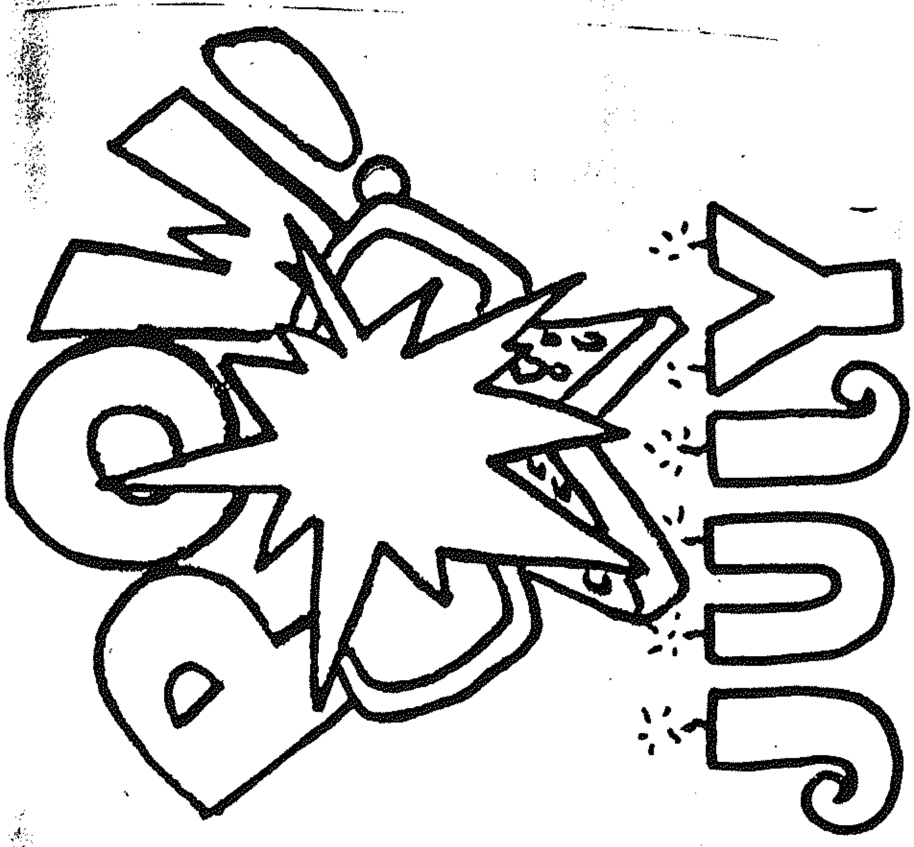
Mass Users of the Ninety-nine and Computer Hobbyists

JULY 1996 Monthly Newsletter Version 15.07



U.N.C.H.
O J. W. COX
5 EGDEBROOK DRIVE
YLSTON, MASS. 01505

XT MEETING: TUESDAY, JULY 9th.

STMASTER: Forwarding and Address Correction Requested.

FIRST CLASS!!

NEXT MEETING TUESDAY, July 9, 1996 7:00 PM. Happy Father's Day
OFFICERS AND NUMBERS (all in 508 area unless noted)

PRESIDENT	Walt Nowak	413-436-7675		
VP./Treas./Editor	Jim Cox	869-2704	MUNCH DUES:	
DEMO LEADERS:	Corson Wyman	865-1213	New Membership	$25.00
	Jack Sughrue	476-7630	Renewal	$15.00
CLERK	Ben Parda	791-9172	Newsletter Sub.	$13.00
Advanced Programmer	Dan Rogers	248-5502		

****

JUNE MEETING. There were six members and a guest at the June meeting. Gray gave a report on the Ohio Fair, there were approximately 90 attenders. We continued our talks of the Internet and E mail, and how they can be used with the T.I.

JULY MEETING. We will continue our discussions on what is new in the T.I. world. We will also cover anything new in the computer area.

HELP WANTED. The following disks in the Tigercub Collection are mission from our set, if you have them please send them to Jim Cox or bring them to a meeting. They are 603, 1079, 1260, 1261, 1309.2, 1382.2, 1472 and 1520.1. The following have read errors 1157, 1297, 1411, 1423, 1436, 1439.1, 1474 and 1487.

RAFFLE. Occasionally we have a raffle to help defer the rental cost of our meeting hall, it depends on the number present.

REPRINTS. Reprints are permitted as long as credit is given to M.U.N.C.H.

ARTICLES. I am always looking for articles for this newsletter, anything which interest you will probably interest other members of the T.I. community, so please share your ideas and opinions with all of us.

DISK LIBRARY. The disk library is at all meetings. We have copies of all disks in the library and they are available to members for just $1.00 for each disk unless otherwise specified. You can order them through the mail, please add $1.00 for the first disk and $.40 for each additional disk ordered to cover postage and handling.

DISK OF THE MONTH. I hope to have a Wheel of Fortune Type game for the DOM.

ADVENTURE II. This is our fund-raiser for now. The cost to members is $4.00, add $2.00 for first class postage. The regular price is $6.95 plus postage. This is a two DSSD disk set, archived. There is also a special on The Adventure Compendium and Adventure II for members it is $8.00 plus $3.00 for first class postage.

FOR SALE: Al Eisenhower of Hyannis, Mass. has T.I. equipment for sale. He is especially interested in trading T.I. stuff for American Flyer trains or other trains. Call him after 5:30 p.m. EST at 508-775-4289.

SORRY. This month's issue has only six pages due to time constraints.

This is an Include File option. Let me give an example;

```
Program Sample;
  Const:______;
  Type:_______;
  Var:________;
{Procedure/Function Section}
{$I #5:Hello}
{$I #5:Howareyou}
{$I #5:Iamfine}
{$I #5:Goodby}
{End of Procedure/Function section}

Begin {Main program}
  Statements;
End.
```

Assume that the above Include File options are procedures and/or functions in the correct place in the program and we have the following files on the #5 drive; Hello.Text, Howareyou.Text, Iamfine.Text, Goodbye.Text and the main program as Sample.Text. Now call the Compiler and when the Compiler calls for the source file enter "#5:Sample". The Compiler will start and when it gets to the first Include File directive it will go to the #5 drive get that file compile it and then get the second file compile it etc. until it finishes. Then the entire program will be made into object code.

The advantage of this option is that it allows you to break the program into small logical parts the sum of which could be larger than the Editor's memory. (The Editor's memory is 13K.) Further the Include File directives can be nested up to three levels deep!

Notice in the program that the Include File directives are not suffixed with .Text. That is because the compiler first looks for the files with a .Code suffixed to their names and if it does not find that file it will look for a file with .Text as a suffix. If you want you can add Sample program.

If you want to have some procedures or functions completely written out as part of the program and not have them as Include File directives then these procedures/functions have to be "Forward" declared before you start the Include File directives. (See a text about Forward).

* * * * * * *

EDITOR'S NOTES - - -

As I promised last month, I have started printing the Pascal tutorial by Stan Katzman. I will complete it next month with my final <u>Geneve Support Article</u>.

* * * * * * *

Bruce's Latest Gift

Last month I revieweda program from Bruce Harrison called Word Search. This month I would like to talk about Bruce's latest figt to the TI community, Font Designer. This is a brand-new public-domain program that allows you to modify an existing font, or to create a completely new font, and then to dump it to your 24-pin or bubble jet printer.

As always, the program is easy to use, is lightening fast, and does exactly what is says it is going to do. The documentation is a model of clarity. I cannot emphasize enough how important that is to dummies like me!

This program lets you use the entire 24x36 pixel character grid. It allows characters to be designed and downloaded over the printable range from 33 to 126. You are not required to redesign every character in a font; those you choose to leave alone remain in their original state.

Font Designer is menu-driven. The first menu offers you a choice of seven options: load a character set, save a character set, edit a character, dump to a printer, delete a character, purge font memory, and exit the program. If you choose to load a character set, the program does so, and then returns to the menu. You can then choose to edit a character, and the program asks you which one. If you decide that you have really messed a character up you can throw it away and return to the original with just a key press. A small box on the left on the screen shows you how the character will look when printed. Of course, while you are designing the character you have a full screen grid where you can toggle the pixels on and off to allow for creation of really elaborate fonts. Once you have a font you want to dump to your printer, you can do so easily by using another program on the disk called QDUMP. Bruce even includes a number of fonts on the disk for those of us too lazy to design our own. I may use a couple of these in the printing of this review to show you what is available.

This program will be available from our library, and on our BBS. It is not only useful, but a lot of fun to just play around with. A note of thanks to the author, and even a couple of bucks would probably not be unwelcome. By the way, Bruce hopes to be at the Chicago TI International World Faire again this year. That's very good news, indeed!

--Hal Shanafield

This is Gothic.
This is Drop.
This is Bijou.
This is Fancy.

More next time.

* * * * * * *

PASCAL/p-CODE PART 9

Stan Katzman

The next topic I would like to talk about is printing ut with a program to a printer. The programming process is imilar to the disk reads and writes.

Below is a program that will print out to a printer.

```
Program Print(input,output,p);
Var
  p:Interactive;
Begin
  Reset(p,'Printer:');
  Writeln(p,'This line will go the the printer');
  Writeln('This line will go to the screen');
  Page(output);{This will clear the screen}
  Page(p);{This will cause the printer to form feed}
  Close(p);
End.
```

In the "Var" section "p" is identified as an Interactive ile. An interactive file is in U.C.S.D. only, it is for ardware such as disk drives, printers and modems.

The printer file is opened with a "Reset(p,'Printer:');" tatement and those lines that you want to go to the printer re "Writeln(p,-------);". Naturally we could have variables n place of the the strings and the values of the variables ould print out.

This entire discussion is prefaced on the fact that the S232 is modified for parallel output in order to access a arallel printer. This would have to be done before any rogram is run using a parallel printer.

When you are done with the printer, its file should be losed. You do not have to use the term "Lock" in this case. his is good programming practice because if you did not lose the file in a regular program and then came back to the rinter file and reopened it the operating system would give ou an error message and stop the program.

I have some room in this issue so I would like to iscuss a programming technique that is crucial. If you look t the disk program in the last issue you will see in the iskmake procedure a For..Do loop. That loop has a egin..End pair. This is very common is Pascal, Begin..End airs group statement lines together. You can also have ested Begin..End pairs and it is important that for every egin there is an End. Further the Begin..End of one pair annot cross another Begin..End pair.

For example;

Correct

```
For N:=1 to 10 Do
```

```
  Begin
   Statement;
   Statement;
    Begin
     Statement;
     Statement;
    End;
  End;
```

Incorrect

```
For N:=1 to 10 Do
  Begin
   Statement;
    Begin
  End;
     Statement;
    End;
```

Also notice that there are no semicolons after the Do loops nor after the Begins.

When I write a program out longhand I keep track of the Begin..End pairs with lines like I have drawn above. This is one of the reasons Pascal source codes are indented, so as to keep track of Begin..End pairs.

More next time.

* * * * * * *

PASCAL/p-CODE PART 10

Stan Katzman

Let's discuss Compiler options. These are directions to the Compiler that control the Compiler's output. I do not know what all of these options do so therefore I will only discuss those that I use.

The options are put into the source code as 'psuedo-comments'. A psuedo-comment is written {$_}.

{$L+} is a listing comment. This option would be the first line of a program. This option would cause the results of a program compilation to be placed on the #4 drive under the title "System.lst.Text". If there are compile time errors they will be listed. The results can be printed out using the "Transfer" option of the Filer. (Note; when the listing is printed out from this file a sheet of paper is form fed through the printer at the start. I tell you this so you will not be startled like I was.)

{$Q+} supresses the Compilers output to the consol, except for error messages. Further it does not give you the prompt "..press space bar to continue, E for Edit or Control(.) to exit". This allows the compiler to go a little faster because it does not have to write to the screen.

{$I filename}. This is one option that I heavily use.

Let us discuss the program from Part 6.

A Pascal program always starts with the reserved word Program followed by the program name. The words in parenthesis following the name are optional in U.C.S.D. Pascal but mandatory in Standard Pascal. The words input and output say that the program will accept data from the user and display out data. All Pascal programs must have a Begin...End. pair. The final statement is End. (period).

All variables must be identified ahead of time in Pascal. The machine will recognize the first eight characters of a variable. (We used three variables (or identifiers) A, B, and C.) U.C.S.C. Pascal has five simple types of values; 1)real, 2) integer, 3) character, 4) string and 5) boolean. (We can identify other types of values also.)

A real variable is a decimal or exponential. A real number has to have a digit on each side of the decimal point. If we input a real number such as "10" the machine adds the zero so it will be "10.0" internally. An integer variable cannot accept a decimal value.

A character (Char) is a single letter or number.

A string is a series of letters and or numbers.

Boolean has only two values true or false.

The statement "Page(Output);" causes the screen to clear and the display to start from the upper left of the screen.

"C:=0.0;" initilizes the variable C to zero. Notice the syntax; if one evaluates a variable in Pascal the symbol ":=" is used. The symbol "=" is used for boolean comparisons.

The procedure "Writeln" puts each line of output on a separate line. Notice the syntax for a screen output "Writeln('......');" there is a parenthesis a single quote, what you want displayed, single quote, parenthesis and finally a semicolon. Furthermore each line of a Pascal program must fit on one line in the Editor, it cannot be between two lines. "Writeln;" by itself causes a blank line to be printed to the screen. Also notice each line ends in a semicolon; there are two exceptions 1)the final "End." ends in a period and 2)the statement before and else in an 'If..then..else' statement does not have the semicolon.

The "Write" procedure prints its output without putting a carriage return at the end of the line. So this procedure keeps putting output on the same line until a "Writeln" or a "Readln" is encountered.

"Readln" stops the program and waits for input from the user. The rest of the program is obvious.

At this point I would like to give a overall format for Pascal programs.

```
Program heading
Const definitions
Type definitions
Var declarations
Procedure or Function declarations
```

```
Begin
  Statements;
End.
```

* * * * * * *

PASCAL/p-CODE PART 8

Stan Katzman

There is so much to Pascal that I doubt that I can describe it the the detail that it deserves. So I feel that you should refer to a text that describes Standard or U.C.S.D. Pascal. What I would like to do is to describe some of the programming techniques specific to the U.S.C.D. Pascal system.

If you look at the statements of Pascal you will not see any Read...Data statements like BASIC has. So how do we get data into a program? It enters in two ways 1)from the keyboard and/or 2)from a disk file. The input from the keyboard is for temporary data. The disk file is for permanent data. Attached is a Pascal program which writes and reads a disk file.

We will describe the special U.C.S.D. Pascal statements dealing in file creation.

The type of file must first be identified in the "Var" section. Here we made a "Text" file. A text file can contain letters, strings and real or integer numbers. There are also untyped files called "file" and files of integers. The most common are "Text" and "File".

The program must tell where the file will be located (we put the file on the #5 drive) and the name of the file. The name of the file was done with a "Readln(Fname)" statement. The entire file name was done using the "Concat" statement. The "Concat" is a statement only in U.C.S.D. Pascal. It is a statement that concatenates strings. The entire file name was done in the main body of the program.

In order to write to the file we opened the file for writing using the "Rewrite(f,Filename)" statement and wrote to the file with the statement "Writeln(f,Number[N]);". The "f" says to write to the Text file. When we finish the file is "Close(f,Lock);". If we do not say "Lock" the file is not recorded to disk.

In order to open a file for reading the "Reset(f, Filname);" statement is used. The file must be read as it was created, if one value is written at a time with "Writeln" then one value at a time should be read with a "Readln". When finished the file is closed with "Close(f,Lock);".

If you notice there are "{}". These are comment delimiters. Between these delimiters you can make comments about the program. What appears between the {} is ignored by the compiler.

original will return.

D(elete. If you want to remove some text enter Control C, position the cursor over the start of the material to be deleted and press "D". Move the cursor with the cursor keys and the material will be removed. To make the deletions permanent press Control C. If you want to remove entire lines press "D" and then Function 9. Move the cursor downward with the cursor key (Function X) until all the material you want removed is gone. Again, to make the deletion permanent press Control C (to retrieve the "removed" material press Control (.)).

P(age. If you have more text than will fit on one screen and you want to search the text we can use the Page command. Enter Control C and then press "P", the screen will move 24 lines at a time. Look at the upper left hand corner of the screen and you will see an arrow (< or >), the default "position" of the arrow is (>). If the arrow is pointing to the right when you press "P" you will move downward through your text. To change the arrow press the appropiate arrow key (shift comma or shift period). If you have a really large text you can press a number like 2 or 3 and then when you press "P" you will move 2 or 3-24 line blocks at a time.

You can also move through text by using the F(ind and J(ump commands. I have not used these but I want you to know of their existence. The commands that I have described here are the ones that I use and I have been able to get along very nicely in the Editor.

* * * * * * *

PASCAL/p-CODE PART 6
Stan Katzman

We have completed our discussion of the Editor and the Filer and it is time that we started writing some Pascal programs. We will start out simply and try to introduce new topics with each issue. Let me say at the outset that I am not an expert in Pascal and I would appreciate any help that I can get in the subject.

Let us write a very simple Pascal program that will show a few basic principles. This program will add two numbers and display the answer.

```
Program Add(input,output);

   Var
      A,B,C:Real;

Begin
  Page(Output);
  C:=0.0;
  Writeln('This program adds two numbers.');
  Writeln;
```

```
  Write('Enter one number ');
  Readln(A);
  Writeln;
  Write('Enter the second number ');
  Readln(B);
  C:=A + B;
  Writeln;
  Writeln('The answer is ',C);
End.
```

Go into the Editor and enter the above program. Notice the indentation, if you indent the program is easier to read. You will notice that when you indent in the Editor by using the space bar that when you press enter the cursor returns to the indented column. To move the cursor to the left use the cursor key (Fctn S).

After you finish typing the program press Control C and save the text under the "W" option as "#5:Add". Now exit the Editor.

Let us compile this program. Enter a "C" on the main command line and you will see the prompt "Compile?", at this point enter "#5:Add". Next you will see the prompt "To what codefile?", at this point enter "#5:Add". The first prompt asks for the source file (that is the file you made in the Editor) which has the suffix .TEXT added to it. Do not add the suffix .TEXT because the Compiler will add it. When the Compiler is done it will make a record of the compiled program on the #5 drive under the name Add.Code.

While the file is being compiled an account of the progress will be displayed. If an error is encountered the compiler stops and displays the following "...<sp>(continue), <esc>(terminate), E(dit". I normally correct the errors as I encounter them so press "E". The Editor will be loaded and it will call for the file name, enter the name minus the .TEXT ending. Editor will display the "offending" error. (Many times this is not where the error is but earlier in the program.) The directions at the top of the screen tell you to press the space bar to continue and after you do you can make the corrections. Enter Control C and then resave the file. Call the compiler and continue until the program compiles.

Now let us run the compiled program. In the main command line enter "X" (for eXicute) and you will see the prompt "What file", enter "#5:Add", the program will exicute and then return to the main command line. To run the program again press "U" (for User restart) and it will run again.

More next time.

* * * * * * *

PASCAL/p-CODE PART 7
Stan Katzman

The V(ols command lets you examine what "volumes" are on line. What is meant by a Volume is a disk drive, the printer, etc. If you do not have a formatted disk in the drive when the system is turned on that disk drive does not exist (it is not "on line".) So have a disk in all drives when you "boot" the system. If you forget to put a disk in the drive to start with, put one in after the system is booted and then call the V(ols from the Filer, this program will put the drive on line and allow you to use it.

Again everything discussed here is in the Filer documentation so you can read it and get a firsthand account.

More next time.

* * * * * * *

PASCAL/p-CODE PART 4
Stan Katzman

What we have discussed in the three previous articles are the Filer and one or two Utility programs. Let us now go into the Editor.

Put the disk we made containing the Compiler, Editor, Filer etc. into drive #4 (let us call this the system disk). Put a empty formatted disk into drive #5. Turn on the P-Box, then the console and then modify the printer for "PIO". Now call the Editor by entering "E". There will be a prompt that says "No workfile, File(<ret> for none)?". This prompt is asking for the name of an old file that you want to work on or if you want to create a new file press enter. Let us press enter. You will now see across the top of the screen a promptline that ends in a question mark. The question mark means that all of the selections cannot be shown and in order to see the rest of the choices press Function I or the question mark key (?). (This is true whenever the (?) shows in a promptline.) The choices that are available to use in the Editor are A(djst, C(opy, D(let, F(ind, I(nsrt, J(mp, K(ol, M(argin, P(age, Q(uit, R(plc, S(et, X(ch, and Z(ap. I do not use all of the above so I am only going to discuss those that I have used to create and edit programs.

In order to enter text in the Editor press "I" (for Insert) and then start typing. When you get to the end of an 80 column line the cursor will stop and will not advance to the next line. In order to get to the next line you have to press the enter key. You do not have to be at the end of the line in order to press enter you can do it when ever you wish.

When you finish typing either all of your document or just a section of the document, you want to save it. To do this first press Control C, (press the Control key and then press the C key.) This puts an invisible End of File marker in the document. This is necessary because the EOF tells the program to accept this text or any changes. After Control C press "Q" (quit) and you will now see this menu ;

U(pdate the workfile and leave
E(xit without updating
R(eturn to the editor, no updating
W(rite to a filename and return

"E" and "R" are more or less self explanatory .

"U" writes to a file on the #4 drive and titles it "System.Wrk.Text". If you use this option while developing your programs you will not have to do anything when you enter the Editor. The file "System.wrk.text" will be loaded automatically so you can work on it. I personally do not use this option, but it has some advantages with small programs. I use the "W" option.

If "W" is pressed you will see "Output file (<cr> to return)" at this point enter "#5:Filename" and the text is stored on the #5 disk under the file name with a suffix of .TEXT added to it. For example if you called your file "Buffer" the Editor would store it as "Buffer.Text". After the file is saved you will see "E(xit or R(eturn to editor?". If you press "E" you will go to the main command line. If you press "R" you will be back in the editor and you can position the cursor using the cursor keys and then press "I" and continue typing in your document.

If there was a document on disk that we wanted to modify when we entered the Editor, we would enter its file name where it said "No workfile, File(<ret> for none)?", the file would then be loaded. We do not have to add the suffix .TEXT because the editor will do that. When it comes time to save the document press "W", the file name is displayed and a statement saying that if we enter "$" the file will be saved under the same file name, without having to type the entire file name. This assumes that we were not using System.wrk.text.

In order to get a printed copy of our document we leave the editor and go to the Filer. After entering the Filer press "T" (transfer), and after the prompt enter "#4:Filename.Text,#6:" and your file will be printed. In this situation you must add the suffix .TEXT or you will get an error message.

Well this is another long session. More later.

* * * * * * *

PASCAL/p-CODE PART 5
Stan Katzman

This time I would like to discuss a few (not all) of the commands used to edit text in the Editor.

X(change. If we see a mistake in our text enter Control C and then put the cursor directly over the error and press "X". When you type the corrections will exchange for the characters on the screen. To make the changes permanent press Control C. If you decide that you do not want the changes made permanent press Control (.) period and the

PASCAL/p-CODE PART 2
Stan Katzman

Let us discuss how to print out data. If you have a
allel printer you cannot use the parallel port of the
232 card as a default, the default output is for a serial
nter at 9600 baud. Therefore the output must be modified
this is done by using a program on the Utilities disk
led MODRS-232. Place the Utilities disk in drive #4 and
other formatted disk in drive #5. Turn on the p-Code
d, the P-Box, and finally the console. Press "X" and at
prompt "What file?" enter "MODRS-232". At the next
play enter a "P" and at the next display enter "PIO".
s now allows you to use your parallel printer. This must
done each time the computer is started. (If somebody
ws how to incorporate this into SYSTEM.PASCAL please write
)
Let us now get a printout of a disk directory. Put the
er disk in drive #4 and press "F". (To get out of the
er and back to "main" command line press "Q".) To get a
k directory listing press either "L" (L(ist dir) or "E"
xtensive dir). "E" gives more information and is the main
that I use, so let us press "E". You will now see "Dir
ting of?", enter "#4" (or "#5" depending on which disk
ve you want a listing of) and there will be a listing on
screen. If you want a listing to the printer, at the
r listing of?" prompt enter the following "#4:,#6:" and
output will be to the printer. (The printer can be
led "#6:" or "Printer:".)
Let us now prepare a disk so we can do Pascal
gramming. Put the Filer disk in #4 and an empty,
matted disk in #5. (It is assumed everything is at least
SD.) Call the Filer by entering "F" and then the Transfer
gram by entering "T". The prompt says "Transfer?", and at
s point we are going to transfer files one at a time.
m the listing of the Filer disk we see the following
tents; System.Filer, System.Editor, System.Pascal,
tem.Syntax. Enter the following to transfer the first
e; "#4:System.Filer,#5:$" and the first file will be
nsferred. You will be returned to the command line so
ss "T" again and transfer the second file by entering at
prompt "#4:System.Editor,#5:$". Do this for the rest of
files on this disk, then remove the Filer disk and put
Compiler disk in #4. The Compiler disk has the following
es; System.Compiler, Screenops.Code, Commandio.Code,
tem.Library. You will be at the command line so press "T"
at the prompt enter "#4:System.Compiler,#5:$". Continue
s process until all the files are transferred. Take out
Compiler disk and put the Utilities disk in #4. On a
line basis we need the MODRS232.CODE file. So press "T"
return to the transfer program and then enter
:MODRS232.CODE,#5:$". You will now have all the important
s needed to make Pascal programs on one disk. Now back

that disk up at least twice. You can also print out the directories of these disks. Also read the corresponding sections in the Filer.

(Notice in the above discussion that when we transferred the file to the second disk we used as a file name "$". When the "$" is used the original file name is used in the second case also. This saves us some typing.)

Well enough this time. More later.

* * * * * * *

PASCAL/p-CODE PART 3
Stan Katzman

This time I would like to discuss some more operations of the Filer. The operations are K(runch, R(em, C(hange, B(ad blks and V(ols.

The p-Code system does not have "fractured" files like the T. I. system. A file is always placed at the end of the existing files. So if we modify a file, when it is rewritten it is placed at the end of the existing files leaving those sectors where it originally was blank. After a while this can leave a series of blank sectors which are wasted. So what has to be done is all of the existing files have to be K(runched together. In order to Krunch a disk go into the Filer and press "K". You will see the prompt "Crunch?", at this point enter "#5", next you will see "From the end of disk, Block 360? (Y/N)" enter "Y" and the disk will be crunched. If you now list the directory under the "E" heading you will see what has happened.

The R(em section removes a file from the disk. In the Filer command mode enter an "R", you will then see "Remove?" at this point type the drive number a ":" and the file name; eg., "#5:Myfile.text", you will then see "Update directory?". If you enter "Y" the file will be removed if you enter "N" the file is not removed. This is a safety feature that might prevent you from accidentally removing a file that you want.

The C(hange command allows you to change file names or disk names. To change a file name enter "C" from the Filer command line. The prompt then says "Change?", at this point enter the change; for example "#5:Go.Code,System.startup" and the file is changed. To change a disk name, at the prompt "Change?" type the old disk name, a comma, then the new disk name. For example "Mydisk12:,Chemdsk:".

The program B(ad blks searches out bad blocks on a newly formatted disk. Enter a "B" from the Filer command line and you will see the prompt "Bad block scan of?", enter "#5". The Filer then prompts "Scan for 180 blocks? (Y/N)", enter "N" and the Filer prompts "Scan for how many blocks?", enter 360. The disk is scanned and if any bad blocks are found it will be printed out. If bad blocks are found they are isolated with the X(amine command. (Ihave yet to find any bad blocks.)

These articles were written by Stanley Katzman, 1142 Skyline Dr., Greensburg, Pa. 15601.

If you can add anything to this body of information please write to me so that we can share information.

If anyone knows of any commercial programs for the p-System please write and tell me what they are, where they can be purchased and what the cost is. T. I. sold one program, that I know of, and that was a income tax program.

I have made a few programs for my own usage which I would be happy to share. The programs are a Rollbook program (I teach college), a Gas and Electric bill analyzer program and several chemical calculation programs. All of the programs work for me. I have been in correspondence with the USUS group but they do not have many programs for the T. I., the majority of their programs are for the Apple. They want people to translate Apple programs to the T. I. I do not feel that the USUS group is for me because there is very little support for the T. I. community.

* * * * * * *

PASCAL/p-CODE PART 1

Stan Katzman

A while back John Willforth asked me if I would write a column on the Pascal/p-Code system. I was hesitant because I don't know too many people that have p-Code or people that want (or can get p-Code), so I don't know how much of a demand there is for knowledge of this system. I decided to do it anyway because this is the least covered system in the T. I. family.

Let us first discuss the hardware and software needed. In order to run Pascal on the T. I. one needs 1) the p-Code card, 2) at least two disk drives (I am running two DS/SD drives.) 3) RS232 card and a printer (this is necessary because the programs can get to be very large and debugging without a printout would be very difficult.), 4) 32K memory, 5) p-Code Editor, 6) Pascal compiler, 7) Filer, 8) Utilities, and 9) Assembler and Linker. I do not use the Assembler and Linker but if you can obtain these pieces of software do so and you can grow into them. You will also need the documentation (which is very extensive).

There are some books that I would also recommend 1)any Pascal text that covers Standard Pascal and/or U.C.S.D. Pascal (not Turbo Pascal), 2)"Introduction to the U.C.S.D. p-System", by Charles W. Grant and Jon Butah, Sybex, 1982, 3)"Advanced U.C.S.D. Pascal Programming Techniques", by Eliakim Willner and Barry Demchak, Prentice-Hall, 1985 and 4)"The U.C.S.D. Handbook", by Randy Clark and Stephen Koehler, Prentice-Hall, 1982.

After getting the hardware and software together the first thing that has to be done is to back up the software, this is very important, please do nothing else until this is done.

We have to format disks in order to back up the software so let us discuss how to do this in the p-Code system. If you only have SS/SD drives you can use the DFORMAT program on the Utilities disk. This software does not work for DS or DD disks, the disk will come out SS/SD in spite of what the menu prompts say in DFORMAT. To format a disk other than SS/SD go into X-Basic and using DM-1000 format a series of disks. Now shut off the console and turn on the p-Code card using the switch on the back and put the Filer disk in the #1 drive and a disk formatted from X-Basic in the #2 drive. (In p-Code the #1 drive is called #4, the #2 drive is called #5 and the #3 drive is called #9; we will use these numbers from here on.) Now turn on the console and the drives will turn on and after about a minute you will see across the top a series of prompts and in the middle of the screen a greeting message. The cursor will be at the top of the screen and at this point press "F" (for Filer) and then press "Z" (Zero). You will now see "Zero dir of ?" now enter "#5:", now it will say "Duplicate dir?", press "Y", next it says "Are there 180 blks on the disk? (Y/N)" (If the disk is DS/SD or DS/DD enter "N", if SS/SD enter "Y"), then the screen reads "# of blocks on the disk" (if DS/SD enter 360, if DS/DD enter 720), next the screen reads "New volume name?", at this point enter a disk name (the one from Basic will not count) with a max of 8 characters starting with a letter, next the screen reads "(new volume name) correct?" at this point enter "Y" or "N" ("N" lets you re-enter the volume name). When the Filer has finished initilizing the disk it displayes "(new volume name) zeroed". Do this until all of your disks are initilized. These are formatted disks. The zeroing process is found on page 52 of the Filer.

If you only have SS drives, put the Utilities disk in #4 and the blank disk in #5. Follow the direction on page 15 of the Utilities documentation. Then take out the Utilities disk and put the Filer disk in #4 and go through the "Zero" process described above.

To back up an entire disk put the Filer in #4 and empty formatted disk in #5. Call the Filer by entering "F" and then enter "T" (Transfer). You will then see "Transfer?", at this point take the Filer out of #4 and put the disk you want to be backed up in #4 then enter "#4:,#5:". You will now see "Destroy (diskname)" ("diskname" is the name of the disk in #5) at this point press "Y" and the copying will be automatic. Do this for all of the disks that came with system.

This article has been a bit long so we will quit here. More next time.

* * * * * * *

M.U.N.C.H. NOTES

by

JIM COX

I have decided to dedicate this issue of our newsletter to Pascal/p-Code. All of these articles appeared in the newsletter of the Chicago User Group. Stan Katzman is the author of these articles and I hope this will spur some interest in the TI community about Pascal/p-Code.

I apologize to anyone who has problems reading some of these pages. They were reduced to fit two pages to a sheet for the Chicago newsletter and I lost the left edge on one of the pages. I think you will still be able to make out the text.

****

I understand that some of you have ben asking about the possibility of a T.I. Faire for this fall. The reason we are not planning a fair this year is that we had a very small turn-out last year. We had less than sixth persons attend our fair last October and this was with as good a line up of T.I. "dignitaries" as you could hope to get. Everyone who attended had a great time but it was lot of work for such limited success.

Gary Fitzgerald has informed me that he has heard from a number of T.I. users via the Delphi service about the fair. One idea that has come up is some type of group meeting. This would have each group present doing a demo of some new product or an innovative use for an something old. there would also be some type of swap-meet type area set up so that there could be an informal exchange of used equipment and other stuff. I do not know when or where this would take place, these are some of the things which would have to be worked out.

If you have some interest in this contact Gary through the Internet on Delphi or write to him care of the M.U.N.C.H. address.