEYOND SPACE - This is a John Phillips creation that was completed in May 1984, but not released until the first quarter of 1985 when Exceltec/Sunware marketed it. It was picked up by Unisource Electronics for their catalog/encyclopedia but pretty much floundered and then just disappeared. It has never resurfaced since both Unisource and Sunware went out of business in 1986.

The game involved two players with each having a ship of equal firing power. The area in space where the two ships confront each other is littered with asteroids which may be moved by firing the ship's laser. The object of the game was to push asteriods into your opponent's space ship to crush and destroy it. The only review I've ever seen written on the program claimed that its speed was too fast to play the game very long, so that may be why it has slipped into oblivion?

BURGERTIME - Phillips provided the final debugging for Burgertime.

D STATION - This John Phillips creation has the distinction of being the only program ever released by the International 99/4 User Group on the Romox ECPC cartridge. You may recall

that during the fourth quarter of 1983, Charles LaFara promised "a library" of programs from the IUG on the Romox ECPC (Edge Connector Programmable Cartridge). D Station was just the first, but it also turned out to be the last.

When the IUG ECPC library failed Exceltec (aka Sunware) picked up the program and marketed it for a short time in 1985. Triton finally introduced D Station in their Fall 1988 catalog along with a brand new D Station II game, also written by John Phillips.

D STATION II - See D Station.

FACEMAKER - Phillips collaborated with Intersoft's Jerry Spacek on this project. Spacek you may recall wrote Defend the Cities, which was the first commercial Mini-Memory assembly language game ever written. In the Facemaker project Spacek translated Spinnaker's source code to TMS9900 assembly language and Phillips ported it to cartridge format.

HOPPER - Michael Archuleta and John Phillips co-wrote Hopper, which was the only cartridge developed entirely on the TI-99/4A Home Computer, using the Editor/Assembler cartridge for all of the programming. All of the other TI-99 cartridge software programs were developed on a TI Mini, not the 99/4 or 4A.

JAWBREAKER II - Phillips converted the original Sierra On-Line source code to TI-99/4A code.

MINI MEMORY'S LINE-BY-LINE ASSEMBLER - Phillips claims responsibility for its development, but I am not sure exactly what that means.

MOONMINE - Programmed by John Phillips from a design by Bob Hendren. You may remember that Hendren was also the project engineer behind Parsec and the person who recruited Aubree Anderson to do the voice for the Parsec game.

PETER PAN'S SPACE ODYSSEY - Phillips and Archuleta collaborated on this program while employed at DLM. It was never officially released but is available as a GRAM file that can be run from P-Gram, Gramulator or the GramKracker.

SLYMOIDS - Slymoids was written by James R. Von Ehr II. The cartridge conversion was accomplished by John Phillips.

STAR TRAP - See Beyond Parsec.

SUPER DEMON ATTACK - Phillips worked on this project, but I have no information on the specific contributions he made to its completion other than possible debugging of the final code. I do know that he actually worked on Demon Attack, not Super Demon Attack, but they are probably the same project with the actual marketed product just having a slightly different name.

THE GREAT WORD RACE- John Phillips authored.

TREASURE ISLAND - Phillips provided the final debugging for this game cartridge, which had apparently become stalled by a bug that no one could find.

WORD RADAR - John Phillips authored.

=eof=
Feb 1993

PLATO COURSEWARE

---------- Bill Gaskill

Plato Courseware is Computer-Aided-Instruction software which was first developed in January 1978 at the University of Illinois by Control Data Corporation. It was originally sold in commercial form for use on main frame computers at colleges and universities, but when the market for personal computer started heating up CDC rewrote much of the software for the smaller computer because of its much larger market potential.

The first personal computer to have a Plato program written for it was the Apple II Plus followed by the Apple IIe. The earliest announcement I can find for TI-99/4A Plato products appears in the July 5, 1982 issue of InfoWorld on page 29. A feature article on Plato also appears in Volume 1, Number 6 of 99er Magazine, which would have been published around the same time, Summer 1982. A little known item of information about the TI/CDC agreement to produce Plato Courseware for the TI-99/4A was that TI agreed to allow Control Data Corporation to put their name on a Peripheral Expansion System produced by Texas Instruments, so that Control Data could market it as their Education WorkStation. Although I have never seen a CDC EWS (their name for it), TexComp's Jerry Price, who has been in the TI-99 business longer than anyone still around, has and he remembers then being marketed for about $2500!

After reading through all of the magazines I have at my disposal, I am reasonably certain that Plato Courseware did not actually become available for the TI-99/4A until December 1982 or even early 1983. The first real attention paid to Plato in the personal computer literature didn't start appearing until April 1983 when Computers and Electronics ran a newsbyte about it on page 35 and Popular Computing ran a feature article on Plato on page 151. Control Data advertising for Plato seems to have been heaviest from late 1983 to early 1984, when it all but stopped?

According to Mike Wright's TI-Cyc book, Plato Courseware was divided into eight curricula:

1-Basic Skills Math
2-Basic Skills Reading
3-Basic Skills Grammar
4-High School Skills Writing
5-High School Skills Math
6-High School Skills Reading

7-High School Skills Science
8-High School Skills Social Studies

A curriculum was divided into Subjects, a Subject consisted of one or more Packages, a Package consisted of one or more Diskettes and a Diskette contained a menu of topics.

Plato categories that would run on the 99/4A were produced both by TI and by Control Data. All Texas Instruments produced diskettes were "flippy" disks, and all those produced by Control Data were the regular SS/SD disks. Control Data also offered an little known "Microcomputer Author's Guide" designed for educators who wished to produce their own Plato Courseware. Jon Creviston of the Dallas TIHCG has actually seen one of the Authoring Guide kits produced for the 99/4A and tried to purchase it some years back, but with no success. Programs produced under the "Microcomputer Author's Guide" program could be distributed royalty free and remained the property of the person who created it.

According to my Fall-Winter 1984 Plato catalog, which was released in September 1984, there were 199 actual titles which were offered as part of the Plato Courseware product. TI-Cyc says that an estimated 503 diskette "sides" were produced in the Plato series.

Plato Courseware was written for the Apple II Plus, Apple IIe, Atari 800, Control Data Corporation Education Worskstation, Commodore 64, IBM PC, and the TI-99/4A. With the exception of the Education Workstation, the TI-99/4A was the runaway-winner when it came to having the most categories/courses written for it.

Apple II Plus	- 78 courses
Apple IIe	- 78 courses
Atari 800	- 9 courses
CDC EWS	- 116 courses
Commodore 64	4 courses
IBM PC	- 66 courses
TI-99/4A	- 125 courses

Texas Instruments produced Plato products from number PHD 5201 to number PHD 5308. Control Data Corporation also produced a number of titles for the 99/4A that were not part of the TI inventory of Plato products. These included titles in areas of Foreign Language, Computer Literacy and Mathematics.

PLATO PROGRAMS
FALL-WINTER 1984

TITLE	CDC NMBR	TI NMBR
NUMBERS 0-9	15205575	PHD 5201
NUMBERS 10-1000	15205595	PHD 5202
ADDITION 1	15205655	PHD 5203
ADDITION 2	15205657	PHD 5204
SUBTRACTION:BASIC CONCEPTS	15205683	PHD 5205
SUBTRACTION SKILLS	15205700	PHD 5206

MULTIPLICATION:BASIC CONCEPTS	15205763	PHD 5207
MULTIPLICATION SKILLS 1	15205765	PHD 5208
MULTIPLICATION SKILLS 2	15205828	PHD 5209
DIVISION:BASIC CONCEPTS	15205830	PHD 5210
DIVISION SKILLS 1	15205847	PHD 5211
DIVISION SKILLS 2	15205873	PHD 5212
FRACTIONS:TERMINOLOGY AND CONCEPTS	15205945	PHD 5213
FRACTIONS:ADDITION AND SUBTRACTION	15205993	PHD 5214
FRACTIONS:MULTIPLICATION AND DIVISION	15207271	PHD 5215
DECIMALS:TERMINOLOGY AND CONCEPTS	15207319	PHD 5216
RATIO, PROPORTION AND PERCENT	15207321	PHD 5217
GEOMETRY-BASIC CONCEPTS	15207347	PHD 5218
MEASUREMENT	15207407	PHD 5219
BASIC WORD BUILDING	15207409	PHD 5220
MORE BASIC WORD BUILDING	15207426	PHD 5221
COMPLEX WORD BUILDING	15207446	PHD 5222
PREFIXES, SUFFIXES AND COMPOUND WORDS	15207509	PHD 5223
MORE PREFIXES AND SUFFIXES	15207511	PHD 5224
PREFIXES AND SUFFIXES IN CONTEXT	15207534	PHD 5225
SELECTING THE PROPER WORDS	15207554	PHD 5226
CHOOSING THE PROPER WORDS	15207580	PHD 5227
DEALING WITH CONFUSING WORDS	15207603	PHD 5228
WORD MEANINGS	15207623	PHD 5229
APPLYING NEW WORDS	15207643	PHD 5230
UNDERSTANDING AND USING NEW WORDS	15207660	PHD 5231
LOCATING BASIC FACTS	15207683	PHD 5232
MORE BASIC FACTS FROM READING	15207700	PHD 5233
UNDERSTANING WHAT YOU READ	15207717	PHD 5234
REMEMBERING WHAT YOU READ	15207737	PHD 5235
REMEMBERING MORE OF WHAT YOU READ	15207800	PHD 5236
INTERPRETING WHAT YOU READ	15207802	PHD 5237
UNDERSTANDING BASIC FACTS	15207819	PHD 5238
UNDERSTANDING MORE OF WHAT YOU READ	15207839	PHD 5239
DESCRIBERS AND CONCLUSIONS	15207856	PHD 5240
DIFFERENT TYPES OF DESCRIBERS AND CONCLUSIONS	15207873	PHD 5241
UNDERSTANDING THE WHOLE STORY	15207893	PHD 5242
FACT AND NON-FACT	15207910	PHD 5243
AUTHOR'S PURPOSE	15207921	PHD 5244
EVALUATING WHAT YOU READ	15207938	PHD 5245
AUTHOR'S PURPOSE AND YOUR CONCLUSION	15207949	PHD 5246
SEPARATING FACT FROM OPINION	15207966	PHD 5247
NOUNS AND VERBS	15204600	PHD 5248
MORE ABOUT NOUNS AND VERBS	15204617	PHD 5249
PRONOUNS	15204634	PHD 5250
MAKING NOUNS AND PRONOUNS AGREE	15204654	PHD 5251
ADJECTIVES AND ADVERBS	15204668	PHD 5252
PREPOSITIONS, CONJUNCTIONS AND ARTICLES	15204688	PHD 5253
LEARNING ABOUT SENTENCES	15204702	PHD 5254
PHRASES AND CLAUSES	15204716	PHD 5255
SUBJECT AND VERB AGREEMENT	15204733	PHD 5256
MORE ABOUT SENTENCES	15204750	PHD 5257
MORE THAN ONE	15204764	PHD 5258
WORD CONFUSION	15204778	PHD 5259
POSSESSIVES	15204795	PHD 5260
CAPITAL LETTERS	15204809	PHD 5261
PUNCTUATION	15204826	PHD 5262
MAKING LETTERS LOOK RIGHT	15204843	PHD 5263

GIVING THE EMPLOYER CORRECT INFORMATION	15204857	PHD 5264
SPELLING	15208506	PHD 5265
PUNCTUATION	15208523	PHD 5266
GRAMMAR 1	15208543	PHD 5267
GRAMMAR 2	15208566	PHD 5268
GRAMMAR 3	15208592	PHD 5269
DICTION	15208609	PHD 5270
SENTENCE STRUCTURE	15208629	PHD 5271
LOGIC AND ORGANIZATION	15208649	PHD 5272
BASIC NUMBER IDEAS 1	15205250	PHD 5273
BASIC NUMBER IDEAS 2	15205267	PHD 5274
MATH SENTENCES IN ONE VARIABLE 1	15205287	PHD 5275
MATH SENTENCES IN ONE VARIABLE 2	15205304	PHD 5276
MATH SENTENCES IN TWO VARIABLES	15205321	PHD 5277
GEOMETRY	15205338	PHD 5278
MEASUREMENT	15205358	PHD 5279
SPECIAL TOPICS	15205378	PHD 5280
PRACTICAL READING 1	15205400	PHD 5281
PRACTICAL READING 2	15205423	PHD 5282
GENERAL READING 1	15205443	PHD 5283
GENERAL READING 2	15205463	PHD 5284
PROSE LITERATURE 1	15205486	PHD 5285
PROSE LITERATURE 2	15205500	PHD 5286
PROSE LITERATURE 3	15205520	PHD 5287
POETRY	15205537	PHD 5288
DRAMA	15205557	PHD 5289
PHYSICS 1	15208860	PHD 5290
PHYSICS 2	15208923	PHD 5291
CHEMISTRY	15208983	PHD 5292
EARTH SCIENCE 1	15209031	PHD 5293
EARTH SCIENCE 2	15209033	PHD 5294
BIOLOGY 1	15209091	PHD 5295
BIOLOGY 2	15209130	PHD 5296
BIOLOGY 3	15209172	PHD 5297
BIOLOGY 4	15209220	PHD 5298
GEOGRAPHY	15208669	PHD 5299
ECONOMICS 1	15208695	PHD 5300
ECONOMICS 2	15208715	PHD 5301
BEHAVIORAL SCIENCE 1	15208735	PHD 5302
BEHAVIORAL SCIENCE 2	15208752	PHD 5303
POLITICAL SCIENCE 1	15208769	PHD 5304
POLITICAL SYSTEMS 2	15208769	PHD 5305
HISTORY 1	15208809	PHD 5306
HISTORY 2	15208826	PHD 5307
HISTORY 3	15208843	PHD 5308
BASIC NUMBER FACTS	15206000	
FRACTIONS	15206010	
DECIMALS	15206020	
WHOLE NUMBERS	15206030	
FRENCH VOCABULARY BUILDER	15206040	
SPANISH VOCABULARY BUILDER	15206050	
GERMAN VOCABULARY BUILDER	15206060	
COMPUTER LITERACY:INTRODUCTION	15206070	
PHYSICS:ELEMENTARY MECHANICS	15206080	

NEWEST TI

from HUGers 9/92

TI model 6600 called the ORGANIZER has 64k of ROM, contains a calendar, memo pad, schedule maker, world time database, calculator, metric/US and monetary converter.

Now, the real kicker -- the "UPLINK KIT" enables you to send your information to any PC. Guess what else?? It can also send to the TI-99/4A, because the output is ASCII and the uplink is the standard RS232. with the correct cables, it can be hocked to the TI-99/4A COMPUTER. Can be found at RADIO SHACK as model #EC 332 and sells for $129. The interface cables cost $5.95 model #65-133.

Check it out.

MIDI-MASTERS

Micropendium, 9/92

Delores P. Werths, musician for Harrison Software, and Jim Peterson of Tigercub Software are trying to organize a by-mail users group for persons making music with MIDI master.

The group plans member exchange of SNF source file disks, a central clearing-house / library and a disk newsletter.

Write Werths at; 5705 40th Place
Hyattsville, Md. 20781

ALGORITHM DESIGN

by Garry Christensen
from the TIsHUG, Brisbane U. G., Australia, Sep. 1992

We will start with a definition:

ALGORITHM - a systematic procedure for solving a problem or accomplishing some end.

Programming some tasks on the computer is easy. Many short programmes can be written quickly and with little fore-thought. This is not always the case. The more complicated the problem, the more involved the solution. There comes a stage when the programmer must sit down and work out how he well approach the problem and how the programme will work.

An ALGORITHM is usually the result of this work. The algorithm is probably the most important part of any programme yet it is so often ignored.

Once the algorithm has been written, the programme is much easier to write and frequently will have fewer bugs. That does not mean that algorithms only apply to computing. An algorithm can apply to any situation where a result is required. Condsider the algorithm for finding the partner of your dreams using a dating service:

Send money.
Wait for reply from dating service.
Contact date.
Go out with selected partner.
Repeat untill suitable partner is found.

This process is far from perfect as you may go broke before getting married but the algorithm still exists.

The first part of algorithm design is the analysis of the problem. In simple cases, the answer is obvious but sometimes it may require some thought to determine the solution. Consider also that the solution may not be unique. There may be other ways of doing the same thing. Before even touching the keyboard, try to determine that the method you have chosen is the best.

The second part is to determine the steps that will lead to the solution. This part involves a method called top-down development or step wise refinement.

Start with the steps written in very broad terms (generally using English) then refine each step into a series of more detailed steps. Each of these steps can then be further refined, and so on utill the steps are close enough to the programming language to allow easy conversion. Do not start to write programme code in your algorithm untill the very last step.

Let's consider the algorithm necessary to instruct a robot to make a cup of coffee. The solution is simple so lets define it in broad terms.

1 Boil the water.
2 Put coffee in the cup.

3 Put water in the cup.

These steps are obviously too complex for a robot that has no experience in the human world so they need to be defined further.

```
1 Boil the water.
 1.1 Fill the kettle.
 1.2 Plug in the power.
 1.3 Turn on the power.
 1.4 Wait untill the water has boiled.
 1.5 Turn off the power.
```

Refine the steps still further.

```
1 Boil the water.
 1.1 Fill the kettle
  1.1.1 Place kettle under tap.
  1.1.2 Turn on the tap.
  1.1.3 Wait untill the kettle is full.
  1.1.4 Turn off the tap.
```

Continue through all the steps using this method untill a full list of detailed instructions is produced.

As you can see, this is an easier way to end up with detailed instructions than simply starting at the top and working straight through. This method of refinement can continue untill the instructions are something that the robot can perform. These are called primitive operations. In computer programming, the primitive operations are the statements that make up the language.

There are two types of traps that I should point out at this stage. The first is called the "first catch your rabbit bugs". We have so far assumed that there is a kettle for the robot to use and that there is a tap available. Nowhere have we told the robot to "find a kettle" or "go to the tap".

The second trap is called the "initial state of things bugs". What will the robot do if the water is already turned on. When told to "turn on the tap" it may screw it out of the wall.

The secret to writing an algorithm is to keep the following in mind:

1. Start at high levels (use English to express very broad steps).
2. Concentrate on refining one step at a time.
3. Do not start writing programme code untill the steps are approaching the level of the programming statements.

Let's now work through a couple of examples.

I will use the selection of all the prime numbers between 1 and 100 as an example. (A prime number cannot be evenly divided by any number other than itself, eg 7). Problem: How do I determine whether a number is prime.

The simplist answer is to divide it by all the numbers less than the number in question and see if any produce a whole

result. That is a valid method but not very efficient.

If the number is 50, dividing it by any number greater than 25 cannot produce a whole result (50 / 26 = 1.something). If the number is called N, we need only divide by all the numbers up to N/2.

To test all the numbers from 1 to 100, we need to test them all in this method. The algorithm development follows:

```
1 Set N to 1.
2 Is it prime?
3 If so then print it.
4 Add 1 to N.
5 Go to step 2 if N is not greater than 100.
```

Steps 1, 4, and 5 are very close to a basic statement so the next step of refinement will convert them to code. The others need a little work.

```
1 FOR N=1 TO 100
2 Is N prime ?
 2.1 Set D to 2
 2.2 Is N/2 a whole number?
 2.3 If so, go to step 4 (not prime).
 2.4 If not, add 1 to D.
 2.5 If D is less than half of N, then go to step 2.2.
3 PRINT N (this step is jumped over if not prime).
4 NEXT N
```

Further refinement is not needed.

```
1 FOR N=1 TO 100
2 Is N prime?
 2.1 FOR D=2 to N/2
 2.2 IF N/D = INT(N/D) THEN step 4
 2.4 NEXT D
3 PRINT N
4 NEXT N
```

The program:

```
100 FOR N=1 TO 100
110 FOR D=2 TO N/2
120 IF N/2 = INT(N/2) THEN 150
130 NEXT D
140 PRINT N
150 NEXT N
```

While this algorithm demonstrates the best way to build up an algorithm, it fails in one point. The approach to the problem is not necessarily the best. Stop reading here and try to find a more efficient way to find prime numbers.

Did you think of one? Follow the next algorithm through. You will notice that I have used indenting to seperate the levels of detail. I you want only a brief outline, read the parts of the algorithm that are not indented. The further indented, the more detailed.

REM This method sets up an array of flags for each number between 1 and 100.

```
1 Initialize the array.
 1.1 Diminsion array.
 1.2 Set array pointer to 1
 1.3 Set array element to 0
 1.4 Increment the pointer
 1.5 Repeat untill pointer is greater than 100
2 Remove the non-prime numbers.
 2.1 Set N to 2
 2.2 Remove all multiples of N up to 100
  2.2.1 Set pointer to value of N.
  2.2.2 Check that this value has not been removed by a
     previous pass.
  2.2.3 If it has then go to step 2.4
  2.2.4 Add N to value of pointer.
  2.2.5 Set array element to 1.
  2.2.6 Go to 2.2.4 and repeat untill pointer is greater
     than 100.
 2.3 Increment N.
 2.4 Repeat untill N is greater than 50 (100/2).
3 Print the remaining numbers.
 3.1 Set poiinter to 1
 3.2 Does the flag indicate that the number is prime?
 3.3 If not then goto step 3.5
 3.4 Print the value of the pointer.
 3.5 Increment the pointer.
 3.6 Repeat untill end of array.
```

The programme will look like this.

```
100 DIM FLAGS(100)
110 FOR I=1 O 100
120 FLAGS(I)=0
130 NEXT I 140 FOR N=2 TO 50
150 IF FLAGS(N)=1 THEN 190
160 FOR I=2*N TO 100 STEP N
170 FLAGS(I)=1
180 NEXT I
190 NEXT N
200 FOR I=1 TO 100
210 IF FLAGS(I)=1 THEN 240
220 PRINT I
230 NEIT I
```

As you can see, the algorithm is easy to follw and you will have to believe me when I say that it made the programming much easier. I tried it first without an algorithm and the result was both longer and less efficient.

It seems that the computer science proverb is correct: 'The sooner you start coding, the longer it will take to write a programme that is correct'.

To give you a bit of practice at algorithm design, I have included several problems. Write algorithms for the following problems. One way to check the answer is to continue on and write the program. You might try to write the programme first, then solve it using step-wise refinement. Compare the answers. Are they the same?

1. Write an algorithm for placing a phone call.
2. Include in the algorithm for question 1 the possibility of busy signal, no dial tone and no answer.
3. Write an algorithm to test if a string is a palindrome. (A palindrome is a word is a word or sentence that reads the same forward as backward, ignoring the spaces. eg RADAR or A MAN A PLAN A CANAL PANAMA).
4. Devise an algorithm for a programme that will ask for numbers to be input untill the value 0 is entered, then print the largest number that was input.
5. Write an algorithm for a programme that will ask for an input of a single positive number then print the number with the order of the digits revesed. eg 13542 results in 24531.

O

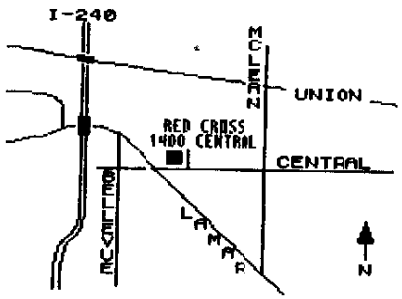
PLAYING CHARGE IN XBASIC

---------- Earl Raguse
Lima Ohio User Group

from the pages of the Ozark 99'er NEWS, October, 1992

For one of our first lessons in coding music in XB, I have chosen the attached program called CHARGE, it plays the bugle call of the same name. Notice, that it uses the stardard CALL SOUND statement for each note using the frequency expersed as a number. This method is rather tedious for a larger program, but for this one it is releatively efficient. I have used a variable L for the note length, because it is used many times. Notice that I used a delay loop to do a rest in line 180. Rests can also be done using CALL SOUND(R, 30000, 30) where R is the rest period. As an aside, a CALL SOUND statement can be used instead of a delay loop, in non-sound programs, provided the delay is not greater than about 4.25 seconds.

```
100 ! SAVE DSK1.CHARGE
110 ! By Earl Raguse 4/12/88
120 DISPLAY AT(12,11)ERASE ALL:"CHARGE"
130 L=125
140 CALL SOUND(L,1047,0)
150 CALL SOUND(L,1175,0)
160 CALL SOUND(L,1319,0)
170 CALL SOUND(L,1568,0)
180 FOR T=1 TO 125 :: NEXT T
190 CALL SOUND(L*2.5,1319,0)
200 CALL SOUND(L*6,1568,0)
210 END
```



LOCATION MAP

WORKSHOP : to be announced.

PROGRAM BIT - third Thursday

APRIL 15th , 1993

MEETING: 7:00pm - Red Cross Building - 1400 Central.

6:45pm - Doors Open

7:00pm - Meeting begins, general discussion.

7:30pm - Demonstration to be announced.

9:00pm - Meeting ends.

9:15pm - Late dinner at location to be announced at meeting.

NOTICE

Information contained in TIdbits is accurate and true to the best of our knowledge. Viewpoints and opinions expressed in TIdbits are not necessarily that of the Mid-South 99'ers. We welcome any opinions/corrections from our readers. Articles may be reprinted elsewhere as long as credit is given to the author and newsletter.

GROUP INFO

Visitors and potential members may receive 2 free issues of TIdbits while they decide if they wish to join (no obligation) On the top of your label is a code. A Y means you are a member, N means 2 free list, UG means user group and B means a business. Beside the Y is a date, one year from that date your dues are due. A dollar sign ($) on the label will indicate that your dues are due. The library is open only to members. Library list is $1. Mail order disk library access is $2 for the first disk and $1 for each additional disk - max of 5 disks per month. Order by disk number only. At meetings, library access is FREE if you exchange your disk for ours or $1 per disk for our disks. Send all mail order library requests to librarian's address! Send dues and correspondence to group address.

CALENDAR

MEETINGS: APR. 15, (3rd Thursday!)
WORKSHOPS: TO BE ANNOUNCED

24HR TI BULLETIN BOARD

The 9640 NEWS BBS 300/1200/2400/4800/7200/9600/12000/14400
Hayes. 901-368-0112

GROUP MAILING ADDRESS

Mid-South 99 Users Group
P.O. Box 38522
Germantown, Tn. 38183-0522

LIBRARY ADDRESS

Jim Saemenes
46 Higgins Road
Brighton, Tn., 38011

MEMBERSHIP APPLICATION

NAME _____ |_| $18.00 FAMILY
ADDRESS_____
CITY_____ST____ZIP_____
PHONE(____)___-_____:INTERESTS_____
_____
EQUIPMENT,ETC._____
_____
Detach and mail with check payable to: Mid-South 99 Users Group, P.O. Box 38522, Germantown, Tn, 38183-0522.