4M
HELP DOCUMENTATION IS ALSO AVAILABLE ON:
WRITING MACROS (HELP MACRO) CREATING PICTURES(HELP PIX)
DEVICE USAGE (HELP DEV) ARITHMETIC PROGRAMMING (HELP FORTRAN)
>>>>>>>>>>>>TYPE "HELP HELP" TO GET MORE INFORMATION<<<<<<<<<<<<
// **********DEV
DEV

DEVICE CONVENTIONS IN GRASS2

"DEV" IN GRASS GENERALLY MEANS ANYTHING KNOWN TO THE SYSTEM THAT CAN HOLD A VALUE FROM -32768 TO +32767 (THE LIMITS OF A 16-BIT NUMBER). BY CONTRAST, FLOATING POINT NUMBERS IN GRASS HAVE MUCH HIGHER RANGES AND TAKE UP MORE BITS AND ARE RELATIVELY TIME-CONSUMING TO CONVERT TO INTEGER. 16-BIT NUMBERS ARE WHAT THE VECTOR GENERAL(VG) FEEDS ON, AND FEEDING THE VG IS WHAT GRASS IS ALL ABOUT AT MACHINE LEVEL.

SO, THERE'S A BASIC DICHOTOMY IN GRASS ARITHMETIC BETWEEN FIXED AND FLOATING POINT NUMBERS. ALL DEV'S ARE FIXED POINT, AND CAN BE TIED DIRECTLY TO THE 16-BIT REGISTERS IN THE VG. DEV'S CURRENTLY INCLUDE DIALS, FIXED VARIABLES, FIXED ARRAYS AND, IN SOME CASES, FIXED POINT EXPRESSIONS. THUS, IN

A=B+D0/10

EVERYTHING BUT THE 10 AND THE =,+,/ ARE DEV'S BUT IN

FA=K+SIN(FB)*FC

ONLY THE K IS A DEV (SEE HELP HELP FOR DEFINITIONS HERE).

NOW, GRASS USES ARITHMETIC VARIABLES THAT ALREADY HAVE NAMES. THIS ELIMINATES THE DECLARE STATEMENTS THAT ARE SO CUMBERSOME IN MANY PROGRAMMING LANGUAGES AND ALSO VASTLY SIMPLIFIES WHAT GRASS HAS TO DO TO FIGURE OUT WHAT YOU'VE TYPED. SOMEDAY, WE MAY DO SOMETHING ABOUT LETTING PEOPLE HAVE ARBITRARY VARIABLE NAMES, BUT FOR THE TIME BEING, THE LEGAL DEV'S ARE:

ARITHMETIC:
- A-Z
- VA-VZ
- WA-WZ

LOCAL ARITHMETIC
- LA-LZ

ARRAY
- AA(...)-AM(...)

SPECIAL ARITHMETIC:
- 0A-0H NORMAL EXCEPT ALSO PUT OUT VOLTAGES TO THE IMAGE PROCESSOR. 0=0 VOLTS, 127=+1 VOLT, -128=-1 VOLT (EXPLAINED MORE LATER).
- SA-SH CONVERSION TO SINE FUNCTION (EXPLAINED LATER)
- CA-CH CONVERSION TO COSINE FUNCTION (EXPLAINED LATER)
- MA-MZ AND NA-NZ LINEAR TIME-BASED VARIABLES (SEE BELOW FOR DETAILED EXPLANATION)
- QA-QZ AND RA-RZ SINUSOIDAL TIME-BASED VARIABLES

INTERACTIVE CONTROL DIALS, ETC:
- D0-D9 TEN CONTROL DIALS
- S0-S9 TEN SLIDE POTENTIOMETERS
- P0-P3 FOUR MORE CONTROL DIALS
- JX,JY,JZ JOYSTICK #1
- KX,KY,KZ JOYSTICK #2
- TX,TY,TZ TABLET (TZ=1 FOR PRESS, 0 FOR NEAR & -1 FOR FAR)

DEV'S CAN BE USED IN ARTIHMETIC EXPRESSIONS (EXCEPT THAT YOU CANNOT WRITE TO DIALS/POTS/ETC.) AND THEY ALSO CAN BE ATTACHED TO ROTATIONS, TRANSLATIONS, SCALING AND SO ON. THIS LATTER CONVENIENCE ACTUALLY REQUIRES SOME COMPLICATED SYSTEMS PROGRAMMING, WHICH YOU SHOULDN'T WORRY ABOUT. ANYWAY, WHEN BEING USED IN ROTATE, MOVE, SCALE, CUTOFF, SETINT, SETCQ, AND SETORG, THE FOLLOWING RULES APPLY TO DEV'S:

RULE 1: IF THE DEV IS SPECIFIED ALONE, THE SYSTEM LOOKS UP ITS VALUE AND USES IT EVERY 1/60 SECOND. THIS IS WHAT GIVES THE INSTANT-SEEMING RESPONSE TO CHANGING DIALS, VARIABLES, ETC.

EXAMPLE: SCALE PIX,D0

RULE 2: IF THE DEV IS PART OF AN EXPRESSION, IT IS EVALUATED ONE-TIME ONLY AND USED AS A CONSTANT:

EXAMPLE: SCALE PIX,D9/100+A

PIX IS SCALED ON THE VALUE OF D9/100+A AT THE INSTANT THE SCALE COMMAND IS EXECUTED. IF A OR D9 CHANGE, THE SCALE DE

THE FIX WILL NOT CHANGE. TO MAKE RULE 2 APPLY, USE SCALE WHERE SINGLE VARIABLES ARE DESIRED. FAKE GRASS OUT BY DOING SOMETHING LIKE (A) OR A+0 TO MAKE IT LOOK LIKE AN EXPRESSION. NOTE THAT CONSTANTS ALONE ARE OK TOO. FOR INSTANCE, SCALE PIX,10000 IS THE SAME NOW AS

```
A=10000
SCALE PIX,A
FIX PIX                (SEE HELP FIX)
```

EXCEPT THAT IT IS FASTER, NEATER AND CLEANER.

RULE 3: IF AN = SIGN PRECEEDS A FIXED-POINT EXPESSION, THE EXPRESSION IS AUTOMATICALLY COMPILED AND THEN RE-EVALUATED EVERY 1/60 SECOND. THIS TAKES A SMALL AMOUNT OF MEMORY SO IT SHOULDN'T BE USED FOR SILLY THINGS LIKE SCALE SAM,=D0 WHICH IS COVERED BY RULE 1 ABOVE.

```
EXAMPLES:      SCALE BEAN,=D0/20
               SCALE BEEP,=A+C*(D-E)
               MOVE BLOOP,=CA+SA,=CB+SB,=CC+SC
```

RULE 4: IF A DEV OBEYING RULE 1 IS THE ONLY ONE SPECIFIED IN A COMMAND EXPECTING MULTI-DEV'S, THE DEV'S LOGICALLY FOLLOWING THE ONE SPECIFIED ARE USED AUTOMATICALLY. FOR EXAMPLE: MOVE SAM,D0 USES D0 FOR X TRANSLATION, D1 FOR Y TRANSLATION AND D2 FOR Z TRANSLATION. ROTATE SAM,Y,D0,D1,VL USES VL FOR X ROTATION ORIGIN, VM FOR Y ROTATION ORIGIN AND VN FOR Z ROTATION ORIGIN. SIMILARLY, THE CUTOFF COMMAND EXPECTS SIX DEV'S SO CUT SAM,D3 USES D3 FOR X HIGH BOUNDARY, D4 FOR X LOW BOUNDARY, D5 FOR Y HIGH, D6 FOR Y LOW, D7 FOR Z HIGH AND D8 FOR Z LOW. SEE THE HELPS FOR EACH COMMAND TO SEE HOW MANY DEV'S ARE EXPECTED.
NOW, AND THIS IS IMPORTANT, DEFAULT VALUES ARE ASSUMED WHERE ARGUMENTS ARE MISSING IF RULE 4 IS NOT APPLICABLE:

RULE 5: IF DEV'S NOT OBEYING RULE 1 ARE USED, DEV'S MISSING AT THE END ARE ASSUMED TO TAKE DEFAULT VALUES (0 FOR MOVE, SETORG, SETINT, SETCQ, +32767 FOR SCALE (ALL SWITCHES) AND CUTOFF X,Y,Z HIGH VALUES, AND -32768 FOR CUTOFF X,Y,Z LOW VALUES).

```
EXAMPLES:      MOVE SAM,1000 IS THE SAME AS
               MOVE SAM,1000,0,0
AND
               MOVE SAM,D0,1000 IS THE SAME AS
               MOVE SAM,D0,1000,0
AND
               MOVE SAM,D0,D1  SETS Z TO ZERO
AND
               MOVE SAM,=D0/10 LEAVES Y AND Z SET TO ZERO
```

RULE 6: IF THE ARGUMENTS ARE NULL (I.E. ",,") THE DEV SETTING IS LEFT ALONE. FOR EXAMPLE:

```
MOVE SAM,1000    MOVES SAM TO 1000,0,0
FOLLOWED BY
MOVE SAM,,-1000 MOVES SAM TO 1000,-1000,0
```

THIS IS PARTICULARLY USEFUL IN THE CUTOFF COMMAND BECAUSE THE DEFAULT VALUES ARE SETUP AS:

```
CUT SAM,32767,-32768,32767,-32768,32767,-32768
```

WHICH IS NOT SO MUCH FUN TO TYPE IF YOU ONLY WANT TO CHANGE THE Z-HIGH COMPONENT.

NOTE: WHEN RULE 2 IS FOLLOWED IN SPECIFYING A DEV (INCLUDING CONSTANTS), THE PICTURE DOES NOT NEED TO BE FIX'ED

SPECIAL DEV'S

1. THE DIALS, POTS, ETC. CANNOT BE WRITTEN INTO, THAT IS, THEY CANNOT BE ON THE LEFT SIDE OF AN EQUAL'S SIGN OR USED IN GETPOINTS

2. DA-DH ARE DIGITAL-TO-ANALOG OUTPUT VARIABLES THAT MAY BE WRITTEN INTO AND READ FROM AS NORMAL EXCEPT THAT THEY ALSO PRODUCE VOLTAGES IN A BOX BY THE IMAGE PROCESSOR. THE RANGE IS +1 VOLT =127, -1 VOLT =-128, AN 8-BIT RANGE.

3. SA-SH AND CA-CH ARE SPECIAL BECAUSE THEY CAUSE A SINE/COSINE CONVERSION WHEN ASSIGNED (I.E. APPEAR ON THE LEFT SIDE OF THE EQUALS SIGN). FOR EXAMPLE:

```
SA=0                   (SEE HELP FORTRAN)
PROM SA      GIVES 0       (SEE HELP PROMPT)
SA=256
PROM SA      GIVES 32767
```

180 DEGREES = PI RADIANS = 512 FOR SA-SH AND CA-CH. SC EACH CHANGE OF 1 EQUALS A CHANGE OF APPROXIMATELY 1/3 DEGREE. THESE VARIABLES ARE USEFUL FOR EASILY CREATING SINUSIODAL MCTIONS. NOTE THAT THE CONVERSIONS WILL NOT TAKE PLACE IN GETP SAM,10,SA,SB,SC,SD. THE CONVERSIONS ONLY TAKE PLACE "OVER" EQUAL SIGNS

5. TIME BASED VARIABLES

THE TIME-BASED VARIABLES COME IN PAIRS (MA-MZ/NA-NZ AND QA-QZ/RA-RZ), THE FIRST SET OF WHICH IS LINEAR AND THE SECOND SET IS SINUSOIDAL. THE FIRST OF THE PAIR IS USED TO SPECIFY HOW LONG IT SHOULD TAKE TO GET TO THE GOAL SET BY THE SECOND OF THE PAIR. THE TIME IS SPECIFIED IN 1/60THS OF SECONDS SO MA=240 MEANS 4 SECONDS, MK=600 MEANS 10 SECONDS, AND SO ON. THE SYSTEM AUTOMATICALLY DECREMENTS THE M AND Q VARIABLES EVERY 1/60 SECOND UNTIL THEY ARE ZERO. YOU CAN TEST THE M AND Q VARIABLES LIKE NORMAL VARIABLES SO YOU CAN SEE WHEN THEY ARE ZERO, OR WHEN FIVE SECONDS ARE UP, AND SO ON. FURTHERMORE, THE LINEAR TIME VARIABLES (MA-MZ) MAY BE CHANGED ANYTIME IN THE COURSE OF EVENTS AND THE SYSTEM WILL ADJUST. THE SINUSOIDAL ONES (QA-QZ MAY BE CHANGED TOO, BUT THE EFFECT IS NOT TERRIBLY SMOOTH UNLESS THEY HAVE ALREADY HIT ZERO.
BASICALLY, THE IDEA IS TO SET THE TIME VARIABLE AND SET THE CORRESPONDING GOAL VARIABLE (QA CORRESPONDS TO RA, MJ TO NJ, ETC.) ON THE SAME LINE. IT IS IMPORTANT TO SET THE TIME VARIABLE FIRST, BECAUSE THE SYSTEM WILL IMMEDIATELY UPDATE THE CORRESPONDING GOAL VARIABLE TO ITS FULL VALUE IF THE TIME VARIABLE IS ZERO (WHICH IT PROBABLY WOULD BE IF YOU SET THE GOAL VARIABLE FIRST). IF YOU DON'T TOUCH THE M'S AND Q'S, YOU CAN USE THE R'S AND N'S AS NORMAL VARIABLES, BY THE WAY.

SOME EXAMPLES:

```
MOVE SAM,NA
MA=0;NA=-1000;MA=180;NA=10000
```

THIS WILL TAKE THREE SECONDS TO GO FROM -1000 TO 10000

```
SCALE SAM,RC
QC=120;RC=-15000
IF QC GT 0,SKIP 0
QC=360;RC=20000
```

THE ABOVE SEQUENCE FIRST SCALES SAM FROM HALF SCALE (BECAUSE RC IS ZERO INITIALLY UNLESS OTHERWISE SET) TO MUCH SMALLER IN TWO SECONDS, AND THEN UP TO ABOUT 2/3 SCALE IN SIX SECONDS, IN A SINUSOIDAL FASHION. NOTE THE WAIT FOR THE QC VARIABLE TO HIT ZERO BEFORE PROCEEDING. YOU MAY VERY EASILY DOLOOP SEVERAL OF THESE SEQUENCES WITH THE GENERAL TIMING BEING UNAFFECTED BY THE AMOUNT OF DOLOOP'ING, SOMETHING WE NEVER COULD DO BEFORE. NOTE THAT THE SYSTEM IMMEDIATELY STARTS TO DECREMENT THE TIME VARIABLES SO YOU SHOULD SET THE GOAL VALUES ON THE SAME LINE WHENEVER POSSIBLE.

THAT'S ALL FOR THE TIME BEING.....

// **********MACROS
MACROS

WRITING MACROS IN GRASS

1. WHAT IS A MACRO ANYWAY??

A MACRO IN GPASS IS SIMPLY A SET OF COMMANDS. BEFORE GRASS HAD MACROS, YOU HAD TO TYPE IN ALL COMMANDS ONE BY ONE. THIS WAS OBVIOUSLY QUITE TEDIOUS WHEN LONG SEQUENCES HAD TO BE REPEATED SO WE SOON ALLOWED PEOPLE TO SAVE SETS OF COMMANDS ON THE DISK WITH NAMES SO THEY COULD BE RECALLED AT ANY TIME. MACROS, AT FIRST, WERE VERY MUCH LIKE PLAYER PIANO ROLLS, BEING ABLE TO STORE ONLY LONG SEQUENTIALLY EXECUTED LISTS OF COMMANDS WHICH WERE PLAYED FROM ONE END TO THE OTHER, WITH NO BRANCHING OR REPEATING. THIS USE OF MACROS IS STILL VERY COMMON AND VALID, SO HERE'S HOW TO DO

THERE ARE THREE WAYS OF ENTERING MACROS--

1. INTERACTIVELY, BY TYPING A NAME, A COLON, AN OPEN BRACKET, THE COMMANDS WANTED, AND THEN A CLOSE BRACKET LIKE THIS:

```
SAM:<GETDSK GLOBE
ROTATE GLOBE,Y,D0
SCALE GLOBE,D1
MOVE GLOBE,D2>
```

THIS ACTION CAUSES A SMALL AMOUNT OF MEMORY (CALLED "CORE" BECAUSE OF THE IRON CORES USED IN THE MEMORY) TO BE ALLOCATED WITH THE NAME SAM. "DO SAM" WILL CAUSE GRASS TO EXECUTE THE COMMANDS IN SAM ONE-BY-ONE UNTIL IT HITS THE END BRACKET. THEN GRASS RETURNS CONTROL TO WHATEVER ISSUED THE DO, WHETHER IT WAS YOU ON THE TELETYPE OR ANOTHER MACRO. NORMALLY, WHEN YOU ARE TYPING ON THE TELETYPE, YOU ARE IN WHAT WE CALL "STAR-LEVEL" (BECAUSE THE SYSTEM PRINTS "*"S TO INDICATE A WILLINGNESS TO ACCEPT COMMANDS).

NORMALLY, MACROS ARE STORED ON THE DISK FOR FUTURE USE BY THE FOLLOWING COMMAND:

```
PUTDSK SAM
```

SAM IS STORED ON THE DISK AS SAM.MAC THE .MAC IS AN EXTENSION AND IT TELLS YOU AND GRASS WHAT TYPE OF THING IS STORED ON THE DISK. A .DEC REFERS TO A GRASS PICTURE FILE (FROM DECIMAL) AND THE REST OF THE GRASS EXTENSIONS ARE GIVEN IN HELP HELP (JUST TYPE HELP HELP).

NOW, TO RETRIEVE SAM AT A LATER DATE, SIMPLY TYPE DO SAM. IF SAM IS ALREADY IN MEMORY, IT WILL START TO EXECUTE. OTHERWISE, IT WILL BE FETCHED FROM THE DISK UNDER THE NAME SAM.MAC. DIRDSK WILL TELL YOU WHAT IS ON YOUR DISK AREA. YOU MAY ALSO TYPE SAM.MAC OR PRINT SAM.MAC OR GETDSK SAM.MAC IF YOU WISH. SEE THE RESPECTIVE HELPS (HELP TYPE, ETC.) FOR MORE INFORMATION.

YOU CAN DELETE A MACRO BY TYPING DELETE SAM (PRESUMING, OF COURSE, AS WE HAVE HERE, THAT SAM IS THE NAME OF YOUR MACRO). YOU CAN GET IT ERASED FROM THE DISK BY TYPING DELETE/D SAM.MAC.

OFTEN, HOWEVER, ONE WANTS TO CHANGE A MACRO RATHER THAN TYPE IT ALL IN AGAIN. THE NEXT WAY OF ENTERING MACROS ALSO ALLOWS CHANGING.

2. USING THE EDITORS. THERE ARE TWO EDITORS IN GRASS--CALL EDIT AND EDIT MNAME--WHERE MNAME IS THE NAME OF ANY MACRO. CALL EDIT IS WORDY, ASKS YOU QUESTIONS AND HAS LOTS OF FEEDBACK. IT IS THE BEST EDITOR TO START WITH. YOU GET IT BY TYPING

```
RESTART CALL EDIT
```

IT WILL ASK YOU FOR THE NAME AND EXTENSION. IT ALSO ASKS FOR SOME INFORMATION ON WHAT THE MACRO IS TO BE USED FOR, AN AID TO YOUR OWN DOCUMENTATION. THIS INFORMATION IS STORED ON THE DISK UNDER MACROS.DOC WHICH YOU CAN TYPE OR PRINT ANYTIME.

THE IDEA BEHIND EDITING MACROS IS VERY SIMPLE. MACROS ARE COMPOSED OF CHARACTER STRINGS AND THEY LOOK TO THE COMPUTER LIKE A BUNCH OF CHILDREN'S BLOCKS WITH LETTERS ON THEM. A COMMAND IN GRASS IS A SEQUENCE OF LETTERS (CHARACTERS) IN A ROW. ROTATE GLOBE,Y,D0 IS ACTUALLY STORED AS 19 BLOCKS IN MEMORY (ACTUALLY CALLED 'BYTES'). IF YOU COUNTED THE LETTERS AND SPACES, YOU GOT 17. THE TWO EXTRA ARE THE CARRIAGE RETURN <CR> AND LINEFEED <LF> WHICH TELL GRASS THAT THE COMMAND LINE IS DONE. (BY CONTRAST, COMPUTER CARDS ARE ALWAYS 80 CHARACTERS, EVEN THOUGH MOST OFTEN MOST OF THE SPACE IS BLANK, A WASTE OF SPACE AND TREES.) GRASS COMMANDS, THEN, ARE TERMINATED BY A CRLF SEQUENCE, WHICH, CONVENIENTLY, ALSO TELLS THE TELETYPE AND LINE PRINTER WHAT MECHANICAL ACTION TO PERFORM WHEN THE END OF THE LINE IS HAPPENING.

SAY YOU HAVE A TEN-LINE MACRO CALLED SAM. THE BASIC THINGS YOU WANT TO DO IS ADD LINES (CALLED INSERTING) AND TAKE LINES OUT (DELETING). WITH CARD DECKS, THIS IS A PHYSICAL PROCESS, BUT IN GRASS, IT IS DONE ELECTRONICALLY AND ELECTROMAGNETICALLY. YOU HAVE TO GIVE THE EDITOR INSTRUCTIONS THAT LOGICALLY CORRESPOND TO ADDING AND PULLING CARDS FROM A DECK. GRASS'S EDITORS USE LINE NUMBERS SO YOU CAN EASILY TELL THE COMPUTER WHICH LINES TO DELETE OR AFTER WHICH LINE TO INSERT NEW LINES. NORMALLY, YOU ALSO WANT TO BE ABLE TO SEE THE CHANGES AS YOU MAKE THEM SO THERE'S A TYPE COMMAND AND A PRINT COMMAND, THE LATTER OF WHICH USES THE LINE PRINTER TO GET YOU HARDCOPY TO TAKE HOME AND SAVE.

MOST PROGRAMMERS ARE TOO LAZY TO TYPE IN A WHOLE NEW LINE WHEN SOMETHING SIMPLE HAS TO BE CHANGED, SO WE ALLOW CHANGES TO BE MADE WITHIN A LINE WITH THE CHANGE COMMAND IN THE EDITOR. ONCE YOU CALL EDIT, AND ENTER THE MACRO NAMES, YOU CAN TYPE HELP TO GET SOME DOCUMENTATION ON THE COMMANDS AVAILABLE. YOU MIGHT AS WELL HAVE SOMEONE FRIENDLY HELP YOU EDIT THE FIRST TIME OR TWO.

WHEN YOU ARE FINISHED EDITING, USE THE EXIT COMMAND TO GET THE MACRO CHANGED ON THE DISK. THE OTHER EDITOR, EDIT MNAME, IS A QUICKY, IN-CORE EDITOR. HELP EDIT DESCRIBES

EDIT FAST.

3. MACROS MAY ALSO BE CONSTRUCTED WITH STRING MANIPULATION COMMANDS. THIS IS PRETTY ESOTERIC STUFF WHICH WE MAINLY USE TO WRITE PROGRAMMING SUPER-LANGUAGES IN GRASS OR GET AROUND THINGS THE SYNTAX OF GRASS CANNOT OTHERWISE HANDLE. SEE HELP STRINGS WHEN YOU'RE READY FOR THIS.

**********NOTE THAT MACROS, WHEN STORED ON THE DISK, OR ENTERED WITH THE EDITORS, DO NOT HAVE THE "NAME:<" AT THE BEGINNING OR THE ">" AT THE END. ONLY THE TEXT IS USED, ACTUALLY, AND THE BRACKETS ARE TO TELL GRASS THAT IT'S A MACRO ONLY DURING THE CREATION MODE DESCRIBED IN 1. ABOVE.

B. MACROS AS COMPUTER PROGRAMS

WE NORMALLY PROGRAM COMPUTERS USING LOOPS. LOOPS ARE SETS OF INSTRUCTIONS REPETITIVELY EXECUTED UNTIL SOME CONDITION IS SATISFIED. GRASS DOES A LOT OF INTERNAL HOUSEKEEPING TO HELP THE USER GET AROUND WRITING TRIVIAL LOOPS THAT GRAPHICS PROGRAMMING IN FORTRAN, SAY, REQUIRES. THE DOLOOP METHOD OF RUNNING MACROS AND THE TIME-BASED VARIABLES AS WELL AS SIMPLE COMMANDS LIKE ROTATE, SCALE AND MOVE ALL HELP SET UP THINGS THAT NORMALLY REQUIRE COMPLICATED LOOPS IN COMMON GRAPHICS LANGUAGES. NEVERTHELESS, LOOPS ARE VERY USEFUL.

THE COMMAND THAT EXPLICITLY CAUSES LOOPING IS THE SKIP COMMAND. IT IS EXACTLY LIKE THE GOTO JAIL IN MONOPOLY OR THE CHUTES AND LADDERS IN CHUTES & LADDERS. LOOPS SHOULD NORMALLY HAVE AN ENDING CONDITION, OTHERWISE, AN INFINITE LOOP RESULTS AND YOU MUST EXIT WITH A CONTROL-C (HOLD THE CTRL KEY DOWN AND HIT THE C AT THE SAME TIME, LIKE A SHIFT C--CONTROL-C IS ABBREVIATED |C). |C ALWAYS GETS YOU BACK TO STAR-LEVEL. WHEN A MACRO IS EXECUTING, GRASS WILL NOT ACCEPT COMMANDS FROM THE TELETYPE, WITH EXCEPTIONS WHICH YOU WILL DISCOVER ON YOUR OWN.

AN EXAMPLE OF A LOOP:

```
MOVE GLOBE,A,B,C
A=-10000
B=-5000
C=-2000
%LOOP A=A+50
B=B+25
C=C+10
IF A LT 15000,SKIP %LOOP
```

THIS LOOP WILL MOVE THE GLOBE FROM THE LOWER MIDDLE LEFT OF THE SCREEN TO THE UPPER MIDDLE RIGHT IN A FEW SECONDS. WHEN THE VALUE OF A IS 15000, THE MACRO STOPS SKIPPING BACKWARDS. YOU SHOULD BE ABLE TO FIGURE OUT THE END VALUES OF B AND C BY NOW.
THE IF STATEMENT WORKS AS FOLLOWS:

IN THE CASE THAT THE CONDITION IS TRUE, THE STUFF FOLLOWING THE COMMA IS EXECUTED. WHEN THE CONDITION IS FALSE, IF DEV CONDITION EXPR,STUFF
THE STUFF FOLLOWING THE COMMA IS IGNORED AND THE NEXT LINE DOWN IS PROCESSED. SEE HELP IF FOR MORE DETAILS.

NOTE THAT ANY COMMANDS MAY BE REPETITIVELY USED IN LOOPS EXCEPT THINGS LIKE RESTART, EXIT AND GOTO WHICH CAUSE THE CONTROL TO BE TRANSFERRED OUT OF THE MACRO. ROTATE, MOVE AND SCALE USUALLY AREN'T USED INSIDE LOOPS, JUST THE VARIABLES NEED BE CHANGED INSIDE THE LOOP.

MACROS CAN USE OTHER MACROS SIMPLY BY SAYING DO MNAME WITHIN A MACRO. THE MNAME IS EXECUTED AND AFTER IT IS FINISHED, CONTROL RETURNS TO THE STATEMENT AFTER THE ONE THAT HAD THE DO MNAME IN IT. IF THIS IS NOT CLEAR AT THIS POINT, ASK SOMEONE TO EXPLAIN THE CONCEPT OF SUBROUTINES.

C. GETTING MACROS TO TALK TO YOU AND LISTEN TO YOU

A. TALKING OR MORE PROPERLY, PROMPT'ING:

THE PROMPT COMMAND TELLS GRASS THAT YOU WANT THE VALUE OR CONTENTS OF SOMETHING TYPED ON THE TELETYPE. PROMPT DO WILL TELL THE CURRENT VALUE OF DO. PROMPT 700/19 WILL GIVE THE VALUE 36.84211, SO YOU CAN USE PROMPT AS A RATHER FANCY POCKET ALCULATOR. SIMILARLY, PROMPT SQR(FA*0) WILL GIVE A RESULT DEPENDING ON FA. PROMPT ALSO CAN TYPE MESSAGES (HENCE ITS NAME) TO THE USER, LIKE

```
PROMPT "WHAT IS YOUR NAME"
```
OR
```
PROMPT "THE VALUE OF A IS ",A
```
OR
```
PROMPT "THIS MACRO HAS EXECUTED ",X,"TIMES"
```

ANY MIX OF VARIABLES AND STUFF IN QUOTES (CALLED STRING LITERALS OR JUST LITERALS) MAY BE PROMPT'ED. PROMPT IS MOST OFTEN USED IN CONJUNCTION WITH THE LISTEN-TO-YOU COMMAND, THE INPUT COMMAND.

B. INPUT FROM THE TELETYPE

INPUT IS ONE WAY OF PUTTING VALUES IN VARIABLES. THE OTHER WAY IS DIRECT ASSIGNMENT LIKE:

```
A=10
K=K+500
```

USUALLY, IN A MACRO, YOU WANT THE USER TO TYPE THE VALUE OF SOME-

WOULD BE CLUMSY. THE INPUT COMMAND ALWAYS PRINTS OUT A QUESTION MARK AND WAITS FOR SOMETHING TO BE TYPED AND CONTINUES TO WAIT UNTIL IT SEES A CRLF(CARRIAGE RETURN). SO

```
BEEP:<INPUT A
PROMPT A*A*3.14159
SKIP -2>
```

WILL TYPE OUT THE AREA OF A CIRCLE WHOSE RADIUS YOU HAVE TO TYPE IN UNTIL YOU HIT A |C TO STOP IT. THIS IS SIMPLER THAN TYPING PROMPT A*A*3.14159 EACH TIME. LESS TRIVIAL LOOPS SHOULD OBVICUSLY BE POSSIBLE.

NORMALLY, THE MACRO SHOULD ASK FOR WHAT IT NEEDS IN THE INPUT. FOR EXAMPLE:

```
BEEP:< PROMPT "WHAT IS THE RADIUS"|
INPUT A
PROMPT "THE AREA IS ",A*A*3.14159
SKIP -4>
```

THE "|" MAKES THE "?" FROM THE INPUT COMMAND APPEAR ON THE SAME LINE AS THE PROMPT. YOU SHOULD NOW TRY SOME OF THESE SIMPLE EXAMPLES, BECAUSE IT'S GOING TO GET REALLY COMPLICATED HERE CN IN.

C. PROMPT'ING AND INPUTTING STRING VARIABLES.

OFTEN A MACRO IS WRITTEN TO BE GENERAL PURPOSE AND WORK ON ANY PICTURE NAME. FOR INSTANCE, THE EARLIER MACRO FOR PLAYING WITH THE GLOBE COULD BE REWRITTEN TO TAKE ANY PICTURE AS FOLLOWS:

```
NEWSAM: <PROM "WHICH PIX"|
INPUT $A
GETDSK $A
ROTATE $A,Y,D0
SCALE $A,D1
MOVE $A,D2
PROMPT $A," IS MOVING ON D2,D3,D4; SCALING ON D1"
PROMPT "AND ROTATING AROUND THE Y-AXIS ON D0">
```

THIS IS AN EXAMPLE OF A MACRO THAT SAVES EFFORT. THERE ARE SEVERAL GOOD REASONS FOR CREATING MACROS LIKE THIS: FIRST, YOU CAN DO THINGS FASTER IF THEY CAN BE REPEATED AT COMPUTER RATHER THAN HUMAN TYPING SPEEDS. SECOND, OTHER PEOPLE CAN WRITE MACROS FOR YOUR USE AND VICE VERSA. THIRD, YOU CAN CALL THIS MACRO FROM OTHER MACROS YOU WRITE AND BUILD UP A LIBRARY OF USEFUL SEQUENCES. EVENTUALLY, YOU WILL FIND THAT YOU NEVER DO ANYTHING TWICE AND THAT'S GOOD.

BY THE WAY, IT'S BETTER TO CALL YOUR MACROS SOMETHING OTHER THAN SAM AND NEWSAM. MACRO NAMES SHOULD BE SUGGESTIVE OF WHAT THEY DO. NOTE THAT MACRO NAMES CAN ONLY BE SIX CHARACTERS LONG.

D. SOME POINTS:

1. MACROS ARE ENDED BY FALLING OFF THE END. NO EXPLICIT END STATEMENT IS REQUIRED AS IN MOST PROGRAMMING LANGUAGES. SKIPPING OFF THE END OF THE MACRO CAUSES IT TO RETURN TO THE MACRO IT WAS CALLED FROM. THE RETURN COMMAND ACTUALLY FAKES A SKIP 9999. SKIPPING BACKWARDS MORE LINES THAN THERE ARE IS A CONVENIENT WAY TO GET BACK TO THE BEGINNING OF A MACRO.

2. COMMANDS AND ALL NAMES IN GRASS (EXCEPT %LABELS) CAN BE ABBREVIATED TO AS FEW LETTERS AS UNIQUELY IDENTIFY THEM. YOU WILL SOON LEARN THAT R IS GOOD ENOUGH FOR ROTATE, AND SO ON, ALL BY EXPERIMENTATION. YOU CAN CREATE THE NAMES SAM AND SAM1, BUT NOT IN THE ORDER SAM1, SAM, BECAUSE THE SYSTEM ASSUMES SAM IS AN ABBREVIATION FOR SAM1 ONCE SAM1 HAS BEEN CREATED. YOU CANNOT HAVE TWO THINGS NAMED THE SAME THING, NOR CAN YOU HAVE A MACRO AND A PICTURE IN CORE WITH THE SAME NAME. THE SYSTEM WILL YELL AT YOU FOR TRYING. THESE RESTRICTIONS DO NOT APPLY TO THE DISK NAMES, HOWEVER.

3. MULTIPLE COMMANDS MAY BE PUT ON A SINGLE LINE BY SEPARATING THEM WITH SEMI-COLONS. THIS IS VERY USEFUL FOR IF STATEMENTS. THERE ARE SEVERAL EXCEPTIONS: SEMI-COLONS MAY NOT BE USED AFTER DO'S OR CALL'S AND AFTER THE TEXT COMMAND. AN EXAMPLE:

```
PROMPT "HOW OLD ARE YOU"|;INPUT B
```

4. STUDY SOME OF THE SIMPLE SYSTEM MACROS LIKE MERGE, JOIN, BIGGER, BIGGST, AND SO ON. YOU CAN GET COPIES BY TYPING PRINT MERGE.MAC 31,3 , ETC.

E. SOME FINE POINTS...

COMMENTS IN YOUR MACRO ARE PUT IN BY YOU TO TELL YOURSELF AT A LATER DATE WHAT THE LOGIC BEHIND THE MACRO WAS. USE COMMENTS IF YOU DO NOT NOW, YOU WILL EVENTUALLY AFTER LEARNING THE LESSON THROUGH CONSIDERABLE DUPLICATION OF EFFORT. A COMMENT IS SIMPLY ANY LINE BEGINNING WITH A "*".

SEVERAL THINGS TO NOTE ABOUT COMMENTS:

1. LINES BEGINNING WITH A * ARE AUTOMATICALLY STRIPPED BY THE SYSTEM WHEN THE MACRO IS GOTTEN FROM THE DISK. THIS ACTION IS TO SAVE SPACE SINCE COMMENTS SHOULD BE AT LEAST AS MUCH TEXT AS COM-

NING WITH *'S.
2. *'D LINES ARE NOT COUNTED BY THE SKIP COMMAND (SEE HELP SKIP), BUT THIS IS NOTEWORTHY ONLY IF YOU USE NUMBERS INSTEAD OF %LABELS IN YOUR SKIPS.

USE OF |S:
|S TEMPORARILY SUSPENDS EXECUTION OF A MACRO. THE SYSTEM PUTS YOU IN "#"-MODE IN WHICH YOU CAN USE ANY COMMAND. TYPICALLY, YOU PROMPT VALUES OF MACROS, OR USE THE LIST OR XLIST COMMAND, OR FIX SOMETHING GONE WRONG OR OTHERWISE MISSING. TO GET BACK INTO THE MACRO, TYPE RESUME.

THE COMPILER
THE COMPILER SPEEDS UP MACROS CONSIDERABLY. SEE HELP COMPILE AND HELP EXECUTE.

LOCAL VARIABLES
THE VARIABLES LA TO LZ ARE FIXED POINT VARIABLES WHICH ARE KNOWN ONLY TO THE MACRO THEY EXIST IN. THEY SHOULD BE USED WHENEVER POSSIBLE. THE SET EA TO EZ ARE FLOATING LOCALS. ALL OTHER VARIABLES ARE GLOBAL, WHICH MEANS ALL MACROS KNOW ABOUT THEM. THIS MAKES GLOBAL VARIABLES USEFUL FOR PASSING INFORMATION, BUT OCCASIONALLY CONFUSES VALUES. YOU WILL PROBABLY DISCOVER HOW AS YOU START TO PROGRAM IN GRASS. AT ANY RATE, YOU CAN REFER TO THE LOCAL VARIABLES OF A MACRO CALLED SAM FROM ANOTHER MACRO BY TYPING LA_SAM.

CLEVERLY PASSING VARIABLES
VARIABLES (NUMERIC AND STRING) MAY BE PASSED AS ARGUMENTS (STUFF WITH COMMAS AROUND) TO THE DO OR CALL OR EXECUTE COMMANDS. FOR EXAMPLE, YOU COULD USE THE MACRO NEWSAM FROM BEFORE BY TYPING:

```
DO NEWSAM,GLOBE
```

THE STUFF FOLLOWING THE MACRO NAME IS AUTOMATICALLY FED TO THE INPUT COMMANDS UNTIL IT RUNS OUT. IF THE MACRO HAS MORE INPUTS LEFT, THE PROMPTS, WHICH HAVE BEEN PUT TO SLEEP, SUDDENLY WAKE UP AND START ASKING QUESTIONS AGAIN. IF YOU WANT TO SUPPRESS THE LAST PROMPTS IN THE MACRO NEWSAM, TOSS IN AN EXTRA COMMA. YOU CAN ALSO USE THIS TECHNIQUE WITH SYSTEM MACROS:

```
CALL JOIN,THING,1000
```

OR

```
CALL EDIT,SAM,MAC,,,
```

YOU CAN ALSO PASS VARIABLES IN GLOBAL VARIABLES OR STRINGS, BUT THE MACRO HAS TO EXPECT THEM THAT WAY.

ERROR MESSAGES
GRASS'S ERROR REPORTING FACILITIES ARE PRETTY GOOD. WHEN YOU GET AN ERROR, THE SYSTEM STOPS, TYPES QUESTION MARKS, AN ERROR NUMBER AND TRIES TO LOCATE THE THING THAT CAUSED THE ERROR BY POINTING AT IT WITH A "|"
IT THEN STICKS YOU IN "#"-MODE, JUST LIKE |S. YOU CAN FIX THE PROBLEM, |C TO EXIT, OR, IF YOU DON'T KNOW THIS ERROR NUMBER BY HEART YET (THERE'S ABOUT 200 YOU MIGHT RUN INTO), TYPE A "?" FOLLOWED BY A <CR> AND THE ERROR WILL BE EXPLAINED AS BEST WE CAN. MOST ERRORS ARE EITHER SPELLING ERRORS, RUNNING OUT OF CORE SPACE, OR NOT HAVING SOMETHING IN CORE THAT YOU REFER TO.

```
//  **********************************************************PIX
PIX
```

CREATING PICTURE LISTS IN GRASS

THERE ARE TWO BASIC WAYS TO CREATE PICTURES IN GRASS:

1. WITH THE EDITOR CALL EDIT

USING CALL EDIT, YOU CAN INPUT A PICTURE BY TYPING .DEC FOR THE EXTENSION. THE THING.DEC YOU CREATE IS STORED ON THE DISK AND MAY BE DISPLAYED WITH A GETDSK THING. THE POINTS THAT MAKE UP THE ENDPOINTS OF THE VECTORS YOU WISH TO DRAW MUST BE ENTERED AS FOLLOWS:

```
X,Y,Z
```

WHERE X, Y, AND Z RANGE FROM +2000 TO -2000. ALL THREE MUST BE SPECIFIED, ONE TO A LINE. THUS, A SQUARE IN THE UPPER RIGHT QUADRANT OF THE SCREEN IS REPRESENTED BY:

```
0,0,0
1000,0,0
1000,1000,0
0,1000,0
0,0,0
```

NOTE THAT FIVE POINTS ARE NEEDED TO DRAW FOUR LINES. THE SYSTEM ASSUMES LINES ARE DRAWN FROM THE PRESENT POINT TO THE NEXT POINT SPECIFIED. NOTE ALSO THAT THE ABOVE FORMAT ALLOWS SPACING AS YOU WISH SO FORTRAN I4 FORMAT ON CARDS WITH COMMAS BETWEEN IS ACCEPTABLE TO GRASS.

TO GO TO THE NEXT POINT WITHOUT DRAWING A VECTOR (CALLED A MOVE-NO-DRAW), PUT A "J" ON THE LINES BETWEEN THE POINTS NOT TO BE CONNECTED. TO DRAW A "T", THE FOLLOWING IS ACCEPTABLE:

```
0,0,0
0,1000,0
J
-500,1000,0
500,1000,0
```

AND SO ON. THE FIRST POINT IS ALWAYS ASSUMED TO BE A MOVE-NO-DRAW.

AFTER TYPING THE POINTS YOU WANT, EXIT FROM THE EDITOR.

2. WITH PUTPOINT

THE PUTPOINT COMMAND IS USED TO CONSTRUCT PICTURES IN MACROS. IT HAS A SIMILAR FORMAT TO THE DISK FORMAT ABOVE, EUT ENCODES THE MOVE-NO-DRAWS DIFFERENTLY. HELP PUTPOINT SHOULD BE CONSULTED FOR EXACT DETAILS. BRIEFLY, HOWEVER, THE PROCEDURE IS AS FOLLOWS:

A. FIRST, YOU MUST OPEN THE PICTURE NAME. OPEN TELLS GRASS WHAT NAME THE ENSUING PUTPOINT'S, DELPOINT'S AND CLOSE WILL REFER TO. YOU MUST USE AN OPEN BEFORE PUTPOINT'ING.

```
OPEN BOX
```

B. THEN YOU TYPE IN, OR HAVE IN A MACRO:

```
PUTP 0,0,0,0
PUTP 1000,0,0,0
PUTP 1000,1000,0,0
PUTP 0,1000,0
PUTP 0,0,0
```

THIS WILL DRAW THE BOX. NOW YOU MUST TELL THE SYSTEM TO CLOSE THE PICTURE OR IT CANNOT BE PUTDSK'D PROPERLY. AND YOU WASTE MOST

CLOSE). THE FOURTH ARGUMENT TO PUTPOINT IS COMMONLY CALLED K AND SPECIFIES A DRAW IF K=0, AND A MOVE-NO-DRAW IF K=1. SO THE "T" FROM BEFORE LOOKS LIKE:

```
OPEN TEE
PUTP 0,0,0,0
PUTP 0,1000,0,0
PUTP -500,1000,0,1
PUTP 500,1000,0,0
CLOSE
```

THE NUMBERS ABOVE MAY BE REPLACED BY VARIABLES OR ARITHMETIC EXPRESSIONS. FOR EXAMPLE, A MACRO TO DRAW A SERIES OF 200 PARALLEL VERTICAL LINES IS:

```
LINES:<OPEN VERTS
A=-10000
*SET BEGINNING POSITION OF A
%LOOPY B=-10000
*SET B EACH TIME THROUGH LOOP
PUTP A,B,0,1
*MOVE-NO-DRAW TO FIRST POINT
B=-B
PUTP A,B,0,0
*DRAW THE LINE
A=A+100
IF A LT 10000,SK %LOOPY
CLOSE>
```

THE ARGUMENTS TO PUTPOINT MAY BE EXPRESSIONS BUT THESE ARE EVALUATED INTEGER-WIZE, THAT IS, WITH TRUNCATION TO INTEGER VALUES AFTER EACH PART OF THE CALCULATION (SO .01*1000 EQUALS 0). YOU CAN USE FLOATING POINT CALCULATIONS TO GET AROUND THIS PROBLEM:

```
FA=.01*1000
FB=SIN(FK)*2000
PUTPOINT FA,FB,0,0
```

PUTDSK VERTS WILL STORE THIS ONE ON DISK UNDER VERTS.DEC. GETDSK VERTS WILL GET IT BACK (.DEC IS THE ASSUMED EXTENSION FOR GETDSK).

THE DELPOINT COMMAND WILL ERASE THE LAST PUTPOINT. A RUBBER BAND EFFECT WILL BE ACHIEVED BY THE FOLLOWING (THE /16 IS FOR SCALING PURPOSES):

```
PUTPOINT D0/16,D1/16,D2/16,0
IF FS1=0,DELPOINT;SK -1
FSOFF 1;SK -2
```

A COUPLE OF THINGS HERE--FIRST, NOTE THAT SEMICOLONS ARE USED TO PUT MULTIPLE COMMANDS ON A LINE. SECOND, NOTE THE USE OF THE FUNCTION SWITCHES. TRY THIS MACRO OUT. YOU MAY WANT TO TRY TO ADD A FUNCTION SWITCH TO CAUSE MOVE-NO-DRAWS, IN WHICH CASE YOU NEED A CURSOR MOVED ON D0,D1,D2 SO YOU CAN SEE THE MOVE-NO-DRAW POINT. YOU MIGHT WANT AN EXIT FUNCTION SWITCH TO DO A CLOSE ALSO. NOTE THAT TO SEE THE Z-COORDINATE, YOU WILL HAVE TO ROTATE THE PICTURE TOO, AND IF YOU WANT THE CURSOR TO BE ATTACHED, YOU WILL HAVE TO FIRST GROUP THE PICTURE AND THE CURSOR AND ROTATE THE GROUP (NOT THE PIX). SEE HELP GROUP FOR DETAILS. THIS TYPE OF THING GETS MILDLY COMPLICATED, BUT IS VERY INSTRUCTIVE IF YOU TRY IT YOURSELF.

C. CHANGING ENDPOINTS WITH GETPOINT AND ZAPPOINT

TWO COMMANDS, GETPOINT AND ZAPPOINT ALLOW YOU TO GET AT AND CHANGE INDIVIDUAL ENDPOINTS. THEY ARE COMPLIMENTARY AND FUNCTION LIKE READ AND WRITE. GETPOINT GETS VALUES INTO VARIABLES AND ZAPPOINT CHANGES ENDPOINTS ACCORDING TO THE VALUES GIVEN IN ITS VARIABLES. HELP GETPOINT AND HELP ZAPPOINT GIVE ALL THE STRAIGHT INFO.

THE SPECIAL THING IS THAT K (WHICH CAN ACTUALLY BE ANY FIXED VARIABLE), IS SET TO 0 FOR DRAWS, 1 FOR MOVE-NO-DRAWS, AND -1 FOR END-OF-LIST. FOR EXAMPLE, A MACRO TO MAKE THE FIRST AND LAST POINT OF A PICTURE THE SAME (THAT IS, CLOSE THE GAP BETWEEN THE FIRST AND LAST VECTORS) COULD BE WRITTEN AS FOLLOWS:

```
CLOSUP:<PROM "WHAT PIX TO BE DE-GAPPED"|
INPUT $N
N=1
GETP $N,1,A,B,C,D
N=N+1
GETP $N,N,X,Y,Z,K
IF K NE -1,SKIP -2
ZAPP $N,N,A,B,C,K
PROM "DONE">
```

AN EXAMPLE TO ADD 500 TO EACH Z IF THE X VALUE IS POSITIVE:

```
INCZ:<PROM "PIX NAME"|
INPUT $A
N=0
%MORE N=N+1
GETP $A,N,A,B,C,D
IF A GE 0,C=C+500
ZAP $A,N,A,B,C,D
IF D NE -1,SK %MORE>
```

MERGE.MAC 31,3 , JOIN2.MAC 31,3 AND SO ON.
// **********FORTRAN
FORTRAN

GRASS USES SIMPLIFIED FORTRAN-STYLE SYNTAX FOR DOING ARITHMETIC. THERE ARE SEVERAL IMPORTANT DIFFERENCES, THOUGH:

1. VARIABLES IN GRASS HAVE FIXED NAMES (SEE HELP DEV). YOU CANNOT HAVE A VARIABLE NAMED "RATE", FOR EXAMPLE. THIS DECISION WAS MADE WHEN DESIGNING GRASS AND IT ELIMINATES THE NEED TO TELL THE SYSTEM WHAT TYPE OF VARIABLE (FIXED, FLOATING, STRING, ARRAY, ETC.) YOUR ARBITRARILY-NAMED VARIABLE IS. IN GRASS, A-Z, VA-VZ, AND WA-WZ ARE FIXED POINT (INTEGER) VARIABLES WHOSE RANGE (MAXIMUM AND MINIMUM VALUES) IS 32767 TO -32768. VARIABLE WZ NORMALLY HAS 32767 IN IT SO IF YOU NEED TO KNOW WHAT FULL VALUE OF AN INTEGER VARIABLE IS, PROMPT WZ.
FA-FZ ARE FLOATING POINT VARIABLES WITH TREMENDOUS RANGE. MORE COMPUTATION IS REQUIRED TO GIVE THIS RANGE, HOWEVER, SO ONE NORMALLY USES FLOATING POINT VARIABLES (CALLED REALS IN FORTRAN) ONLY WHEN DOING FANCY CALCULATIONS.
THERE ARE ALLO VARIABLES KNOWN ONLY WITHIN MACROS.
THESE ARE CALLED LOCAL VARIABLES AND HAVE THE NAMES LA-LZ (INTEGER) AND EA-EZ(FLOATING). SEE HELP DEV FOR MORE INFO.
THE RULE IS THAT THE TYPE OF VARIABLE IS INDICATED BY ITS FIRST LETTER.
A LIST OF VARIABLES IS GIVEN IN HELP DEV AND HELP HELP

2. THERE ARE TWO BASIC TYPES OF ARITHMETIC STATEMENTS: ONES WHICH INTEGER DESTINATIONS AND ONES WHICH HAVE FLOATING DESTINATIONS. FOR REASONS OF EFFICIENCY, ASSIGNMENT STATEMENTS (ONES WITH '=' SIGNS) TO INTEGER VARIABLES (LIKE A=...) ARE PROCESSED DIFFERENTLY FROM ASSIGNMENTS TO FLOATING VARIABLES. FLOATING ASSIGNMENTS ARE MORE GENERAL; INTEGER ASSIGNMENTS ARE USED FOR COUNTING, ETC.

RULE 1. WHEN ASSIGNING TO AN INTEGER VARIABLE, TRUNCATION TO INTEGER VALUES IS DONE AFTER EVERY STEP OF THE CALCULATION. IN FLOATING ASSIGNMENTS, THE RESULT IS ALWAYS KEPT TO 8 SIGNIFICANT FIGURES. FOR EXAMPLE:

```
FD=.01
K=FD*100        EVALUATES TO 0 SINCE THE .01 IS
                TRUNCATED TO 0 FIRST.
FA=FD*100       EVALUATES TO 10 SINCE THERE IS NO
                TRUNCATION.
```

RULE 2. AS EXPLAINED IN HELP DEV, ONLY INTEGER VARIABLES, OR THINGS THAT EVALUATE TO INTEGER VARIABLES MAY BE DIRECTLY ATTACHED TO PICTURE TRANSFORMATIONS (WHICH IS WHAT GRASS AS A LANGUAGE IS MOST CONCERNED WITH). YOU CANNOT ROTATE SOMETHING ON A FLOATING VARIABLE WITHOUT EXPRESSLY CONVERTING IT TO AN INTEGER VARIABLE AS IN THE FOLLOWING:

```
ROTATE GLOBE,Y,P
%LOOP FP=SIN(FA)*1000
P=FP
FA=FA+.01
SKIP %LOOP
```

THE P=FP CAUSES A FLOATING-TO-INTEGER CONVERSION.

RULE 3. ONLY FLOATING ASSIGNMENTS CAN USE THE BUILT-IN FUNCTIONS (SIN, COS, ATN, EXP, ETC. EXPLAINED BELOW).
THIS IS ALSO FOR REASONS OF EFFICIENCY.

RULE 4. ALL EXTERNAL CONTROL DEV'S (D0-D9, TX,TY,TZ, ETC.) ARE INTEGER AND RANGE FROM 32767 TO -32768.
VARIABLE WZ IS SET TO 32767 TO HELP YOU USE THIS SILLY NUMBER (32768 IS REALLY 2 TO THE 15TH POWER).

RULE 5. BECAUSE OF DIGITAL WRAP-AROUND, 32767+1 EVALUATES TO -32768. YOU WILL NOTICE THE EFFECT OF THIS IF YOU TRY THE FOLLOWING MACRO:

```
MOVE ANYPIX,A
A=A+100
SKIP -1
```

THE MEANING OF WRAP-AROUND WILL BE REAL CLEAR.

RULE 6. DON'T PUT SPACES AROUND THE '=' SIGN. SPACES ANYWHERE ELSE ARE OK THOUGH.

3. IF YOU RUN OUT OF VARIABLES, OR NEED A WAY OF SPECIFYING SEQUENCES OF VARIABLES, USE ARRAYS (SEE HELP ARRAY).

4. PRECEDENCE OF OPERATORS IS AS NORMAL FOR ALGEBRA.
YOU MAY USE PARENTHESES TO CHANGE THE ORDER OF PRECEDENCE:

```
3+4*10 IS 43
(3+4)*10   IS 70
```

5. THE PROMPT COMMAND CAN BE USED TO TYPE OUT THE VALUE OF

```
            PROMPT FA
            PROMPT SQR(10000)

6. LOOPING ARITHMETIC STATEMENTS IS DONE JUST LIKE
      ALL OTHER LOOPS IN GRASS.  THE SKIP COMMAND IS USED
      FOR TRANSFER WITHIN A MACRO AND THE DO COMMAND IS USED TO
      CALL OTHER MACROS AS SUBROUTINES. (SEE HELP SKIP, HELP DO).

7. COMPILING OF MACROS (SEE HELP COMPIL) SPEEDS UP THE
      ARITHMETIC PROCESSING AS MUCH AS 200 TIMES.  COMPILING IS
      RECOMMENDED IF YOUR MACRO HAS LOTS OF ARITHTIC
      AND IS RUNNING TOO SLOW FOR YOUR TASTES.

8. THE AVAILABLE BUILT-IN FUNCTIONS ARE:
      1. SIN(EXPR)     WHERE EXPR IS IN RADIANS (3.14159 RADIANS
            EQUALS 180 DEGREES).
            EX: FA=SIN(2.7*FC)
      2. COS(EXPR)
            EX: FA=COS(SIN(2.7*FC))
      3. ATN(EXPR)     ARC TANGENT
            EX: FK=ATN(FE)
      4. LOG(EXPR)     .LOG TO THE BASE E
            EX: FD=LOG(1.7E14)
      5. ABS(EXPR)     ABSOLUTE VALUE
            EX: FF=ABS(FA)
      6. RND(EXPR)     RANDOM NUMBER
            EX: FT=RND(0)*1000
                  THE EXPR IS INCLUDED TO KEEP THE SYNTAX
                  CHECKER HAPPY BUT IS NOT USED.
      7. SGN(EXPR)     SIGN
            EX: FZ=SGN(RND(0))

FLOATING POINT NUMBERS MAY BE ENTERED IN E FORMAT:
      FD=1.2E7
      FE=-2E-10
THE LARGEST POSITIVE NUMBER FOLLOWING THE E IS 9808 AND THE  SMALLEST
IS -9808.

USE TRACE, PROMPT AND LIST TO HELP DEBUG YOUR ARITHMETIC PROGRAMS.


THAT'S ALL FOR NOW.
//    **********************************************************************
/*

THE EXPLANATION YOU REQUESTED DOES NOT EXIST (CHECK SPELLING).
//    **********************************************************************

//    ********************************************************************** ARRAY
ARRAY

SYNTAX:           ARRAY ARRNAME,EXPR
      OR          ARRAY ARRNAME,LOWER BOUND:UPPER BOUND
                        WHERE LOWER AND UPPER ARE EXPR'S

      THE GRASS ARRAY STATEMENT IS LIKE THE DECLARE STATEMENT IN
PL/1 AND LIKE THE DIMENSION STATEMENT IN FORTRAN.  ITS PURPOSE IS TO
SET UP THE ARRAY'S DIMENSIONS, THAT IS, HOW BIG AN AREA IN CORE YOU
WANT TO SET ASIDE FOR THE ARRAY.

      ARRAYS HAVE A SPECIAL SET OF NAMES: AA THROUGH AZ.

      THE NORMAL USE OF ARRAY ASSUMES YOUR ARRAY DIMENSIONS
START AT 1 AND GO TO THE EXPR.  CONSEQUENTLY,
      ARRAY AB,10
WILL DIMENSION AN ARRAY TO HAVE 10 INTEGER (LIKE A-Z TYPE)
ELEMENTS REFERENCED BY AB(1),AB(2),...,AB(10).  IN THE SECOND
CASE, FOR THOSE OF YOU WHO MUST HAVE UNNATURAL LOWER BOUNDS
TO YOUR ARRAYS, THE LOWER AND UPPER BOUNDS MUST BE EXPRESSLY
STATED.  NOTE THE SPECIAL SYNTAX--A ':' MUST SEPARATE THE BOUNDS.
      MULTI-DIMENSIONAL ARRAYS (UP TO FOUR) MAY BE CREATED TOO.
YOU SIMPLY INCLUDE THE DIMENSIONS FOR THE EXTRA COLUMNS
IN THE OBVIOUS WAY:
      ARRAY AQ,10,10,10
THIS WILL DIMENSION A FLOATING POINT ARRAY 10X10X10.
MULTI-DIMENSIONAL ARRAYS USE LOTS OF SPACE, SO BE ADVISED.
      THE ARRAY IS CONSIDERED FIXED POINT (INTEGER) IF ITS NAME
IS AA-AN.  THE ARRAY IS FLOATING POINT IF THE ARRAY NAME IS AC-AZ.
      THERE ARE ONLY ARITHMETIC ARRAYS; NO STRING ARRAYS EXIST
IN GRASS.
      STORING ARRAYS ON THE DISK REQUIRES ASSIGNING A NAME
WITH THE PUTDSK COMMAND:
      PUTDSK T1ARR.ARA,AK
NOTE THE SYNTAX. THE ARRAY NAME ON DISK IS T1ARR.ARA.
YOU COULD SAY PUTDSK AK.ARA,AK BUT YOU MIGHT NOT REMEMBER WHAT AK.ARA
WAS FOR.
      TO GET IT BACK INTO CORE, YOU TYPE:
```

GETDSK T1ARR,AP,AL
NOTE THAT IT CAN BE CALLED SOMETHING DIFFERENT THIS TIME, BUT YOU
CANNOT GET FIXED AND FLOATING MIXED UP SO THE FOLLOWING WOULD BE
UNWISE IN THIS CONTEXT:
GETDSK T1ARR,AP
SINCE GRASS WOULD ERRONEOUSLY THINK IT WAS A FLOATING ARRAY.
NOTE ALSO THAT YOU DO NOT USE THE ARRAY COMMAND PRIOR TO
GETDSK'ING AN ARRAY--GETDSK DOES THE DIMENSIONING AUTOMATICALLY.
EXAMPLES:

THIS SEGMENT OF CODE SETS UP THE ARRAY, AND THE LOOP FILLS
IT IN WITH THE INPUT COMMAND.

```
ARRAY AB,300
A=0
%MORE A=A+1      ;*FIRST ELEMENT
PROM "ENTER ARRAY ELEMENT #",A|
INPUT AB(A)
IF A LE 300,SK %MORE
PROM "DONE"
```

EXAMPLE2:

```
A=120
H=40
K=10
ARRAY AR,4:10,(A+H)/L
AR(4,0)=3.14159
AR(4,1)=2.789
AR(8,16)=.003
```

NOTE: OTHER OTHERWISE FILLED, ALL ARRAY ELEMENTS ARE SET TO 0.

// **********

/*
END OF A'S

// **********

// ********** BACKUP

BACKUP

SYNTAX: BACKUP OLDNAME,NEWNAME

OR

BACKUP OLDNAME

SWITCHES: /R

BACKUP CAUSES THE FILE SPECIFIED BY OLDNAME
TO BE WRITTEN ONTO DECTAPE (UNIT 0). IF A SECOND ARGUMENT
IS SUPPLIED THE NEW NAME ON TAPE WILL BE THAT NAME. IF NOT
THE NAME ON TAPE WILL BE THE SAME AS THAT ON DISK. THE /R SWITCH
REVERSES THIS TRANSFER CAUSING A FILE ON TAPE TO BE WRITTEN ONTO
DISK. THE GENERAL FORM OF THE FILENAMES IS FILNAM.EXT #,# . IF THE
#,# IS NOT SPECIFIED THE DEFAULT IS THE AREA INTO
WHICH THE USER IS CURRENTLY LOGGED.
BE SURE THE DECTAPE IS ON REMOTE WITH THE WRITE ENABLE SWITCH
ON AND THAT IT HAS BEEN INITIALIZED FOR YOUR AREA.

EXAMPLES: BACKUP ROBIN.DEC

CAUSES THE FILE ROBIN.DEC TO BE WRITTEN ON
TAPE WITH THE FILE NAME ROBIN.DEC

BACKUP ROBIN.DEC,JUNK.MAC

CAUSES THE FILE ROBIN.DEC TO WRITTEN ON
TAPE WITH THE FILE NAME JUNK.MAC

BACKUP/R ROBIN.DEC

CAUSES THE FILE ROBIN.DEC ON TAPE ALREADY TO
BE WRITTEN ON THE DISK. IT MUST NOT ALREADY
EXIST ON THE DISK

// **********BLANK

BLANK

SYNTAX: BLANK PIX ESOTERIC////

BLANK ALLOWS THE PIX TO BE UPDATED & GROUPED AND
SO ON BUT CAUSES THE VECTORS TO BE UNDISPLAYED SO THEY
DO NOT TAKE UP COMPUTER TIME TO DISPLAY. IT IS
ALSO USEFUL FOR FLASHING A PIX OFF AND ON. IT IS USEFUL SOMETIMES,
AS IN CERTAIN CASES OF PERSP, AND IN SYNCHRONIZING
ROTATIONS TO NOT HAVE THE PIX PUTLIB'D WHILE AT THE SAME
TIME, A SETI PIX,-32000 WOULD STILL POSSIBLY ADD TO FLICKER.
THERE IS ALSO A GETLIB/W OPTION NOW THAT ALLOWS YOU

FLASH YOU NORMALLY GET WHEN GETLIB'ING A PIX
UNDER THESE CIRCUMSTANCES. GETL/W ESSENTIALLY DOES A BLANK
AND THEN A BLANK/R AFTER 1/30 SECOND TO ALLOW ALL THE UPDATES TO
TAKE EFFECT. IF YOU HAVE NEVER RUN INTO THIS PROBLEM, DON'T
WORRY ABOUT IT, THOUGH.

SWITCHES: /R TURNS THE VECTORS BACK ON

EXAMPLES:
```
BLANK SAM
TICK 4
BLANK/R SAM
SK -3
```
THIS WILL FLASH SAM ON AND OFF 15 TIMES A SECOND
(ONE SECOND EQUALS 60 TICKS).

// ********** BLE
BLEND

SYNTAX: BLEND PIX1,PIX2,DEV,EXPR

BLEND TAKES TWO PIX AND DOES A LINEAR INTERPOLATION BETWEEN THEM.
THE NUMBER OF STEPS IN THE INTERPOLATION IS GIVEN BY EXPR. THE STEP
YOU ARE AT IS GIVEN BY THE DEV (WHICH FOLLOWS GRASS2 DEV CONVENTIONS).
THE PIX SHOULD HAVE THE SAME NUMBER OF VECTORS FOR BEST RESULTS, ALTHOUGH
THE WORST THAT SHOULD HAPPEN IS SOME RANDOM GARBAGE AT THE END
OF THE SMALLER PIX OR LOSING SOME OF THE LARGER PIX. THE DEV
VALUE NORMALLY RANGES FROM THE EXPR VALUE TO ZERO BUT MAY
EXCEED THESE BOUNDS IF YOU WISH, IN WHICH CASE THE EFFECT IS
WEIRD BUT OCCASIONALLY INTERESTING.
BLEND, ONCE STARTED FOR TWO PIX, CONTINUES TO OPERATE UNTIL
BLEND/U PIX1 IS TYPED. BLEND USES A LOT OF COMPUTER TIME, SO YOU
DON'T WANT TO HAVE IT OPERATING WHEN YOU DON'T NEED IT.
BLEND ALLOWS MANY PIX TO BE BLENDING INTO OTHER
PIX AT THE SAME TIME, USING MORE COMPUTER TIME, NATURALLY.
YOU MAY NOTICE THAT DIRCOR AND OTHER THINGS SLOW DOWN A LOT WHEN
LARGE PIX ARE BLENDED.
BLEND ALSO REQUIRES SCRATCH SPACE EQUIVALENT TO PIX1 WHILE
OPERATING TOO, SO REMEMBER TO BLEND/U WHEN IT'S OVER
BLEND ALSO WORKS WELL WITH THE TIME-BASED VARIABLES.

EXAMPLES:
```
BLEND SAM,TOM,D0,32767
```
THIS WILL BLEND SAM TO TOM ON DIAL0
```
QA=400;BLEND SAM,TOM,QA,400
```
THIS WILL BLEND SAM TO TOM IN 400 STEPS TAKING JUST
UNDER SEVEN SECONDS (SEVEN SECONDS = 420 TICKS).
YOU MAY HAVE TO REVIEW THE TIME BASED VARIABLES TO
UNDERSTAND THIS ONE.
```
BLEND SAM,TOM,100,200
```
THIS WILL BLEND SAM HALFWAY TO TOM AND STAY THERE.
```
BLEND SAM,TOM,A,1000
A=A-10
IF A GT 0,SK -1
A=A+10
IF A LT 1000,SK -1
SK -4
```
THIS WILL BLEND TOM TO SAM AND BACK UNTIL STOPPED.

NOTE THAT A DEV VALUE = ZERO MEANS ALL THE WAY TO THE
SECOND PIX, AND A DEV VALUE = THE EXPR MEANS PIX1
WILL LOOK LIKE PIX1.

SWITCHES: /U UNDO THE BLENDING
EX: BLEND/U SAM

// ********** BUMP
BUMP

SYNTAX: BUMP $VAR STRING MANIPULATION////

BUMP IS USED TO STEP THROUGH THE FIXED & FLOATING POINT
VARIABLE NAMES (A-WZ,FA-FZ,AA-AZ,LA-LZ). THE IDEA IS TO ALLOW
MACROS TO SETUP CODE TO MOVE, SCALE, ETC. A VARIABLE NUMBER OF
PICTURES ON VARIABLES. BUMP ONLY WORKS WITH $VAR'S WITH PROPER
VARIABLE NAMES IN THEM. WHEN $A='K', BUMP $A WILL CHANGE THE
'K' TO 'L' AND SO ON. 'Z' GOES TO 'VA' AND 'VZ' GOES TO 'WA'.

EXAMPLE:
```
PROM "NUMBER OF COPIES"
INPUT N
M=0
$B='VA'
M=M+1
$A='COPY',M
COPY NAME,$A
SETI $A,$B
BUMP $B
IF M LE N,SK -5
```

THIS WILL
```
SETINT COPY1,VA
SETINT COPY2,VB
```

ETC.

```
//    ************************************************************************
/*
```

THERE NO DESCRIPTION AVAILABLE FOR THIS COMMAND AT THIS TIME, OR
COMMAND NAME HAS BEEN MISSPELLED. ---- SORRY

```
//    ************************************************************************
//    ************************************************************************ CALL
```

CALL

SYNTAX: CALL MNAME

CALL IS LIKE 'DO' EXCEPT THAT IT REFERENCES THE COMMON AREA ONLY.
CALL, AS WELL AS OTHER COMMANDS, MAY BE USED AS AN ARGUMENT TO RESTART.

EXAMPLES:
```
CALL MERGE,PIX1,PIX2
CALL EDIT
RESTART CALL EDIT
```
NOTE THAT CALL AUTOMATICALLY GETS THE COMPILED VERSION OF THE
MACRO FIRST, IF IT EXISTS. TO FORCE USING THE MACRO VERSION
(NON-COMPILED) TYPE
```
DO MNAME.MAC 31,3
```

```
//    ************************************************************************ CUTOFF
```

CUTOFF

SYNTAX: CUTOFF PIX,DEV1,DEV2,DEV3,DEV4,DEV5,DEV6

CUTOFF IS A HARDWARE FUNCTION WHICH CUTS OFF THE
PICTURE EITHER OUTSIDE OR INSIDE A RECTANGULAR BOUNDARY SPECIFIED
BY THE SIX DEV'S. THE DEV'S REPRESENT THE X-HIGH, X-LOW, Y-HIGH,
Y-LOW, Z-HIGH AND Z-LOW BOUNDARIES IN THAT ORDER. THE DEFAULT VALUES
ARE:
```
CUTOFF PNAME,32767,-32768,32767,-32768,32767,-32768
```
CUTOFF EXPECTS SEVEN VALUES. IF YOU SPECIFY JUST ONE DEV LIKE
```
CUTOFF PNAME,D1
```
YOU WILL GET D1,D2,D3,D4,D5,D6 AND D7. YOU SHOULD SET
D1,D3,D5 TO FULL POSITIVE AND D2,D4,D6 TO FULL NEGATIVE TO SEE THE
PIX. THIS WILL SHOW PNAME AS IT APPEARS WITHIN THE BOUNDARIES
SET BY THE DIALS. OF COURSE, ANY LEGAL DEV'S ARE OK. (SEE HELP DEV).
THE /R SWITCH WILL SHOW THE PIX OUTSIDE THE BOUNDARIES.
NOTE: CUTOFF SEEMS TO SHIFT WHEN SETP IS USED.
ALSO: CUTOFF IS NOT SOFT'ABLE (SEE HELP SOFT). YOU SHOULD USE THE
WINDOW COMMAND FOR REAL CLIPPING (SEE HELP WINDOW).

SWITCHES: /R OUTSIDE INSTEAD OF INSIDE

EXAMPLES:
```
CUT GLOBE,,,,,,0
```
THIS WILL LEAVE THE DEFAULTS FOR ALL BUT Z-LOW
SO IT WILL SHOW HALF THE GLOBE (ALL POSITIVE Z-POINTS)
```
CUT BOX1,D0,=-D0,D0,=-D0,D0,=-D0
CUT/R BOX2,D0,=-D0,D0,=-D0,D0,=-D0
```
THIS SEQUENCE WILL DO A CENTER WIPE FROM BOX1 TO BOX2
(FIGURE THAT ONE OUT).

```
//    ************************************************************************CLEAR
```

CLEAR

SYNTAX: CLEAR

CLEAR SIMPLY CLEARS THE VT05 SCREEN.

```
//    ************************************************************************ CLOS
```

CLOSEO

SYNTAX: CLOSE

CLOSE ENDS THE PIX PREVIOUSLY SPECIFIED BY OPEN. CLOSE
ALLOWS THE PIX TO BE PUTDSK'S, IT RELEASES ANY SPACE NOT NEEDED,
AND IT ALLOWS ANOTHER PIX TO BE OPEN'D.

EXAMPLE:
```
OPENO SAM
PUTPOI 500,500,500,0
PUTPOI -500,500,500,0
PUTPOI 0,0,0,0
PUTPOI 500,500,500,0
CLOSEO
```
```
//    ************************************************************************ CLIP
```

CLIP

SYNTAX: CLIP CLIPEE,CLIPER,CLIPPED
WHERE CLIPEE,CLIPER, AND CLIPPED ARE PIX

IT IN THE X AND Y DIRECTIONS SO THAT IT FITS INSIDE ANOTHER PICTURE, THE "CLIPPER", AND PUTS THE RESULT IN THE NEW PICTURE, THE "CLIPPED".

THE "CLIPPER" IS ASSUMED TO BE TWO-DIMENSIONAL, HENCE THE Z-COORDINATES ARE IGNORED. IT IS ALSO ASSUMED THAT THE "CLIPPER" HAS NO JUMPS IN IT, AND THEREFORE IF THERE IS A JUMP, A LINE WILL BE ASSUMED IN THE GAP, ALSO, AND THIS IS IMPORTANT, THE FIRST POINT OF THE "CLIPPER" MUST BE THE SAME AS THE LAST POINT. BASICALLY THE "CLIPPER" CAN BE ANY OBJECT WITH A CLEARLY DEFINABLE INSIDE AND OUTSIDE WHETHER IT BE CONCAVE OR CONVEX, SUBJECT ONLY TO THE ABOVE RESTRICTIONS. SO FOR EXAMPLE, A FIGURE "8" DRAWN WITH A SINGLE BORDER IS ACCEPTABLE, BUT A FIGURE "8" DRAWN WITH A DOUBLE BORDER IS "BAD NEWS". THE OUTLINE OF A STAR IS ALSO ACCEPTABLE. HOWEVER, IF A LINE SEGMENT OF THE "CLIPEE" EXACTLY COINCIDES WITH A LINE SEGEMENT OF THE "CLIPPER", IT WILL BE EXCLUDED OR INCLUDED IN THE THE "CLIPPED" PICTURE DEPENDING ON THE JUDGEMENT OF THE MOON AND STARS, ASSUMING THAT THE CORRECT ATMOSPHERIC CURRENTS ARE PREVAILING.

NOTE: THAT THE THE ORGINAL PICTURE IS LEFT UNMOLESTED, AND THAT THERE ARE NO RESTRICTIONS WHATSOEVER THE CLIPEE

*-NOTE: AT THIS POINT IN TIME THERE EXISTS AN IMPLEMENTATION RESTRICTION IN THAT THE CLIPPER MAY NOT CONTAIN MORE THAT 100 POINTS. THE REST OF THE BIG CLIPPER'S POINTS WILL BE IGNORED

SWITCHES: /R - FOR REVERSE CLIPPING
THE CLIPPED PICTURE WILL CONTAIN ONLY THE LINES AND PORTIONS OF LINES LEFT OUT IN ORDINARY CLIPPING. NOTE THAT IF THE CLIPPER DOES NOT FOLLOW THE ABOVE-MENTIONED RULES, CERTAIN ANOMALOUS LINES WILL BE LEFT OUT IN BOTH TYPES OF CLIPPING.

// ********** COMPIL
COMPILE

SYNTAX: COMPILE MNAME,BNAME
OR COMPILE MNAME

COMPILE TRANSLATES MACROS INTO EXECUTABLE BINARY MACHINE INSTRUCTIONS, AND IS USED TO INCREASE THE EXECUTION SPEED OF A MACRO. MNAME IS ANY MACRO AND IT IS CALLED INTO CORE AUTOMATICALLY IF NECESSARY. BNAME IS THE NAME OF THE .CPL FILE GENERATED AND IT CONTAINS THIS BINARY MACHINE CODE. IF BNAME IS NOT SPECIFIED, THE .CPL FILE IS NAMED THE SAME AS MNAME (MNAME THE MACRO IS AUTO-DELETED IN ANY CASE).
BNAME CAN BE SAVED BY USING: PUTDSK BNAME.CPL
(THE .CPL MAY BE OMITTED, PUTDSK IS SMARTER THESE DAYS.)
TO RUN .CPL FILES THE EXECUTE COMMAND CAN BE USED.

NOTE: DON'T TRY TO TYPE OR PRINT .CPL FILES, THEY ARE NOT IN THE SAME FORMAT AS OTHER FILES.
THE ONLY COMMANDS FULLY COMPILABLE ARE ARITHMETIC ONES, GETPOI, PUTPOI, ZAPPOI, SKIP, FLOATING POINT STUFF, AND IF.

NOTE: ALL COMMANDS CAN BE COMPILED. ALL COMMANDS NOT LISTED ABOVE ARE NOT CONVERTED INTO TRUE MACHINE INSTRUCTIONS, SO COMPILING DOES SPEED UP PROCESSING MUCH.

EXAMPLE 1:
```
GLOB:<A=A+1;SKIP 0>
COMPILE GLOB,TEST
PUTDSK TEST.CPL
EXECUTE TEST
```

EXAMPLE 2:
```
COMPILE GLOB
PUTDSK GLOB.CPL
```

// ********** COPY
COPY

SYNTAX: COPY PIX1,PIX2

COPY CAUSES PIX2 (THE NEW NAME) TO SHARE THE DATA OF PIX1 (THE OLD NAME). ANY COMMAND THAT DOES NOT MODIFY DATA LISTS (E.G. ROTATE, SCALE, MOVE, ETC.) MAY BE THEN INDEPENDENTLY BE USED ON EITHER PIX. COMMANDS THAT ALTER DATA LISTS (E.G. SMOOTH, SOFTROT) WILL MODIFY BOTH COPIES. COPY MAY BE USED TO REFLECT A PICTURE AROUND AN AXIS BY USING THE SINGLE DIMENSIONAL SCALE.
A SURE WAY TO CRASH THE SYSTEM IS TO DELETE THE FIRST PICTURE (PIX1) BEFORE DELETING THE COPY (PIX2)--SO DON'T DO IT.
WE NOW HAVE NEW SWITCHES FOR COPY, SOME OF WHICH ACTUALLY COPY THE VECTORS TOO, SO, IN THESE CASES (SEE SWITCHES BELOW), ALL THE WARNINGS ABOUT DELETING THE FIRST PIX AND SOFT AND SMOOTH, ETC. DO NOT APPLY.

SWITCHES: /T COPIES LIKE COPY BUT ALSO COPIES ALL THE

/P PUTLIB'S THE COPY
/V COPIES THE VECTORS TOO (BUT NOT THE TRANSFORMATIONS
/A LIKE /V/T
/Q LIKE /V/P

THERE ISN'T ANY /T/P (DO YOU WANT IT??)

EXAMPLE:
```
COPY GLOBE,WORLD
ROTATE GLOBE,X,D0,D1,D2
ROTATE WORLD,Z,D9
SCALE WORLD,D6
```
THE ABOVE SHOW INDEPENDENT MOVEMENT OF COPIES
```
COPY/V SAM,TOM
SC TOM,0
SOFT TOM
PUTD TOM
```
THE ABOVE PUTS A HALF-SIZED VERSION OF SAM ON THE DISK UNDER THE NAME TOM

// ********** CRUNCH

CRUNCH ESOTERIC/////

SYNTAX: CRUNCH CPLNAME,PNAME,EXPR1,EXPR2,XVAR,YVAR,ZVAR

CRUNCH IS AN AUTOMATIC GETPOI/ZAPPOI COMBINATION. IT EXPECTS A COMPILED MACRO AS ITS FIRST ARGUMENT WHICH IS SUPPOSED TO AFFECT IN SOME WAY THE XVAR, YVAR AND ZVAR SPECIFIED BETWEEN THE POINT RANGE EXPR1 TO AND INCLUDING EXPR2.

EXAMPLE:
```
FOO:<X=X*2
Y=Y*3
Z=X+Y>
COMPILE FOO
CRUNCH FOO,CIRC9,5,100,X,Y,Z
```
THE ABOVE WILL DO THE SAME AS:
```
N=4
N=N+1
GETPOI CIRC9,N,X,Y,Z,K
ZAPP CIRC9,N,X*2,Y*3,X+Y,K
IF N LT 100,SKIP -3
```
EXCEPT THAT CRUNCH IS MUCH MUCH FASTER AND TAKES INTO ACCOUNT CASES WHEN EXPR2 IS GREATER THAN THE NUMBER OF POINTS. TO APPLY THE CPLNAME TO ALL THE POINTS, SPECIFY 9999 AS EXPR2. THE CPLNAME CAN, OF COURSE, BE STORED ON THE DISK; CRUNCH GETS IT AUTOMATICALLY.

// ********** CORE

CORE

SYNTAX: CORE
OR CORE,VAR1,VAR2

CORE PRINTS OUT THE TOTAL NUMBER OF FREE WORDS OF CORE AVAILABLE AND ALSO PRINTS THE SIZE OF THE FRAGMENTS. IF VAR1 IS SPECIFIED IT GETS THE TOTAL NUMBER OF FREE WORDS. IF VAR1 AND VAR2 ARE SPECIFIED VAR1 GETS THE TOTAL NUMBER OF FREE WORDS AND VAR2 GETS THE SIZE OF LARGEST FRAGMENT.

// **********
/*
THERE IS NO DESCRIPTION AVAILABLE FOR THIS COMMAND AT THIS TIME, OR COMMAND HAS BEEN MISSPELLED. ----SORRY.
// **********
// ********** DASHES

DASHES

SYNTAX: DASHES PIX

DASHES CHANGES THE PIX'S VECTORS TO DASH MODE.

SWITCHES:
NONE DASHES MODE
/R DOTS MODE (REALLY SHORT DASHES)

EXAMPLES:
```
DASHES LAMP
DASHES/R DIAMON
```
// ********** DELETE

DELETE

SYNTAX: DELETE ANAME

DELETES THE ANAME FROM CORE, REMOVES THE NAME AND RECLAIMS THE STORAGE THE ANAME TOOK.

SWITCHES:
NONE AS ABOVE
/D DELETES ON YOUR DISK AREA (REQUIRES EXTENTION).
/T DELETES ON YOUR TAPE AREA

DUE TO USE OF KEEP COMMAND OR "|C"'ING DURING
EXECUTION.

EXAMPLES:
DELETE GLOBE
DELETE JSDRAW
DELETE/D WITCH.DEC

// ********** DELPOI

DELPOI

SYNTAX: DELPOI
OR DELPOI EXPR

DELPOI DELETES THE LAST PREVIOUSLY PUT POINT IN AN OPEN'ED PICTURE. THE NAME OF THE PICTURE IS AUTOMATICALLY ASSUMED TO BE THE LAST PICTURED OPEN'ED.
IF AN EXPR IS INDICATED, THAT NUMBER OF POINTS IS BACKED UP.

EXAMPLE:
```
<PUTPOI D0/16,D1/16,D2/16,0
IF FS1=1, DELPOI
SKIP -2>
```
THIS CAUSES VECTORS TO BE DRAWN FROM THE POSITIONS OF DIALS 0,1, AND 2 UNLESS FS1 IS BEING HELD DOWN, IN WHICH CASE, A RUBBER-BAND EFFECT IS ACHIEVED.

EXAMPLE: DELP 5
THIS WILL DELETE THE LAST FIVE POINTS IN AN OPEN'D LIST.

// ********** DIRCOR

DIRCORE

SYNTAX: DIRCORE

DIRC GIVES THE USER A MAP OF WHAT'S GOING ON IN USER MEMORY. PICTURES, MACROS, .CPL'S, ARRAYS AND SO ON ARE LISTED WITH NOTES ON HOW MUCH SPACE THEY TAKE UP. THE TRANSFORMATIONS THAT ARE CURRENTLY ATTACHED TO PICTURES ARE ALSO LISTED.

// ********** DIRDSK

DIRDSK

SYNTAX: DIRDSK (OPTIONAL NAME) (OPTIONAL AREA)

DIRDSK GIVES A LIST OF STUFF THAT'S STORED ON THE DISK. THE OPTIONAL NAME (SEE EXAMPLES) ALLOWS ONE TO LOOK AT PART OF THE DIRECTORY. COMBINED WITH THE '*' FEATURE, ONE CAN SELECT PATTERNS OF AREAS OR NAMES.
IF THE OPTIONAL AREA IS SPECIFIED, IT MUST BE A VALID LOGIN AREA ON THE DISK.

SWITCHES:
/L DIRECTORY IS LISTED ON THE LINE PRINTER INSTEAD OF THE VT05.

EXAMPLES:
```
DI                    GIVES YOU A LOOK AT YOUR DISK DIRECTORY
DIRDSK *.DEC  1,1   LISTS ALL NAMES WITH A .DEC
EXTENSION IN AREA  1,1
DIRDSK S*.* LISTS ALL NAMES BEGINNING WITH S
DIRDSK P*.M* LISTS ALL NAMES BEGINNING WITH P
AND EXTENSIONS BEGINNING WITH M
```
// ********** DIRTAP

DIRTAPE

SYNTAX: DIRTAPE (OPTIONAL NAME) (OPTIONAL AREA)

DIRTAPE GIVES A DIRECTORY LISTING OF A DECTAPE ON THE VT05 SCREEN. THIS HAS THE "*" FEATURE SAME AS THE DIRDSK COMMAND. DEFAULT AREA IS THE USERS OWN TAPE AREA. BE SURE WHEN USING THIS COMMAND THAT THE DECTAPE IS ON REMOTE AND SET FOR UNIT 0.

EXAMPLES:
```
DIRTAPE
        THIS GIVES A DIRECTORY LISTING OF
        EVERYTHING ON THE USERS TAPE AREA.

DIRTAPE  *.DEC  1,1
        THIS GIVES A DIRECTORY LISTING OF
        OF ALL FILES WITH A .DEC EXTENSION
        IN TAPE AREA 1,1.
```
// ********** DO

DO

SYNTAX: DO MNAME
OR DO MNAME.MAC XX,XX
OR DO MNAME,ARG1,ARG2.....

DO IS USED FOR EXECUTING DISK- OR CORE-RESIDENT MACROS.
THE EXTENSION AND DISK AREA (XX,XX) MUST BE INCLUDED IF YOU

ARE CALLING A MACRO FROM SOMEONE ELSE'S AREA. DO'S MAY BE USED
AS OPERANDS TO THE RESTART COMMAND AND ISSUED WITHIN OTHER
MACROS IF DESIRED. ARGUMENTS MAY BE PASSED TO THE MACRO BY THE
USE OF ARG1,ARG2,.... (READ BY THE INPUT COMMANDS INSIDE THE MACRO).
IF ANOTHER MACRO IS CALLED THE OLD ARGUMENTS ARE LOST.

```
EXAMPLES:       DO FLUFF
                DO FLUFF.MAC 30,4
                DO FLUFF.MAC 31,3       (SAME AS "CALL FLUFF")
                DO FLUFF,10,22,SAM.DEC
                                (10 IS PASSED TO FIRST INPUT
                                 22 IS PASSED TO SECOND INPUT
                                 THE STRING SAM.DEC IS PASSED TO
                                 THE THIRD INPUT COMMAND, ANY
                                 FURTHER INPUTS WILL GO TO THE
                                 KEYBOARD.)
                               NOTE THAT PROMPTS ARE SUPPRESSED
                                     UNTIL THE ARGS RUN OUT.

//    ********************************************************************** DOLOOP
DOLOOP

SYNTAX:         DOLOOP MNAME1,MNAME2,....     ESOTERIC////
        OR      DOLOOP MNAME
        OR      DOLOOP <....
                       ...
                       ...>
```

DOLOOP IS AN ALTERNATIVE TO 'DO' AND IT RUNS THE MACRO
(OR .CPL) AS A BACKROUND JOB. WHEN USING DOLOOP, ALL MACROS
SHOULD BE DOLOOPED SINCE *-LEVEL & NORMAL MACRO LEVEL HAVE
PRIORITY. THINGS REQUIRING INPUT FROM THE TERMINAL IN A DOLOOP'ED
MACRO WILL STOP ALL MACROS, AS WILL SWAP MODULES WHILE THEY ARE
SWAPPING IN. DOLOOPED MACROS MAY HAVE SKIPS, BUT A SKIP AT THE END
BACK TO THE BEGINNING IS ASSUMED. TICK WILL WORK INDIVIDUALLY FOR
EACH MACRO. LOCAL VARIABLES SHOULD BE USED WHENEVER POSSIBLE.
AT LEAST BE VERY CAREFUL WHEN USING VARIABLES WITH DOLOOPING SO
YOU DON'T HAVE TWO MACROS CHANGING THE SAME VARIABLE UNINTENTIONALLY.
DOLOOPED MACROS ARE EASILY STOPPED BY |C.

```
SWITCHES:       DOLOOP/E MNAME,EXPR
                 DOES THE MACRO EXPR NUMBER OF TIMES
                DOLOOP/V MNAME,VAR
                 DOES THE MACRO UNTIL #OF TIMES = VAR

EXAMPLES:       DOLOOP <D=D+D5/32>
                DOLOOP SAM,FIRED,COMPACT
```

NAMED DOLOOPED MACROS MAY BE INDIVIDUALLY CANCELLED BY UNLOOP.

```
//    ********************************************************************
/*
 THERE IS NO DESCRIPTION FOR THIS COMMAND AT THIS TIME, OR
COMMAND HAS BEEN MISSPELLED.  ----SORRY.
//    ********************************************************************
//    ********************************************************************** EXECUT
EXECUTE

SYNTAX:         EXECUTE BNAME                 ESOTERIC/////
        OR      EXECUTE BNAME,ARG1,ARG2,.......
```

EXECUTE IS TO .CPL (COMPILED) MACROS WHAT "DO" IS TO
NORMAL MACROS. THE COMPILED MACRO IS GOTTEN FROM THE DISK (YOUR AREA
UNLESS SPECIFIED IN. XX,YY LOGIN CONVENTIONS) UNLESS IT'S ALREADY
IN CORE, AND THEN THE COMPILED MACRO IS RUN. THERE IS NO
EQUIVALENT TO GOTO FOR COMPILED MACROS, BY THE WAY.
|C AND |S CAN STOP EXECUTION OF COMPILED MACROS.

```
//    ********************************************************************EDIT
EDIT

SYNTAX:         EDIT MNAME
```

EDIT IS AN IN-CORE QUICKY EDITOR WHICH WORKS ONLY ON MACROS.
IT IS FAST, SMALL, AND HAS THE ADVANTAGE THAT YOU DO NOT HAVE TO RESTART
GRASS EACH TIME YOU WANT TO USE IT. MOREOVER, IT CAN BE ENTERED AND
RE-ENTERED VERY QUICKLY.
IT'S MAJOR DISADVANTAGE IS THAT IT DOES NOT DO EVERYTHING
THAT CALL EDIT DOES. FOR INSTANCE, YOU TYPE |C TO GET OUT OF THIS
EDIT, AND THE MACRO IS UPDATED ONLY IN CORE, AND NOT ON THE DISK.
FOR THIS REASON, WE HAVE IMPLEMENTED A NEW PUTDSK SWITCH--PUTDSK/D--
WHICH PUTS THE FILE ON THE DISK, FIRST CREATING A BACKUP
(.BAK) IF THE FILE ALREADY EXISTS ON DISK. THEREFORE, AS SOON AS
YOU HAVE A WORKING VERSION OF A MACRO, PUTDSK/D IT OR YOU WILL LOSE
IT UPON RESTART. ONE MORE THING: THIS EDIT WILL NOT CREATE MACROS,
IT WILL ONLY WORK ON MACROS THAT EXIST ON DISK OR IN CORE ALREADY.
EDIT WILL FETCH A MACRO FROM THE DISK IF IT IS NOT ALREADY
IN CORE, BY THE WAY.

```
TO TYPE THE WHOLE MACRO                          <CR>
TO TYPE LINE 100                                 100<CR>
TO TYPE LINES 50 TO 200                          50,200<CR>
TO DELETE LINE 20                                20-<CR>
TO CHANGE AAA TO JHKL IN LINE 80                 80/AAA/JHKL/<CR>
                                          OR     80 AAA JHKL <CR>
TO CHANGE ALL OF LINE 400                        400 THIS IS NEW LINE 400<CR>
TO INSERT AFTER LINE 30                          31 THIS IS A NEW LINE<CR>
TO EXIT FROM EDIT                                |C

SOME NOTES:
1. THIS EDIT RENUMBERS THE LINES AFTER EACH INSERT OR
   DELETE.  THIS MEANS THAT YOU SHOULD LIST THE LINES YOU WANT
   TO PLAY WITH AFTER EACH INSERT OR DELETE BECAUSE THE LINE NUMBERS
   WILL SHIFT.  HOWEVER, YOU CAN MULTIPLY DELETE LINES
   20 THRU 40 BY TYPING "20-<CR>" THREE TIMES (TRY IT).
   YOU CAN ALSO MULTIPLY INSERT AFTER LINE 20 BY TYPING:
        21 THIS IS LINE AFTER 20
        31 THIS IS THE NEXT LINE
        41 THIS IS THE NEXT LINE
   ETC.  THIS MAY LOOK WEIRD BUT IT'S EASY TO GET USED TO.

2. LOCAL VARIABLE TABLES ARE ZEROED BY THIS EDITOR.  OTHERWISE, IT
   IS POSSIBLE TO EDIT MACROS OVER AND OVER AGAIN WITHOUT
   RESTARTING, WITH NO SIDE EFFECTS.

3. IF YOU ARE TIGHT ON SPACE AFTER USING THIS EDIT, DEL/C EDIT TO
   GET RID OF IT
//   ********************************************************************** EXIT
EXIT

SYNTAX:          EXIT

        EXIT IS SIMPLY A PROGRAMMED |C.  IF YOUR MACRO SEES AN
"EXIT" COMMAND, THE SYSTEM RETURNS TO *-LEVEL DOING WHATEVER A |C WOULD.
//   **********************************************************************
/*
WE DON'T HAVE THAT ONE AROUND....COMPLAIN LIKE HELL
//   **********************************************************************
//   ********************************************************************** FILMIN
FILMING

SYNTAX:          FILMING                              ESOTERIC/////
        OR       FILMING EXPR
                 FILMING/M EXPR

        FILMING IS A COMMAND WHICH ALTERS THE SPEED OF THE UPDATES
OF ROTATIONS, MOVES, SCALES, TIME-BASED VARIABLES, DIAL
READINGS, VIP'ED CPL'S, ETC. FILM (WITHOUT ANY EXPR) STOPS
UPDATES AND WILL CONTINUE ONLY WHILE FS1 IS HELD DOWN.
"FILM EXPR" CAUSES THE UPDATES TO BE DONE WITH
TIME DELAYS EQUAL TO THE NUMBER OF TICKS SPECIFIED BY THE EXPR.
SO FILM 2 WILL UPDATE EVERY OTHER 1/60 SEC (30 TIMES A SECOND)
AND FILM 60 WILL UPDATE ONCE A SECOND.  THIS COMMAND FEATURE IS
USEFUL TO CAUSE MOTIONS TO APPEAR TO STEP, OR, IS OCCASIONALLY
USEFUL IF TOO MUCH BLENDING, ROTATIONS ON GROUPS, ETC.,
CAUSE THINGS LIKE DIRCOR OR PUTDSK TO OPERATE TOO SLOWLY.

        FILM/M EXPR IS USED TO TRIGGER THE MOVIE CAMERA SITTING
ON TOP OF THE SCOPE.  YOU SHOULD TALK TO TOM ABOUT USING THE
CAMERA SINCE IT IS A PRIMITIVE, SILVER-EATING THING
WHICH IS NOT REAL-TIME IN OPERATION.  FILM/M DOES A
PRETTY GOOD JOB OF SLOWING DOWN PROPERLY WRITTEN MACROS
SO IMAGES RECORD WELL ON FILM.

EXAMPLES:        FILM
                 FILM/M 100
//   ********************************************************************** FIX
FIX

SYNTAX:          FIX PNAME

        FIX FREEZES THE POSITION OF A PICTURE OR THE VALUE OF A PNAME'S
MODIFIER ACCORDING TO THE SWITCH OPTIONS.  NO SWITCH FIXES EVERYTHING.
FIX ALSO REMOVES DEV ASSIGNMENTS IF ANY.

SWITCHES:
        NONE     FIXES ALL OF THE BELOW
        /R       FIXES ROTATION
        /S       FIXES SCALE
        /M       FIXES MOVE
        /P       FIXES PATHMOV
        /I       FIXES INTENSITY
        /Q       FIXES Z-AXIS CUEING (SETCQ)
        /C       FIXES CUTOFF PLANES
        /O       FIXES SETORG

EXAMPLES:        FIX/R CARTER
```

```
//   ********************************************************************** FSOFF
FSOFF

SYNTAX:          FSOFF   EXPR1,EXPR2,EXPR3, ETC.

        FSOFF TURNS OFF THE FUNCTION SWITCHES CORRESPONDING TO THE
EXPRESSIONS.  FSOFF ALONE TURNS OFF ALL THE SWITCHES.
        WHEN A FUNCTION SWITCH IS TURNED OFF, ITS VALUE IS 0.

EXAMPLE:         FSOFF 1,3,5,7,9,11,13,15
        THIS TURNS OFF THE ODD FUNCTION SWITCHES.
                 FSOFF
        THIS TURNS OFF ALL THE SWITCHES.
//   ********************************************************************** FSON
FSON

SYNTAX:          FSON EXPR1,EXPR2,EXPR3....

        FSON TURNS ON (SETS TO ONE) FUNCTION SWITCHES CORRESPONDING
TO EXPR1,EXPR2, ETC.  FSON DOES NOT WORK WITH THE PANIC BUTTON.

EXAMPLE:         FSON 0,2,4,6,8,10,12,14
        THIS LIGHTS UP THE EVEN SWITCHES AND SETS THEM TO EQUAL 1.
//   **********************************************************************
/*
SOMETHING IS WRONG HERE, WE DON'T SEEM TO HAVE THE HELP ON THE COMMAND
THAT YOU TYPED IN. HMMMMMMM, YOU SURE YOU DIDN'T SPELL THE NAME
WRONG??????
//   **********************************************************************
//   ********************************************************************** GETDSK
GETDSK

SYNTAX:          GETDSK DNAME.EXT XX,XX
        OR       GETDSK DNAME.EXT XX,XX ,EXPR1,EXPR2,VAR

        GETDSK GETS THE DNAME FROM THE DISK AREA INDICATED BY " XX,XX "
(DEFAULT IS YOUR AREA).  IF NO .EXT, IT DEFAULTS TO .DEC.
DNAME BECOMES THE NAME OF THE THING YOU ARE GETDSK'ING. IF EXPR'S ARE
GIVEN, THEY GET LINES STARTING AT LINE  EXPR1  UP TO AND INCLUDING
LINE  EXPR2 IN THE FILE. IT RETURNS A VALUE OF IN VAR IF THERE IS MORE
AND A  VALUE OF 1 IF END OF FILE.  GETDSK WITH EXPR'S IS      ESOTERIC/////
EXTENSIONS:
        .DEC     STANDARD PICTURE MODE (DEFAULT)
        .CPL     COMPLD FORMAT (FOR BINARY CODE.  DON'T TYPE OR
                               PRINT THESE FILES, WEIRD THINGS
                               HAPPEN.)
        .MAC     MACRO FORMAT (FOR MACROS, NOT FOR PICTURES)
NOTE THAT .DEC TYPE PICTURES COME UP ON THE SCREEN.

SWITCHES:        /P       (FOR .DEC ONLY) PICTURE IS PUTLIB'D
                          FIRST. WITH THIS SWITCH, ONE
                          CAN GET A PICTURE, ROTATE IT, ETC.,
                          AND THEN GETL/W IT SO IT WILL APPEAR
                          IN THE RIGHT PLACE WHEN FIRST SEEN BY
                          WHOEVER IS IN FRONT OF THE VG.
                 /M - IGNORES EXTENTION CHECKING AND GETS
                      THE FILE AS IF IT WERE A MACRO.IT ALSO
                      ALLOWS REMARKS TO BE LEFT IN THE MACRO.
                      THIS IS USEFUL FOR HANDLING .DEC FILES IN
                      ASCII.

NOTE: GETDSK NORMALLY STRIPS OFF ALL LINES BEGINNING
WITH THE CHARACTER '*' IN .MAC FILES, SO
COMMENTS MAY BE LIBERALLY PROVIDED IN YOUR MACRO WITHOUT
WORRYING THAT THE MACRO WILL TAKE UP TOO MUCH SPACE.


EXAMPLES:        GETDSK WITCH
                 GETDSK WITCH.DEC
                 GETDSK WITCH.DEC 30,10
                 GETDSK DRAW.MAC
                 G CALCUL.CPL
                 G FORD.ARA,AH              (PUTS FORD.ARA INTO ARRAY AH
                                            NO DIMENSIONING NECESSARY.
                                            NOTE THAT THE ARRAY COMMAND
                                            IS ONLY FOR SETTING UP NEW ARRAYS.
                 G/P BLOB
                 ROT BLOB,Y,D0
                 SC BLOB,15000
                 MOV BLOB,TX
                 GETL/W BLOB                AUTO-PUTLIB FEATURE
                                            USED INTELLIGENTLY.
//   ********************************************************************** GETHIT
GETHIT

SYNTAX:          GETHIT PLX,N,Y,X,Z,K
```

GETHIT RETURNS THE STATUS OF A LIGHTPEN HIT WHERE N GETS THE NUMBER OF THE POINT IN PIX, X,Y,Z GET THE RESPECTIVE COORDINATES, AND K IS SET TO ZERO IF IT IS A DRAWN POINT, ONE IF IT IS THE INITIAL POINT OF A DRAWN VECTOR (I.E.FOLLOWING A JUMP), AND MINUS ONE IF IT IS THE LAST POINT IN THE PIX.

EXAMPLE: GETHIT PANAM,F,U,C,K,Q
(AN EXAMPLE IN THE HUMOR STYLE OF MAINE EAST.)

// ********** GETLIB
GETLIB

SYNTAX: GETLIB PNAME ESOTERIC/////
OR GETLIB PNAME,PNAME2

GETLIB RETRIEVES THE PNAME FROM THE NON-DISPLAYED IN-CORE PICTURE LIST AND DISPLAYS IT. IF PNAME2 IS INDICATED, IT IS PUT IN THE TREE AFTER PNAME1 (SEE HELP TREE, HELP GROUP). IF PNAME2 IS NOT SPECIFIED, PNAME BECOMES THE FIRST ELEMENT TREE.

NOTE THAT PNAME2 MUST BE DISPLAYED AND PNAME MUST HAVE BEEN PUTLIB'ED AT SOME TIME (BY PUTLIB OR GETDSK/P OR COPY/P). GETLIB AND PUTLIB ARE THE ONLY WAY TO MOVE THINGS AROUND IN THE TREE STRUCTURE.

EXAMPLES: GETLIB SAM
GETLIB MARY,SAM

// ********** GETPOI
GETPOIN

SYNTAX: GETPOIN PIX,N,X,Y,Z,K
WHERE N IS AN EXPR AND X,Y,Z,K ARE VARS

GETPOIN GETS A POINT FROM AN LNAME. THE X,Y,Z COORDINATES OF THE NTH POINT ARE RETURNED IN VARIABLES INDICATED HERE BY X,Y,Z. N RANGES FROM THE 1 (THE FIRST POINT) TO THE LAST POINT IN THE LNAME (WHICH DEPENDS ON HOW BIG THE PICTURE IS). K IS A VARIABLE IN WHICH THE FOLLOWING IS INDICATED:

K=0 DRAWN VECTOR
K=1 NON-DRAWN VECTOR (JUMP) THAT IS, NO LINE DRAWN TO THIS POINT FROM THE LAST POINT.
K=-1 END OF LIST (LAST VECTOR)

SWITCHES: /N - GETS POINTS SEQUENTIALLY FROM A PICTURE. ONCE GETPOIN IS EXECUTED, THEN /N GETS NEXT POINT(S) & USES THE SAME VARIABLES. THIS HAS NO ARGUMENTS. IT ALSO SPEEDS UP PROCESSING GREATLY

EXAMPLE:
GETPOIN WITCH,20,G,H,I,K
THIS WILL GET THE COORDINATES IN DECIMAL OF THE TWENTIETH VECTOR IN WITCH AND PLACE THEM IN VARIABLES G,H,I AND INDICATE WHETHER THIS LINE WAS DRAWN, A JUMP OR THE END OF THE LIST, IN VARIABLE K.
THE APPLICATION FOR THE SWITCH IS:

```
<N=0
 N=N+1
 GETPOIN FL,N,X,Y,Z
 SK -2>
```

THIS MACRO WILL GET ALL THE POINTS IN THE PIX CALLED FL, BUT IT CAN BE DONE FASTER WITH THIS MACRO:

```
<GEPOIN FL,N,X,Y,Z
 GETPOIN/N
 SK -1>
```

IN BOTH CASES, THE MACRO WILL GIVE AN ERROR MESSAGE AT THE END OF THE PIX LIST. THE SECOND WILL EXECUTE MUCH FASTER SINCE IT ALREADY KNOWS WHICH VARIABLES TO USE. GETP/N DOES NOT WORK WELL WITH DOLOOPING, THOUGH.

A WAY TO GET TO THE END WITHOUT GETTING AN ERROR MESSAGE, BY THE WAY, IS:

```
<N=0
N=N+1
GETP PIX,N,X,Y,Z,K
IF K#-1,SK -2
PROM "PIX HAS ",N," VECTORS">
```

// ********** GOTO
GOTO

SYNTAX: GOTO MNAME+EXPRESSION ESOTERIC/////

GOTO IS USED TO TRANSFER CONTROL TO A MACRO WHEN YOU NEVER WANT TO RETURN TO THE MACRO THAT THE

DON'T DO IT UNLESS YOU REALLY HAVE TO.
// ********** GROUP
GROUP

SYNTAX: GROUP PNAME1,PNAME2,GNAME

GROUPS PNAME1, PNAME2 AND EVERYTHING BETWEEN THEM INTO A GROUP NAMED GNAME. PNAMES MUST BE DISPLAYED. PUTLIB/GETLIB SEQUENCES MAY BE USED TO ALTER GROUP STRUCTURE BY ELIMINATING OR ADDING PICTURES IF NECESSARY. ANY COMMAND WHICH WILL WORK ON PNAMES WILL WORK ON GROUPED PICTURES (EXECPT DELETE). CHECK "TREE" BEFORE GROUPING TO MAKE SURE THE PNAMES ARE ON THE SAME LEVEL & THE RIGHT STUFF IS GROUPED.
GROUP WORKS WITH BLANK'D PIX TOO.

THE COMMANDS WHICH WORK WITH GROUPS ARE:
GROUP CUTOFF FIX GETLIB MOVE PATHMOV PUTLIB RESET
ROTATE SCALE SETCQ SETINT SETORG

WITH GETLIB AND PUTLIB, YOU CAN MOVE PIX & GROUPS AROUND IN THE TREE DATA STRUCTURE.
FOR EXAMPLE, IN

```
GETD PLANE
GETD PROP
GROUP PROP,PLANE,SAM
```

THE TREE LOOKS LIKE:

```
SAM
    PROP
    PLANE
```

NOW A PUTLIB PROP WILL UN-DISPLAY IT AND TAKE IT OUT OF THE GROUP SAM. A GETLIB PROP WILL RE-DISPLAY IT BUT IT WILL NOT BE IN SAM. TO GET IT BACK IN SAM, YOU HAVE TO GETLIB PROP,PLANE.
FURTHERMORE,

```
GETD/P WHEELS
GETL/W WHEELS,PLANE
```

WILL GET WHEELS UNDER SAM WITHOUT ANY UNSIGHTLY FLASHING (DUE TO THE GETL/W WHICH WAITS FOR THE TRANSFORMATIONS TO TAKE EFFECT BEFORE DISPLAYING).

// **********
/*
GEE WHIZ, WHAT CAN I SAY EXCEPT THAT THERE ISN'T ANY INFO ON THE COMMAND THAT YOU TYPED IN, MAKE SURE THAT YOU SPELLED IT RIGHT. TRY IT AGAIN.
// **********
// ********** HELP
HELP

SYNTAX: HELP COMMAND-NAME

HELP IS HERE TO ANSWER SYNTAX QUESTIONS.
CERTAIN ABBREVIATIONS HAVE BEEN USED WHICH WILL, HOPEFULLY, AID UNDERSTANDING:

```
ANAME    IS ANY PICTURE OR MACRO NAME
PNAME    IS ANY PICTURE NAME (GROUP OR SINGLE PICTURE)
PIX      IS ANY SINGLE PICTURE
GNAME    IS ANY GROUP NAME
MNAME    IS ANY MACRO NAME
DNAME    IS ANY DISK FILE NAME
ARRNAME  IS ANY ARRAY NAME
CPLNAME  IS ANY COMPILED MACRO IN BINARY FORM
<CR> IS THE SYMBOL FOR CARRIAGE RETURN, THE KEY YOU MUST PUSH
TO END LINES YOU TYPE ON THIS TERMINAL.
```

NAMES IN CORE MAY BE ABBREVIATED TO ENOUGH LETTERS TO INSURE UNIQUENESS.

>>>>>>>>DEV IS ONE OF THE FOLLOWING ANALOG DEVICES OR VARIABLES

```
DIALS 0-9        (D0-D9)
SLIDE POTS       (S0 - S9)
TEN TURN POTS    (P0 - P3)
TABLET           (TX, TY AND TZ)
JOYSTICKS        (JX,JY,JZ) AND (KX,KY,KZ)
VARIABLES        (A-Z)
                 (VA -VZ)
                 (WA - WZ)
                 (FA - FZ)
                 (SA-SH)          SINE VARS
                 (CA-CH)          COSINE VARS
                 (LA-LZ)          LOCAL FIXED VARS
                 (EA-EZ) LOCAL FLOATING VARS
                 (OA-OH) ANALOG OUTS
                 PLUS THE TIME-BASED VARIABLES
```

SEE HELP DEV FOR HOW TO USE VARIABLES.

>>>>>>>VAR IS A DEV THAT DOES NOT OBEY DEVICE CONVENTIONS AS OUTLINED IN HELP DEV. VARS ARE USED WHEN
SOMETHING HAS TO BE RETURNED IN A VARIABLE

OR THE COMMAND IS NOT SMART ENOUGH TO ACCEPT
FULL-BLOWN DEV'S FOR SOME REASON.

>>>>>>>EXPR IS A MIX OF NUMBERS,DEVS AND ARITHMETIC OPERATORS WHICH
ALWAYS EVALUATES TO A SINGLE NUMBER.
THE OPERATORS ARE +,-,*,/.
EXAMPLES: A
D7
200
A+(D7/200)*35-B-(K/17)
THERE IS OPERATOR PRECEDENCE WHEN EXPRESSIONS
ARE EVALUATED I.E., EXPONENTIATION IS DONE FIRST
(ONLY FOR USE WITH FLOATING POINT NUMBERS),
MULTIPLICATION AND DIVISION ARE DONE NEXT, FINALLY
ADDITION AND SUBTRACTION ARE PREFORMED.
PARENTHESIS CAN BE INSERTED AROUND PARTS OF AN
EXPRESSION SO THAT DESIRED OPERATIONS WILL BE
PERFORMED FIRST. THUS THE FOLLOWING EXPRESSION
(52+8)/2+12 EQUALS 42.
THERE IS A MOD FUNCTION "%" SO 10%4 EVALUATES TO 2.
SEE HELP FORTRAN FOR MORE DETAILS ON ARITHMETIC STATEMENTS.

>>>>>>>$VAR IS A STRING VARIABLE AND HOLDS TEXT.
STRING MANIPULATION IS COMPLICATED EXCEPT
FOR THIS CASE:
PROM "WHAT'S THE PIX NAME"
INPUT $A
GETDSK $A
ROTATE $A,X,D0,D1
THIS EXAMPLE USES $A AS A DUMMY SO MACROS CAN BE GENERAL
PURPOSE. SEE HELP MACROS FOR MORE DETAILS.

DOCUMENTATION CONCERNING SYNTAX FOR A PARTICULAR COMMAND MAY
BE RETRIEVED BY TYPING HELP FOLLOWED BY THE COMMAND NAME:
HELP ROTATE
WILL GET INFORMATION ON THE ROTATE COMMAND.

THERE ARE ALSO SYSTEM MACROS FOR MOST COMMANDS SO IF YOU DO
NOT UNDERSTAND THE DOCUMENTATION IN HELP, A MACRO WILL STEP YOU
THROUGH MOST COMMANDS. CALL CLIP, FOR INSTANCE, WILL
RUN YOU THROUGH THE CLIP COMMAND.

IF YOU GET AN ERROR, TYPE A "?" FOLLOWED BY A CARRIAGE RETURN
(AGAIN, REFERRED TO AS <CR>). THE ERROR MESSAGE WILL BE PRINTED OUT.

COMMANDS MARKED "(STRING MANIPULATION)" AND " ESOTERIC/////"
ARE BEST SKIPPED BY NOVICES.

IF THE HELP DOCUMENTATION DOES NOT MAKE SENSE, PLEASE MAKE A
LOT OF NOISE ABOUT IT. IT'S HARD TO GUESS WHAT YOU DON'T UNDERSTAND.

THE HELP DOCUMENTATION IS MOSTLY FOR SYNTAX QUESTION. SOME ADDITICNAL
CLUES ON HOW TO WRITE GRASS PROGRAMS ARE CONTAINED IN HELP FORTRAN,
HELP DEV, HELP PIX, AND HELP MACROS.

NOTE THAT THE CNTL W BUTTON WILL HALT A HELP PRINTOUT UNTIL YOU HIT
CNTL Q. YOU TYPE CNTRL Q AND THE OTHER CONTROL FUNCTIONS BY HOLDING
DOWN THE CTRL BUTTON AND TYPING THE LETTER KEY. IT WORKS JUST LIKE
A SHIFT KEY ON A TYPEWRITER.

THE CNTL S BUTTON WILL HALT EXECUTION OF A MACRO AND PUT THE USER
IN THE #-SIGN MODE. IN THIS MODE THE USER CAN PERFORM ANY OF THE
SYSTEM COMMANDS. TO RETURN TO MACRO EXECUTION, TYPE 'RESUME'.

REMEMBER TO USE THE CNTL C BUTTON IF THINGS GET OUT OF HAND, AND THAT
THE WORST YOU CAN DO IS STALL THE COMPUTER.

```
//    ************************************************************************
/*
 THERE IS NO DESCRIPTION AVIALABLE FOR THIS COMMAND AT THIS TIME, OR
COMMAND HAS BEEN MISSPELLED.  ----SORRY.
//    ************************************************************************
//    ************************************************************************ IF
IF
```

SYNTAX: IF VAR OPR EXPR,COMMAND
OR IF FS# OPR EXPR,COMMAND
OR IF $VAR OPR ALPHA-EXPR,COMMAND

IF IS THE SYSTEM'S CONDITIONAL FOR USE WITHIN MACROS. THE
CRYPTIC SYNTAX ABOVE IS BECAUSE THE IF STATEMENT IS SO FLEXIBLE. DEV
AND EXPR ARE DEFINED IN "HELP HELP"
OPR CAN BE EQ, NE, LE, LT, GE, GT, =, OR # (NOT EQUALS)
FS# IS FS0 - FS15 FOR FUNCTION SWITCHES (0=OFF, 1=ON)
$VAR IS $A - $Z
ALPHA-EXPR IS EITHER $A-$Z OR 'ANY STRING IN QUOTES'. IF ONE OF THE
CHARACTERS IS A "*", THEN ALL STRINGS MATCHING THE CHAPACTERS UP TO

THE COMMAND (STUFF FOLLOWING THE ',') CAN BE ANY LEGAL LINE OF
GRASS STATEMENTS.

```
EXAMPLES:          IF FS1=1,DO PRETTY
                   IF FS1 EQ 1,DO PRETTY   (SAME THING AGAIN)
                   IF DO LT D9/22,A=A+900
                   IF QA GT 0,SK 0
                   IF TX GT 0,IF TY GT 0,PUTP TX,TY,0,0
                   IF $A=$B,SKIP 57
                   IF $A='YES',SKIP %YES
                   IF $A#'Y*',SKIP %NO
                   IF $B='TIGER',FSON FS1;DO TIGMAC
NOTE THAT     IF FA+32 GT B+10    IS ILLEGAL. YOU HAVE TO CODE IT LIKE:
                   IF FA GT B+10-32,...   ONLY VARS ARE ALLOWED ON THE
                   LEFT SIDE OF THE OPR.

//   ********************************************************************** INPUT
INPUT

SYNTAX:            INPUT VAR
          OR       INPUT $VAR
```

INPUT IS THE COMMAND YOU USE IN A MACRO TO GET A RESPONSE FROM THE TELETYPE. YOU HAVE THE CHOICE OF INPUTTING NUMBERS INTO FIXED OR FLOATING VARIABLES, OR INPUTTING NAMES OR OTHER CHARACTER STRINGS INTO $VARIABLES.

INPUT CAN BE FAIRLY ESOTERIC IN USE. FOR EXAMPLE, WHEN USING THE ARGUMENT FEATURE OF DO, CALL OR EXECUTE, THE ARGUMENTS ARE AUTOMATICALLY FED VIA THE INPUT COMMAND INTO THE MACRO. THIS MEANS THAT YOU CAN TEST THE MACRO INTERACTIVELY AND AND THEN USE IT AS PART OF ANOTHER MACRO WITHOUT REWRITING ANYTHING EXCEPT THE DO, CALL OR EXECUTE COMMAND LINE. INPUT IS NORMALLY PREFACED WITH A PROMPT SO THE USER KNOWS WHAT IS EXPECTED.

INPUT/0 AND INPUT/1 THROUGH INPUT/9 ARE ESOTERIC TOO. INPUT/0 WILL GET THE LAST CHARACTER TYPED AND WILL NOT WAIT. IN THIS MODE, INPUT OFF THE KEYBOARD MAY BE USED LIKE FUNCTION SWITCHES. INPUT/1 WILL WAIT FOR ONE CHARACTER TO BE TYPED AND THEN GRAB CONTROL BACK WITHOUT WAITING FOR A <CR>. INPUT/5 WILL WAIT FOR FIVE CHARACTERS, ETC. NORMAL INPUT COMMAND USAGE ALWAYS RESULTS IN THE SYSTEM WAITING FOR A <CR>.

INPUT OF NUMBERS MAY BE SEPARATED BY COMMAS IN BOTH THE INPUT COMMAND AND/OR THE RESPONSE. IF ENOUGH INFORMATION IS SUPPLIED, THE MACRO CONTINUES; OTHERWIZE THE SYSTEM WAITS.

FOR EXAMPLE:

```
PROM1:  <PROM "ENTER THE TWO NUMBERS"
        INPUT A,B
        PROM "THE SUM IS ",A+B>
```

SWITCHES:

ALL THESE SWITCHES ARE ESOTERIC/////

- /L - TAKES ARGUMENT AS PASSED FROM DO AND THEN FLUSHES THE REST. (USED TO PREVENT CASE OF USER PASSING TO MANY ARGUMENTS.)
- /O - INPUT VALUE IN OCTAL (BASE EIGHT)
- /T - TERMINATES INPUT FROM ARGUMENT LIST AND SAYS "TAKE NEXT INPUT FROM TTY AND TURN PROMPTS BACK ON"
- /0-/9 AS EXPLAINED ABOVE

```
EXAMPLES:    INPUT A          - WHERE A IS A NUMERIC
             INPUT $B         - WHERE $B IS A CHARACTER STRING
             INPUT FC         - WHERE FC IS A FLOATING POINT NUMBER

            *<PROMPT "WHAT PICTURE ?"
            +INPUT $A
            +ROTATE $A,X,DO>

//   ********************************************************************
/*
 YOU ARE OUT OF LUCK, THERE IS NO HELP ON THAT COMMAND. CHECK YOUR
SPELLING, BY TRYING IT AGAIN, OK??
//   ********************************************************************
//   ********************************************************************
/*
 JUMPING JOSAPHATS---- THERE AIN'T NO COMMANDS THAT START WITH A "J".
CHECK YOUR SPELLING.......
//   ********************************************************************
//   ********************************************************************* KEEP
KEEP

SYNTAX:            KEEP COMMAND
```

KEEP IS USED TO SAVE CERTAIN COMMANDS IN CORE IF YOU USE THEM OFTEN. NORMALLY, THE COMMAND IS BROUGHT IN FROM THE DISK AND EXECUTED, AND THEN DELETED. IF SUCH A COMMAND IS IN A LOOP,

ONE OF THE FOLLOWING COMMANDS, KEEP IT. AFTER
YOU'RE DONE WITH IT, USE DELETE/C COMMAND TO GET RID OF IT IF
YOU'RE SHORT ON CORE.

THE COMMANDS WHICH ARE DISK RESIDENT CURRENTLY ARE:

ARRAY	BACKUP	BUMP	COMPIL	COPY	CHANGE	CLIP	CONVRT
DIRDSK	DIRCOR	DIRTAP	FIX	HELP	LOGIN	LOOK	MDIRC
PUTDSK	PUTTEX	PUSHVA	POPVAR	PRINT	PERSP	RENAME	RESET
RENAME	RESCLV	SEARCH	SHADE	SMOOTH	SOFTRO	SOFTEX	TEXT
TREE	TYPE	WINDOW	ZAPTEX				

COMMANDS THAT KEEP THEMSELVES AUTOMATICALLY ARE
EDIT PATHMO TRACE

```
EXAMPLES:       KEEP GETLIB
                KEEP TREE
                KEEP PUTDSK
```

BY THE WAY, YOU HAVE TO USE THE KEEP COMMAND BEFORE YOU USE THE
COMMAND ITSELF IF YOU WANT IT TO STAY IN CORE.

```
//  **********************************************************************
/*
```

THERE IS NO HELP ON THAT ONE, IT MUST BE A DOOZY. CHECK YOUR SPELLING
BY TRYING THAT ONE AGAIN.....

```
//  **********************************************************************
//  ********************************************************************** LENGTH
```

LENGTH

SYNTAX: LENGTH CHAR STRING,VAR (STRING MANIPULATION)

LENGTH PLACES THE LENGTH OF THE STRING INTO THE VARIABLE
SPECIFIED.

```
EXAMPLE:        *$B='GRAPHICS SYMBIOSIS'
                *LENGTH $B,A
                *PROMPT A
                 18
```

```
//  ********************************************************************** LINES
```

LINES

SYNTAX: LINES LNAME

LINES PUTS LNAME'S VECTORS IN REGULAR DRAWING MODE. IT
IS USED TO RECOVER FROM DASHES AND POINTS MODES.

```
EXAMPLE:        LINES DIAMON
```

```
//  ********************************************************************** LIST
```

LIST

SYNTAX: LIST

LIST CAUSES LINES OF A MACRO TO BE ECHOED DURING EXECUTION.
IT IS A USEFUL DEBUGGING TOOL. LIST IS TURNED OFF BY XLIST.

```
//  ********************************************************************** LOGIN
```

LOGIN

IF YOU DON'T KNOW HOW TO USE LOGIN BY NOW, YOU'RE A SUSPICIOUS PERSON.
IN FACT, THE POLICE ARE COMING RIGHT NOW.....

```
//  ********************************************************************** LOOK
```

LOOK

```
SYNTAX:         LOOK ANAME,VAR
        OR      LOOK/D DNAME,VAR
```

LOOK CHECKS TO SEE THE THE ANAME IS IN CORE
(OR WITH /D, ON THE DISK). IF IT IS ON IN CORE (OR ON DISK
WITH /D) IT RETURNS A VALUE OF 0 IN THE VAR INDICATED. IF IT IS
NOT THERE, IT RETURNS A VALUE OF 1.

```
EXAMPLE:        PROM "WHAT PIX"
                INP $B
                LOOK/D $B.DEC,C
                IF C=1,PROM $B," IS NOT ON DISK";SK -3
                ...
```

```
//  ********************************************************************** LPNAME
```

LPNAME

SYNTAX: LPNAME $VAR

LPNAME RETURNS THE NAME OF THE PICTURE CORRESPONDING
TO A LIGHT PEN "HIT" IN THE $VAR INDICATED. IF NO LIGHT PEN
HIT IS SEEN, THE $VAR REMAINS UNCHANGED.

```
EXAMPLE:
                $C=''
```

```
                IF $C='', SKIP -1     (WAIT FOR HIT)
                DELETE $C              (DELETE THE PICTURE, FOR EXAMPLE)
//    ***************************************************************
/*
 I LOOKED FOR A HELP LISTING FOR THAT COMMAND, BUT I COULDN'T FIND
ANY   MUST NOT BE THERE RIGHT NOW, OR YOU DIDN'T SPELL IT RIGHT.
//    ***************************************************************

//    ********************************************************** MDIRC
MDIRC

SYNTAX:           MDIRC

        MDIRC IS LIKE DIRCOR EXCEPT IT DOESN'T PRINT OUT ALL THE
INFORMATION THAT DIRCOR SUPPLIES (I.E. NOTES OR TOTALS).  MDIRC
TAKES UP LESS SPACE THAN DIRC SO IT IS USEFUL IF YOU GET
AN ERRCR #13 (OUT OF CORE) WHEN TRYING TO USE DIRC.

//    ********************************************************** MODIFY
MODIFY

SYNTAX:           MODIFY ANAME+EXPR              ESOTERIC/////
        OR        MODIFY ANAME+EXPR,EXPR1,VAR,EXPR2
        OR        MOD EXPR

        MODIFY ALLOWS THE USER TO LOOK AT CORE LOCATIONS FOR
DEBUGGING PURPOSES. THE OUTPUT RADIX IS SET BY SWITCHES AND
THE SYSTEM THEN WAITS FOR INPUT TO CHANGE THE CORE LOCAATION.
IF A <CR> IS SEEN, NO MODIFICATIONS ARE DONE AND MODIFY IS FINISHED.
IF A <LF> IS SEEN, THE NEXT LOCATION WILL BE ACCESSED.
IF A "|" IS SEEN, THE PREVIOUS LOCATION WILL BE ACCESSED.
IF A "@" IS SEEN,THE CONTENTS OF THE ADDR IS USED AS AN ADDR (INDIRECT).

IF A NUMBER IS SEEN, IT WILL REPLACE THE PRESENT LOCATION'S CONTENTS
WITH THAT NUMBER AND DISPLAY THE NEXT LOCATION.
IF EXPR1,VAR,EXPR2 ARE USED: EXPR1 IS ADDED TO ADDR(DECIMAL)
                             VAR GETS OLD CONTENTS OF LOCATION
                             EXPR2 REPLACES LOCATION IN CORE(DECIMAL)
IF EXPR IS LEFT OUT ENTIRELY, MOD USES THE LAST ADDRESS USED BY
MOD OR SETBRK.

SWITCHES:
                /A - FOR ASCII REPRESENTATION

                /B - FOR BYTE  REPRESENTATION

                /D - FOR DECIMAL REPRESENTATION

                /O - FOR OCTAL REPRESENTATION (DEFAULT)

                /V - FOR VECTOR GENERAL ADDRESSES


EXAMPLES:        MOD VG+2000
                 MOD GRAPE+100
                 MOD 77646
//    ********************************************************** MOVE
MOVE

SYNTAX:          MOVE PNAME,DEV1,DEV2,DEV3

        THE MOVE COMMAND TRANSLATES THE PNAME ALONG THE COORDINATE
AXES THE AMOUNT INDICATED BY DEV1 FOR X, DEV2 FOR Y AND DEV3 FOR Z.

MOVE USES THE DEVICE CONVENTIONS OUTLINED IN "HELP DEV"

EXAMPLE:         MOVE GRAPE,D1
                 MOVE TOM,F
     THE FIRST EXAMPLE  MOVES THE GRAPE ACCORDING TO D1,D2, AND D3.
     THE SECOND EXAMPLE MOVES TOM ACCORDING TO THE VARIABLES F,G,AND H.
                MOVE SAM,D5,RA,1000
        THIS MOVES SAM ACCORDING TO D5 (X-AXIS), VARIABLE RA(Y-AXIS)
AND A CONSTANT 1000 (FOR Z-AXIS).
//    ***************************************************************
/*
 HMMMMMM, SOMETHING IS AMISS HERE, EITHER THERE ISN'T ANY HELP ON
 THAT COMMAND, OR YOU CAN'T SPELL IT RIGHT.  LET'S TRY IT ONE MORE
 TIME, OK?
//    ***************************************************************
//    ***************************************************************
/*
 NO NO NO, YOU'RE GONNA GET NO WHERE FAST ASKING FOR HELP ON A COMMAND
THAT STARTS WITH AN "N".      LET'S  FACE IT, YOU BLEW THE SPELLING
//    ***************************************************************
//    ********************************************************** ONERRC
ONERROR
```

SYNTAX: ONERROR VA,COMMAND

ONERROR SETS UP A COMMAND TO BE EXECUTED IN THE EVENT AN ERROR OCCURS. THE SYSTEM STICKS THE ERROR NUMBER IN THE VARIABLE INDICATED AND EXECUTES THE COMMAND YOU SPECIFIED AFTER THE COMMA IN PLACE OF THE COMMAND WHICH CAUSED THE ERROR. (NOTE THAT IT ALSO THROWS OUT THE REST OF THE LINE IN ERROR TOO.)

SKIPS IN ONERROR COMMANDS ARE TAKEN RELATIVE TO THE LOCATION OF THE COMMAND IN ERROR, NOT TO THE LOCATION OF THE ONERROR COMMAND ITSELF. IF THIS CAUSES CONFUSION, REMEMBER TO HAVE ALL ONERROR SKIPS REFERENCE %LABELS INSTEAD. THEN IT'S OK AND EVEN EASY TO FIGURE OUT.

YOU CAN EVEN CALL A MACRO IN THE EVENT OF AN ERROR. THIS IS IMPORTANT IF THERE IS A POSSIBILITY OF MULTIPLE ERRORS OCCURRING. THIS IS WHY THE ERROR NUMBER IS PUT IN THE VARIABLE.

```
EXAMPLE:        ONERROR VA,DO FIXUP
        WHERE FIXUP IS:
                FIXUP:  <IF VA=13,DELETE $B;RETURN
                        IF A=1,GETDSK $A;RETURN
                        ...ETC.>
```

THE ERROR NUMBERS ARE GIVEN IN ERRMES.SYS 31,4 IN THE RIGHT MARGIN.

SWITCHES: /A - IGNORE ALL ERRORS

NOTE THAT /A IS A DANGEROUS WAY TO DO THINGS.

NOTE: ONERROR WITHOUT ANY OPERAND CANCELS THE SETUP. YOU SHOULD CANCEL ONERRORS WHEN YOU ARE DONE WITH THEM SO THEY DO NOT DO WEIRD THINGS WHEN AN ERROR IS ENCOUNTERED SOMETIME LATER ON. RESTART CANCELS ALL ONERRORS.

EXAMPLE:

```
        ONERROR A, GETDSK WITCH.DEC 31,1
        GETDSK WITCH
                IN THIS EXAMPLE, IF THE WITCH IS NOT FOUND
                IN THE USERS AREA, INSTEAD OF GIVING AN ERROR
                MESSAGE, THE WITCH WILL BE TAKEN FROM THE
                COMMON AREA OF THE DISK.
    ONERROR A,GETD $A;SK 0
    ROTATE $A,X,DO
```

THE ABOVE WILL GET THE PIX FROM THE DISK IF IT IS NOT IN CORE BECAUSE ROTATE SENSES AN ERROR. THE "SK 0" RE-EXECUTES THE ROTATE INSTRUCTION.

```
//  ********************************************************************** OPEN
OPENO
```

SYNTAX: OPEN PIX

OPEN IS USED TO SET THE NAME FOR FUTURE PUTPOI'S AND DELPOI'S. IF YOU DO A PUTPOI WITHOUT AN OPEN, YOU WILL GET AN ERROR MESSAGE.

EXAMPLES:

```
        OPEN SAM
        OPEN $A
```

THE NORMAL SEQUENCE IS:

```
        OPEN PIX
        X=1000
        PUTPOI X,X,0,0
        PUTP X,-X,0,0
        PUTP -X,-X,0,0
        PUTP -X,X,0,0
        PUTP X,X,0,0
        CLOSE
```

THIS WILL DRAW A BOX.

A SIMPLE MACRO TO COPY EVERY OTHER POINT IN A PICTURE WOULD BE:

```
        <OPEN FOO
        N=0
        %MORE N=N+1
        GETP OLD,N,X,Y,Z,K
        IF K=-1,CLOSE;RETURN
        N=N+1
        GETP OLD,N,X,Y,Z,K
        IF K#-1,PUTP X,Y,Z,K;SK %MORE
        CLOSE>
```

NOTE THAT K=-1 IN A PUTPOI IS REALLY TAKEN AS A 0. IT DOES NOT TERMINATE THE PICTURE AS IT WOULD IN ZAPPOI. ONLY CLOSE CAN TERMINATE AN OPEN'ED PICTURE.

```
//  **********************************************************************
/*
 AH-OH, I CAN'T SEEM TO FIND HELP ON THAT COMMAND.  IT MIGHT NOT BE
 HERE, BUT CHECK YOUR SPELLING, AND TRY THAT ONE AGAIN.
//  **********************************************************************
//  ********************************************************************** PATHMO
PATHMOV
```

SYNTAX:
OR PATHMOV PNAME,PIX,VAR,DEV,DEV

PATHMOV MOVES PNAME TANGENTIALLY ALONG PIX. THE VAR INDICATES THE POSITION ALONG THE PATH AND RANGES FROM 0 TO THE NUMBER OF POINTS IN PIX. RESETTING THE VAR ANYTIME CAUSES THE PATHMOV TO RESUME AT THAT POINT. SETTING IT TO ZERO STARTS IT FROM THE BEGINNING.
THE FIRST DEV IS THE SPEED. BELOW 0, IT CAUSES AN INTERPOLATION BETWEEN POINTS. FULL NEGATIVE STOPS IT. IF THE DEV IS POSITIVE, THE SPEED IS TAKEN AS THAT MANY POINTS PER 1/60 SECOND. THE SECOND (OPTIONAL) DEV IS USED TO GIVE FINENESS WHICH HELPS SMMOOTH THE PATH SOMEWHAT. PATHMOV DOES NOT USE THE NEW DEV CONVENTIONS IN HELP DEV. WHEN THE END OF THE PATH IS REACHED, THE VAR IS SET TO -1.

EXAMPLE: PATH BIRD,PAT1,A,D0

NOTE: THE VAR (A, IN THIS CASE) IS SET TO -1 WHEN THE END OF THE PATH IS REACHED SO THE FOLLOWING SETUP WOULD CAUSE THE PATH TO BE INDEFINITELY REPEATED:

```
PATH BIRD,PAT1,A,D0
DOL <IF A LT 0,A=0>
```

SEE HELP DOLOOP IF THIS IS CONFUSNG. BASICALLY, IT JUST SETS A TO ZERO WHENEVER IT GOES NEGATIVE.

// ********** PENOFF

PENOFF

SYNTAX: PENOFF

PENOFF TURNS THE LIGHT PEN OFF.

// ********** PERSP

PERSP

SYNTAX: PERSP PIX1,PIX2,AEXPR,SCALE
WHERE SCALE IS AN EXPR

PERSP CREATES A NEW PICTURE BY MAKING VARIOUS HARDWARE MODIFICATIONS TO A PICTURE (SPECIFIED BY PIX1), AND PUTTING THE MODIFIED PICTURE INTO THE NAME GIVEN BY PIX2.
PERSP THEN DOES A VIEWING PERSPECTIVE TRANSFORMATION ON THE NEW PICTURE, BY ASSUMING THAT THE VIEWER IS AT A SCREEN COORDINATE SPECFIED BY AEXPR ON THE Z-AXIS. 500 IS THE DEFAULT VALUE USED FOR THE SCALE ARGUMENT. A SCALE FACTOR OF 1000 WILL DOUBLE THE SIZE OF THE OUTPUT PICTURE, 250 WILL REDUCE THE SIZE OF THE OUTPUT PICTURE BY 1/2, ETC.
THE OLD PICTURE IS NOT AFFECTED IN ANYWAY BY DOING A PERSPECTIVE.

SWITCHES: /C - OPERATES THE SAME AS ABOVE EXCEPT HARDWARE MODIFICATIONS ARE IGNORED AND AEXPR CAN BE SUBSTITUTED WITH XEXPR, YEXPR, ZEXPR THAT WILL SPECIFY THE POINT OF PERSPECTIVE ORGIN (I.E. THE /C SWITCH ASSUMES THE VIEWER IS LOOKING TOWARD THE SCREEN ORIGIN FROM THE COORDINATES GIVEN BY XEXPR, YEXPR AND ZEXPR, ONE OF WHICH MUST BE NON-ZERO)

EXAMPLES:

```
PERSP TV,FOO,500,250
```

THIS WILL DO A PERSPECTIVE TRANS. ON THE PICTURE TV WITH THE VIEWER AT Z-COORDINATE 500. THE RESULTING PICTURE IN "FOO" WILL BE 1/2 THE SIZE OF THE ORIGINAL.

```
PERSP/C STEEPLE,DOW,0,2000,1000,1000
```

THIS PERSPECTIVE WILL BE DONE BY PLACING THE VIEWER AT COORDINATES 0,2000,1000 AND THE RESULTING PICTURE WILL BE TWICE NORMAL SIZE.

/C PERSPS ARE GOOD FOR FOLLOWING A PATH TRROUGH AN OBJECT. USE THE PATH COORDINATES TIMES 16 AS ARGUMENTS TO PERSP/C.

// ********** POINTS

POINTS

SYNTAX: POINTS PIX

POINTS CHANGES PIX'S VECTORS TO DISPLAY ONLY THE ENDPOINTS.

EXAMPLE: PCINTS MOREY

// ********** POPVAR

POPVAR

SYNTAX: POPVAR ARGUMENTS ESOTERIC/////

POP IS THE OPPOSITE OF PUSH. IT IS USED TO RESTORE PUSHED VARIABLES. THE ARGUMENTS ARE PAIRED AND MUST BE IN THE ORDER A-Z, VA-VZ, WA-WZ, & FA-FZ. THE ARGUMENTS SPECIFIED ARE

```
EXAMPLES:
                POP
                POP C,FZ                        RESTORES ALL BUT A & B
                POP A,M,O,VB,VE,FK,FM,FZ        RESTORES ALL BUT N, VC
                                                VD, & FL
                POP VA,VA                       POPS JUST VA

THE FOLLOWING TWO EXAMPLES WILL CAUSE AN ERROR BECAUSE
THE VARIABLES ARE NOT BEING POPPED OFF IN THE PROPER ORDER.

                POP VA,VZ,A,B
                POP A,K,C,F

//   ********************************************************************** PRINT
PRINT

SYNTAX:         PRINT DNAME.EXT XX,XX

        PRINT IS THE SAME AS "TYPE" BUT IT CREATES HARDCOPY ON THE
LINE PRINTER INSTEAD OF DISPLAY ON THE VT05.  ANYTHING AFTER
THE NAME (PLUS SPACE) IS TREATED AS A COMMENT AND IS PRINTED AT
THE TOP OF THE PAGE.  DEFAULT PRINTING IS SINGLE SPACING.

SWITCHES:       /D      - THE HARDCOPY IS PRINTED WITH DOUBLE
                          SPACING (WASTES PAPER).
                /N      - SUPRESS THE DATE AND TIME

EXAMPLE:        PRINT DRAW.MAC
                PRINT PATH.DEC 50,53
//   ********************************************************************** PROMPT
PROMPT

SYNTAX:         PROMPT THING,THING,THING.....

        PROMPT IS THE COMMAND USED TO OUTPUT VALUES OF VARIABLES OR
DEV'S TO THE VT05. IT CAN ALSO BE USED TO
WRITE MESSAGES TO A MACRO USER AND THUS PROVIDES A MEANS FOR
ASKING QUESTIONS BEFORE INPUT COMMANDS, TYPING INFORMATION OUT, ETC.
THING IS EITHER A VARIABLE (STRING OR ARITHMETIC) OR A BUNCH OF LETTERS
IN DOUBLE QUOTATION MARKS.  IF A "|" IS THE LAST CHARACTER ON THE
LINE, THE CARRIAGE RETURN WILL BE SUPPRESSED AND THE INPUT CAN
THEN APPEAR ON THE SAME LINE AS THE ONE TYPED OUT. IF ANY ARGUMENTS ARE
PASSED TO THE MACRO ALL PROMPTS WILL NOT PRINT TO THE TERMINAL UNTIL
ALL OF THE ARGUMENTS ARE GONE OR INPUT/T IS USED.

        SWITCHES:       /F - FORCES PROMPT EVEN IF SUPRESSED BY
                             ARGUMENTS IN MACRO CALL.
                        /L - PROMPTS ON THE LINE PRINTER.

                        /P - IS USED TO ACTUALLY PRINT THE DATA
                             PREVIOUSLY SENT TO THE LINE PRINTER.
                             NOTE: THIS HAS NO ARGUMENTS.

                        /O - PRINT OCTAL VALUE OF VARIABLE

EXAMPLES:       PROMPT D0
                PROMPT "THE ANSWER IS ",D0
                PROMPT "A+B=",A+B,"AND C+D= ",C+D
                PROMPT  "PUSH FS13 TO EXIT"
                PROMPT "X COORDINATE"|
                PROMPT FA,FB,C

                PROM "WHAT PIX DO YOU WANT TO SEE"|
                INP $A
                GETDSK $A
                ...
//   ********************************************************************** PUSHVA
PUSHVAR

SYNTAX:         PUSHVAR                         ESOTERIC////

        PUSH PUSHES ALL THE VARIABLES IN THE ORDER A-Z, VA-VZ, WA-WZ,
& FA-FZ IN A PUSHDOWN STACK MANNER (20 DEEP).
        EACH UTILITY MACRO SHOULD PUSH UPON ENTRY & POP UPON EXIT
TO AVOID CONTAMINATING THE USER'S VARIABLES.  IT IS THE MACRO WRITERS
RESPONSIBILITY TO PAIR THE PUSHES & POP'S CORRECTLY.

SWITCHES:       /C -  CLEARS THE VARIABLES AFTER PUSHING THEM
                      (I.E. SETS THEM TO 0).
                      NOTE: UNLESS THIS SWITCH IS USED, THE VARIABLES
                      ARE NOT CHANGED BY PUSH SO ARGUMENTS MAY STILL
                      BE PASSED IN VARIABLES.

NOTE: PUSH AND POP WORK POORLY WITH DOLOOPING.
//   ********************************************************************** PUTDSK
PUTDSK
```

SYNTAX: PUTDSK DNAME,EXT

PUTDSK PUTS DNAME OUT TO YOUR AREA ON THE DISK IN THE FORMAT INDICATED BY THE EXTENSION.

EXTENSIONS:

.DEC	PICTURES(DEFAULT)
.MAC	MACROS
.CPL	COMPILE FILES
.ARA	ARRAY FILES

WHEN THE EXTENSION IS NOT ONE OF THE ABOVE, THE FILE WILL BE PUT ON DISK IN .MAC FORMAT, EVEN THOUGH IT MAY HAVE ANOTHER THREE LETTER EXTENSION. THERE IS ALSO AN OPTIONAL ARGUMENT WHICH MUST BE A $VAR. IF THIS IS SPECIFIED, THE $VAR'S CONTENTS WILL BE PUT ON THE DISK WITH THE DNAME SPECIFIED. IF THE EXT IS NOT PRESENT PUTDSK WILL TRY TO FIGURE OUT WHAT IT IS AND USE THE PROPER EXTENSION FROM ABOVE. IF NO EXT IS SPECIFIED THE PROPER ONE WILL BE ASSUMED.

YOU CANNOT PUTDSK ON ANY AREA BUT YOUR OWN.

SWITCHES:

/D - CREATES A BACKUP FILE WITH THE EXT .BAK IF THE FILE ALREADY EXISTS ON THE DISK. VERY USEFUL WITH THE QUICKY EDITOR (EDIT).

/A - THIS WILL ADD TO THE FILE ALREADY ON DISK. THIS COULD CREATE VERY LARGE DISK FILES IF USED INDISCRIMINATELY. IF THE DISK FILE DOES NOT EXIST, THE /A IS IGNORED AND THE COMMAND PROCEEDS AS A NORMAL PUTDSK.

/M - COMPLEMENT TO GETDSK/M, AS IT IGNORES EXTENTION CHECKING. IT PUTS THE FILE ON THE DISK AS IF IT WERE A MACRO.

/B - COMBINATION OF /M & /A

EXAMPLES:

```
PUTDSK DRAW.MAC
PUTDSK GLOBE
PUTDSK WITCH.DEC
PUTDSK TEMP.MAC,$C
PUTDSK HELLO.TRY
```

// ********** PUTLIB

PUTLIB

SYNTAX: PUTLIB PNAME

PUTS PNAME INTO NON-DISPLAYED DATA STRUCTURE AND REMOVES PNAME FROM HIGHER GROUPS IF ANY. PNAME MUST BE DISPLAYED. THE COMMANDS WHICH WORK ON A PUTLIB'D PNAME ARE AS FOLLOWS:

BLANK	CUTOFF	CLOSE	CLIP	COPY	DASHES	DELETE	DELPOI
FIX	GETLIB	LOOK	LINES	MOVE	PATHMOV	PERSP	POINTS
PUTDSK	PUTPOI	PUTTEX	RENAME	RESET	ROTATE	SCALE	SETCQ
SETINT	SETORG	SHADE	SMOOTH	SOFTRO	ZAPPOI	ZAPTEX	WINDOW

NOTE THAT THE UPDATES TO PNAMES ARE NOT DONE WHILE THE PNAME IS PUTLIB'D. TO HAVE THE UPDATES DONE BUT NOT USE UP COMPUTER TIME TO DISPLAY THE PIX (FOR PIX ONLY) YOU CAN USE THE BLANK COMMAND. GETLIB/W IS ALSO USEFUL FOR GETTING THE PNAME BACK SMOOTHLY.

EXAMPLE: PUTLIB OUTLINE

// ********** PUTPOI

PUTPOI

SYNTAX: PUTPOI X,Y,Z,K
WHERE X,Y,Z,K ARE EXPRS

OR PUTPOI/C PNAME,EXPR1,EXPR2,K

PUTPOI PUTS A POINT INTO THE PIX INDICATED BY THE PREVIOUS OPEN. X,Y,Z ARE ANY EXPRESSIONS DETERMINING THE X,Y, AND Z COORDINATES OF THE POINT IN DECIMAL (RANGE -2048 TO 2047). K INDICATES THE FOLLOWING:

K=0 DRAW THE VECTOR

K=1 MOVE TO NEW LOCATION BUT DO NOT DRAW (THAT IS, JUMP)

TO TERMINATE THE LIST, USE THE CLOSE COMMAND. ALSO NOTE THAT THE FIRST POINT IS ALWAYS STORED AS K=1.
SEE HELP PIX FOR MORE INFORMATION ON CREATING PICTURE LISTS.

SWITCH: /C COPIES A RANGE OF POINTS FROM ANOTHER PIX

EXAMPLE:

```
PUTPOIN D0/16,D1/16/D2/16,1
PUTPOI A,B,C,0
PUTPOI 0,0,0,-1
```

THIS WILL CREATE A PICTURE WITH A LINE DRAWN FROM THE INSTANTANEOUS POSITIONS OF D0,D1,AND D2 TO THE POSITIONS OF THE VARIABLES A,B, AND C. (THE /16 IS NECESSARY BECAUSE DIALS ARE READ LEFT JUSTIFIED.)

EXAMPLE:

```
OPEN SAMBOX
PUTP 0,0,0,0
PUTP 0,1000,0,0
```

```
                PUTP 1000,0,0,0
                PUTP 0,0,0,0
                CLOSE
THIS WILL CREATE A BOX IN THE UPPER RIGHT QUADRANT
OF THE SCREEN.  THE BOX IS NAMED SAMBOX.

EXAMPLE:        OPEN BEEP
                PUTP/C PAUL,10,20,1
THIS WILL COPY VECTORS 10 TO 20 OF PAUL INTO BEEP, STARTING OFF WITH AN
UNDRAWN VECTOR.  REGULAR PUTPOI'S CAN PRECEED AND FOLLOW PUTP/C'S.
//   ********************************************************** PUTTEX
PUTTEXT

SYNTAX:         PUTTEXT XEXPR,YEXPR,ZEXPR
                +SOME TEXT TO BE USED

        THIS COMMAND WILL DISPLAY THE SPECIFIED TEXT ON THE VG.
THIS TEXT WILL THEN BE INCORPORATED INTO THE CURRENTLY OPEN'D
PICTURE (SEE OPEN).  IT IS LIKE PUTPOIN EXCEPT
IT PUTS TEXT INSTEAD OF POINTS.  THE TEXT IS DISPLAYED AT THE
COORDINATES SPECIFIED BY XEXPR, YEXPR, AND ZEXPR.  THE EXPR
VALUES RANGE FROM -2048 TO 2047.

SWITCHES:       JUST LIKE IN TEXT

NOTE:             TEXT SHOULD BE PUT AT THE END
                OF THE PICTURE IF THE PICTURE IS TO LATER BE
                MANIPULATED WITH GETPOI OR ZAPPOI.
                DON'T PUT A   '*'  AS  THE FIRST CHARACTER,
                BECAUSE THE SYSTEM WILL THINK IT'S A
                COMMENT.


//   ************************************************************
/*
 SORRY ABOUT THIS, BUT THE INFORMATION THAT YOU WANT DOESN'T EXSIST
 AT THIS POINT IN TIME.  IS YOUR SPELLING INOPERATIVE??
//   ************************************************************
//   ************************************************************
/*
 QUIT IT WILL YA, YOU MUST BE QUAZY IT THE HEAD. YOU TRIED TO GET
HELP ON A COMMAND THAT THAT BEGINS WITH A "Q".  CHECK YOU'RE SPELLING.
//   ************************************************************
//   ********************************************************** RENAME
RENAME

SYNTAX:         RENAME ANAME1,ANAME2

        RENAME SIMPLY CHANGES THE FIRST NAME TO THE SECOND NAME.
SWITCHES:
        NONE    AS ABOVE
        /D      RENAMES DISK FILE ON YOUR AREA

EXAMPLES:       GETDSK GRAPE
                RENAME GRAPE,PRUNE
                GETDSK GRAPE
        NOW THERE ARE TWO COPIES OF GRAPE ON THE VG SCREEN.

                RENAME/D GRAPE.DEC,PRUNE.DEC
        GRAPE.DEC IS NOW CALLED PRUNE.DEC ON THE DISK.
//   ********************************************************** RESET
RESET

SYNTAX:         RESET PNAME

        RESET IS THE SAME AS FIX EXCEPT THAT THE ORIGINAL VALUES OF THE
ROTATION MATRIX, ORIGIN, SCALE, INTENSITY, ETC. ARE RESTORED.

SWITCHES:
        SAME AS FIX

EXAMPLE:        RESET/R PROP
//   ********************************************************** RESOLV
RESOLVE

SYNTAX:         RESOLVE MNAME1,MNAME2          ESOTERIC/////
        OR      RESOLVE MNAME

        RESOLVE CONVERTS %LABEL PSEUDO-LABELS IN MNAME1. IF A
MACRO HAS PSEUDO-LABELS, RESOLVE CAN BE USED TO REMOVE THEM. THE
ONLY REASON TO USE IT IS TO MAKE THE MACRO RUN SLIGHTLY FASTER.
PSEUDO-LABELS ARE USED TO ELIMINATE THE TASK OF COUNTING
LINES FOR SKIP ARGUMENTS IN LONG MACROS.  MNAME1 IS AUTO-DELETED.
IF THE SINGLE MNAME IS USED THE RESOLVED MACRO WILL HAVE THE
SAME NAME AS THE UNRESOLVED ONE.
NOTE:     WHEN YOU ORIGINALLY DEFINE THE %LABEL YOU MUST HAVE
          A SPACE OR A TAB IMMEDIATELY FOLLOWING THE LABEL
```

```
EXAMPLE:        GLOB:<IF FS1, SKIP %FLUFF
                SKIP -1
                %FLUFF  DIRCOR>
                RESOLVE GLOB,CGLOB
        CGLOB NOW LOOKS LIKE:
                CGLOB:<IF FS1,SKIP 2
                SKIP -1
                DIRCOR>
        GLOB IS AUTO-DELETED AND CGLOB CAN BE EXECUTED.
NOTE THAT THIS ALL REALLY ISN'T NECESSARY SO FORGET RESOLV.

//   ********************************************************** RESUME
RESUME

SYNTAX:         RESUME

        RESUME IS HOW YOU GET BACK TO THE MACRO IN PROGRESS WHEN
YOU ARE IN #-MODE.  YOU GET INTO #-MODE FROM TYPING |S, WHEN
AN ERROR CONDITION IS ENCOUNTERED, OR WHEN YOUR MACRO
HAS A WAIT COMMAND EXECUTED (FAIRLY RARE,ACTUALLY).
YOU CAN ALSO GET INTO #-MODE WITH A |S DURING A GETDSK.
        IN #-MODE YOU CAN EXECUTE ANY GRASS COMAMANDS OR
EVEN MACROS BUT ALL FURTHER |S USAGE IS IGNORED UNTIL RESUME IS TYPED.
//   ********************************************************** RESTAR
RESTART

SYNTAX:         RESTART
        OR      RESTART DO MNAME
        OR      RESTART ANY OF THE COMMANDS ON THE SYSTEM

        RESTART RE-INITIALIZES THE ENTIRE GRASS SYSTEM.
IF THE CALL OR DO MNAME IS INCLUDED, RESTART CLEARS THE SYSTEM AND THEN
DOES THAT MACRO AUTOMATICALLY.  (ACTUALLY, MOST COMMANDS WILL WORK
--E.G. RESTART GETC GRAPE-- BUT FEW ARE USEFUL IN THIS CONTEXT.)

SWITCHES:       /V - THE VALUES OF THE ARITHMETIC VARIABLES
                     WILL NOT BE SET TO ZERO.

EXAMPLES:       REST
                RESTART CALL DEMORY
                RESTART DO CTITLE
                RESTART DO DBLPOI.MAC XX,XX

NOTE THAT UPON RESTART, VARIABLE WZ IS RESET TO 32767.
//   ********************************************************** RETURN
RETURN

SYNTAX:         RETURN

        RETURN IS USED TO LEAVE A MACRO FROM THE MIDDLE.  IT IS
CONCEPTUALLY AND OPERATIONALLY EXACTLY EQUIVALENT TO A
SKIP 9999.

EXAMPLE:        FIXUP:  <IF A=13,PROM "OUT OF CORE";RETURN
                IF A=1,PROM "CAN'T FIND THAT";RETURN
                IF A=2, PROM "THAT'S DUMB">

NOTE THAT A RETURN IS ASSUMED AT THE END OF EVERY MACRO. YOU DO NOT
HAVE TO PUT ANTHING SPECIAL AT THE END OF A MACRO, IT RETURNS AUTOMATICALLY.
//   ********************************************************** ROTATE
ROTATE

SYNTAX:         ROTATE PNAME,AXIS,SPEED
        OR      ROTATE PNAME,AXIS,SPEED,TILT
        OR      ROTATE PNAME,AXIS,SPEED,TILT,DEV1,DEV2,DEV3
        OR      ROTATE PNAME,7,SPEED,X1,Y1,Z1,X2,Y2,Z2
                        WHERE AXIS,SPEED,TILT,X1,Y1,Z1,X2,Y2,Z2 AND
                        DEV1,DEV2.DEV3 ARE ALL DEVS

        ROTATE TAKES THE PNAME AND ROTATES IT ACCORDING TO THE
AXIS (MUST BE INDICATED AS "X", "Y", OR "Z") AT THE
SPEED (OR WITH /D, THE ANGLE) INDICATED BY A DEV.  IF INCLUDED,
THE TILT MODIFIES THE AXIS ACCORDING TO THE SETTING OF A DEV.
IF INCLUDED, DEV1, DEV2, AND DEV3 ARE THE ORIGIN OF THE AXIS
OF ROTATION.

THE /D SWITCH MAKES THE SPEED INTO AN ANGLE.
TO GET AN ANGLE OF 180 DEGREES, SET THE SPEED DEV TO 32767 (=WZ)
                   90 DEGREES, SET THE SPEED DEV TO 16383 (=WZ/2)
                   45 DEGREES, SET THE SPEED DEV TO 8191  (=WZ/4)
             30 DEGREES, SET THE SPEED DEV TO WZ/6 ETC.

ROTATE USES THE DEVICE CONVENTIONS OUTLINED IN "HELP DEV"
SWITCHES:
        /D      DEV INDICATES ANGLE OF ROTATION INSTEAD OF SPEED
        /X      COMPLEX ROTATION ADDED TO REGULAR ROTATION (X)
        /Y      SAME AS /X BUT FOR Y AXIS
```

/Z SAME AS /X BUT FOR Z AXIS

NOTE: ON /X,/Y,/Z ROTATIONS,THE AXIS IS NOT SPECIFIED AGAIN.
ALSO NOTE: A REGULAR ROTATION MUST BE DONE BEFORE A /X,/Y,/Z
MAY BE USED.
ALSO ALSO NOTE: SINCE ROTATIONS ARE NOT COMMUTATIVE, IT IS IMPORTANT
TO KNOW THAT THE ORDER YOU SPECIFY THE /X,/Y OR /Z ROTATIONS WILL CHANGE
THE OVERALL ROTATION.

IF THE CHARACTER '7' IS SPECIFIED IN PLACE OF THE AXIS OF
ROTATION, SEVEN DEV'S ARE EXPECTED, ONE FOR SPEED AND SIX TO
INDICATE THE ENDPOINTS (X1,Y1,Z1 AND X2,Y2,Z2 IN THAT ORDER)
OF THE AXIS OF ROTATION TO BE USED.

ALL SWITCHES WORK IN THE SEVEN-DIAL ROTATE (/D ETC.) TOO.

SETORG WORKS WEIRDLY WITH 7-DIAL BUT DOES A PRE-ROTATION
MOVE WITH 2 OR 5 DIAL ROTATES. SETORG DOES NOTHING
WITH SINGLE DIAL ROTATES EXCEPT SET THE ZOOM POINT FOR SCALES
WITH SWITCHES. (SEE HELP SCALE, HELP SETORG)
EXAMPLES:
```
ROTATE PROP,X,D9
ROT/D CUBE,Z,D8
ROTATE GLOBE,X,D0,D1
ROTATE/Z GLOBE,WA
ROTATE BLADE,Z,D0,D1,D2
ROT/X BLADE,D5
ROT/Y BLADE,D6
        (THESE THREE INSTRUCTIONS CAUSE A
         ROTATION WITH LOTS OF CONTROL)
ROTATE SAM,X,D0,D1,A
ROTATE/D FRED,K,L,F
ROT LARRY,7,D0,A,B,C,D,E,F   WHERE ENDPOINTS OF THE
                                AXIS OF ROTATION ARE
                                IN VAR'S A-F
```
// **********
/*
 THAT INFORMATION ISN'T IN THE FILES AS OF YET, BUT CHECK YOUR SPELLING
 JUST TO MAKE SURE.
// **********
// ********** SCALE
SCALE

SYNTAX: SCALE PNAME,DEV

SCALE CHANGES THE SIZE OF PNAME BY A FACTOR INDICATED
BY THE DEV. THIS SCALE ONLY SCALES DOWN FROM THE ORIGINAL SIZE.
TO SCALE UP, USE THE MACRO "CALL BIGGER".

SWITCHES:

NONE	USES HARDWARE SCALE WHICH SCALES TO PICTURE'S ORIGIN AS MODIFIED BY MOVE, ROTATE, ETC..
/X	SCALES ALONG X AXIS (SINGLE DIMENSIONAL SCALING).
/Y	SCALES ALONG Y AXIS.
/Z	SCALES ALONG Z AXIS.
/A	SINGLE DIMENSIONAL SCALING ON DEV,DEV+1 AND DEV+2.
/F	SAME AS /A BUT ALL ON SAME DEV

SCALING WITH SWITCHES ALSO ALLOWS THE ORIGIN OF SCALE TO BE
RESET BY USING SETORG COMMAND.

EXAMPLES:
```
SCALE TOM,D0
SCALE/X NOSE,D9
SCALE/A SPIRAL,J
```
THE LAST LINE TAKES J FOR X SCALE, K FOR Y AND L FOR
Z SCALE.
NOTE: SCALE USES THE NEW DEVICE CONVENTIONS (SEE HELP DEV).

// ********** SEARCH
SEARCH

(STRING MANIPULATION)

THE SEARCH COMMAND IS A STRING MANIPULATOR COMMAND WITH THE
FOLLOWING SYNTAX:

SEARCH/R $A,B,$C,$D	- IN THE TARGET STRING $A, START SEARCH AT LOCATION B RELATIVE TO 1ST STRING POSITION, SEARCH FOR FIRST OCCURRENCE OF $C AND REPLACE WITH $D. (NULL ARG FOR B STARTS SEARCH AT POSITION 1.)
SEARCH/R A,B,XC,$D	- LIKE ABOVE EXCEPT SEARCH FOR THE XTH OCCURRENCE OF $C WHERE X CAN BE ANY LEGIT GRASS ALGEBRAIC EXPRESSION.
SEARCH/R $A,B, $C,$D	- LIKE ABOVE EXCEPT SEARCH FOR ALL OCCURRENCES OF $C AND REPLACE WITH $D
SEARCH/R $A,B,$C	- LIKE ABOVE EXCEPT FIRST OCCURRENCE OF

$C IS DELETED.

SEARCH/F A,B,XC,D - STARTING AT LOCATION B, SEARCH $A FORWARDS FOR THE XTH OCCURRENCE OF $C AND PUT FIRST CHAR'S LOCATION IN D. (DEFAULT IS THE FIRST OCCURRENCE IF X IS OMITTED).

SEARCH/L $A,B,$C,D - STARTING AT B, SEARCH $A BACKWARDS FOR THE XTH OCCURRENCE OF $C AND PUT THE FIRST MATCHING CHAR'S LOCATION IN D.

SEARCH/T $A,B,C,$D - IN $A AT LOCATION B FOR C CHAR'S, PUT SUBSTR IN $D (PL/1 SUBSTRINGING).

SEARCH/S $A,B,C,$D - IN $A AT B FOR C CHAR'S REPLACE WITH $D.

// ********** SETBRK
SETBRK

SYNTAX: SETBRK EXPR ESOTERIC/////
OR SETBRK ANAME+EXPR (INCLUDING ALL NAMES IN NAMES TABLE)

SETBRK SETS US A BREAK-POINT AT THE LOCATION INDICATED. THE REGISTERS ARE PRINTED OUT AND THEN ANY COMMAND INCLUDING MODIFY MAY BE USED TO HELP DEBUG SOME CODE. TO DEBUG SWAP MODULES, "KEEP" THEM AND SET THE BREAKPOINT AFTER THEY ARE IN CORE. CORE RESIDENT MODULES MAY BE DEBUGGED BY SETTING BREAKPOINTS ART ABSOLUTE LOCATIONS OR OFFSETS TO THE VARIOUS MODULES (E.G. VG+2276, SUB+5000). REMOVE THE BREAKPOINT BY TYPING SETBRK 0 OR SETTING A DIFFERENT BREAKPOINT LOCATION. YOU CAN RESUME THE CODE BY TYPING "RES", ALTHOUGH THE BREAKPOINT WILL GO AWAY.

EXAMPLES: SETBRK DIRCOR+2200
SETBRK HELP+4

// ********** SETCQ
SETCQ

SYNTAX: SETCQ PNAME,DEV

SETCQ SETS THE CUTOFF PLANE (Z-AXIS CUEING) ACCORDING TO THE VALUE SPECIFIED BY THE DEV. IF THE DEV IS SET TO SLIGHTLY NEGATIVE, THEN VARYING THE INTENSITY SET BY SETINT WILL MOVE THE CUTOFF PLANE WITH RESPECT TO THE Z-AXIS. IF DEV IS SET TO FULL POSITIVE, THE DEPTH CUEING FEATURE WILL BE MOST PROMINENT WHEN THE INTENSITY DEV IS VARIED.

EXAMPLE: SETCUT GLOBE,D8

NOTE: SETCQ USES THE NEW DEV CONVENTIONS (SEE HELP DEV)
// ********** SETDEL
SETDELA

SYNTAX: SETDELA EXPR

THE DELAY, INDICATED BY A EXPR, IS USED TO SLOW DOWN GETDSK FOR .DEC MODE. SETDELA REMAINS SET FOR THE REST OF YOUR SESSION OR UNTIL RESET BY, FOR EXAMPLE:
SETDELA 0
ANYTHING LARGER THAN 10 IN SETDELA IS PRETTY SLOW.

// ********** SETINT
SETINT

SYNTAX: SETINT PNAME,DEV

SETINT SETS THE INTENSITY OF THE PNAME AS INDICATED BY THE DEV USED.

EXAMPLE: SETINT GLOBE,D0
NOTE: SETINT USES THE NEW DEV CONVENTIONS (SEE HELP DEV)
// ********** SETORG
SETORG

SYNTAX: SETORG PNAME,DEV1,DEV2,DEV3

SETORG SETS THE ZOOM POINT OF A PICTURE FOR SINGLE-DIMENSIONAL SCALING. IT DOES NOT AFFECT REGULAR SCALING. SETORG ALSO SETS THE PRE-ROTATION TRANSLATION FOR 2 AND 5-DIAL ROTATES.
EXAMPLE: SETORG SAM,JS

NOTE: SETORG USES THE NEW DEV CONVENTIONS (SEE HELP DEV)
// ********** SHADE
SHADE

SYNTAX: SHADE PIX1,PIX2,SPACING,ANGLE
WHERE SPACING AND ANGLE ARE EXPRS

SHADE FILLS IN SURFACES OF THE PICTURE SPECFIED BY PIX1. A PICTURE TO BE SHADED MUST BE COMPOSED OF A SERIES OF COPLANAR SURFACES (I.E. SITUATED IN THE SAME PLANE). EACH SEPARATE PLANE OF THE PICTURE MUST BE A CLOSED SURFACE FOR PROPER SHADING (THAT IS, THE FIRST POINT OF A PART IS ALSO ITS LAST POINT BEFORE JUMPING TO THE NEXT PLANE OF THE PICTURE).

PIX2 IS THE NAME GIVEN TO THE GENERATED SHADE VECTORS.
SPACING IS THE DISTANCE BETWEEN THE SHADE VECTORS.
IF SPACING IS NOT GIVEN THE DEFAULT VALUE IS 10 UNITS.
ANGLE IS THE ANGLE, IN DEGREES, AT WHICH THE SHADE VECTORS ARE DRAWN. IF NOT GIVEN AN ANGLE IS CALCULATED THAT USES THE LEAST AMOUNT OF CORE SPACE.

BE CAREFUL WHEN USING SHADE. BE SURE PIX1 IS DRAWN IN PROPER FORMAT AND WATCH SPACING OF SHADE VECTORS BECAUSE CORE GETS USED UP FAST WITH CLOSE SPACING.

EXAMPLES:

```
SHADE PRYM,P00,20,90
SHADE BOX,SAM
```

// ********** SKIP

SKIP

SYNTAX: SKIP EXPR
OR SKIP %LABEL

SKIP IS USED TO TRANSFER CONTROL WITHIN A MACRO FOR LOOPING. EXPRESSION MUST EVALUATE TO A NUMBER WHICH, IF NEGATIVE, SKIPS BACKWARDS THAT NUMBER OF LINES. IF POSITIVE, CONTROL SKIPS FORWARD THAT NUMBER OF LINES. NUMBERS LARGER THAN POSSIBLE WITHIN THE MACRO WILL RESULT IN TRANSFER TO THE FIRST STATEMENT OF THE PROGRAM IF NEGATIVE, OR PAST THE LAST (AND THUS EFFECTIVELY A RETURN) IF LARGE POSITIVE.

EXAMPLE:

```
<X=X+1
PROMPT X
IF FS1=0, SKIP -2>
```

THIS EXAMPLE WILL INCREMENT X AND PRINT IT UNTIL FS1 IS PUSHED.

```
<...
IF FS2=0, SKIP 0
...>
```

THIS EXAMPLE WILL WAIT UNTIL FS2 IS PUSHED BEFORE CONTINUING.

```
<...
%BACK
%BACK N=N+1
IF N LT 200,SK %BACK
...>
```

THE ABOVE SHOWS THE USE OF %LABELS. YOU SHOULD USE %LABELS NEARLY ALL THE TIME TO HELP PREVENT THE ONSET OF INSANITY.

// ********** SMOOTH

SMOOTH

SYNTAX: SMOOTH PIX,EXPR

SMOOTH USES A MODIFIED QUADRATIC SMOOTHING TECHNIQUE TO SMOOTH THE 3-D PIX THE NUMBER OF TIMES INDICATED BY THE EXPR.

EXAMPLES:

```
SMOOTH PATH,7
SMOOTH GLOBE,11/8-A
```

// ********** SOFTEX

SOFTEXT

SYNTAX:

```
SOFTEXT CNAME,PIX,VAR1,VAR2
+ONE LINE OF ANY SOFTEXT EXCLUDING "<" AND ">"
```

SOFTEXT AUTOMATICALLY CALLS THE CHARACTER SET CNAME INTO CORE IF NECESSARY AND THEN CREATES AN PIX CONSISTING OF ACTUAL 3D COORDINATES. VAR1 IS USED FOR SPACING BETWEEN CHARACTERS AND VAR2 IS USED FOR SIZE OF CHARACTERS. THE CHARACTERS ARE REDRAWN EVERY 1/10 OF A SECOND USING NEW VALUE OF VAR1 & VAR2. TO STOP REDRAW OF PIX PUSH FS15.

ROM2	DOUBLE LINE ROMAN FONT (UPPER AND LOWER, COMPLETE PUNCTUATION)
SCR	SINGLE LINE SCRIPT FONT (UPPER AND LOWER)
SCR2	DOUBLE LINE SCRIPT FONT (UPPER AND LOWER)
MAP	SELECTED MAP SYMBOLS
ITALL	LOWER CASE ITALIAN GOTHIC FONT
ITALU	UPPER CASE ITALIAN GOTHIC FONT
ENGL	LOWER CASE ENGLISH GOTHIC FONT
ENGU	UPPER CASE ENGLISH GOTHIC FONT
GRKL	LOWER CASE GREEK FONT

```
          RUSU     UPPER CASE CYRILLIC FONT
          ROMSU    UPPER CASE ROMAN TYPE
          ROMSL    LOWER CASE ROMAN TYPE
          ROM3U    UPPER CASE ROMAN TYPE (TRIPLE LINE)
          ROM3L    LOWER CASE ROMAN TYPE (TRIPLE LINE)
          ROM4U    UPPER CASE ROMAN TYPE (QUAD LINES)
          ROM4L    LOWER CASE ROMAN TYPE (QUAD LINES)
          GERMU    UPPER CASE GERMAN TYPE
          GERML    LOWER CASE GERMAN TYPE
          PUNCT    PUNCTUTAION SET
          SPUNCT   SPECIAL PUNCTUTAION SET
TO INDICATE LOWER CASE IN CHARACTER SETS WITH BOTH UPPER AND LOWER CASE,
YOU MUST TYPE A "|" BEFORE EACH LOWER CASE CHARACTER TO BE USED.  A
DOUBLE "|" (I.E. "||") IS USED AS A SHIFT LOCK.  IF "||" IS USED,
" " MUST BE USED TO INDICATE AN UPPER CASE LETTER.
"  " WILL SHIFT LOCK TO UPPER CASE AGAIN.  TEXT IN A $VARIABLE CAN BE
DISPLAYED ALSO.

EXAMPLES:        SOFTEXT ITALL,TOMMY,D0,D1
                 THIS IS AN EXAMPLE

                 A=625
                 SOFTEXT SCR2,SAM,A,D0
                 G||RAPHICS   SYMBIOSIS   SYSTEM

                 A=300;B=1000;FSON 15
                  $A='HELLO'
                  SOFTEXT ENGU,POGO,A,B
                  +JELLO $A YELLOW

        THIS WILL DISPLAY "JELLO HELLO YELLOW".

NOTE THAT SPACES MUST BE ENTERED AS UPPER CASE CHARACTERS. PUNCTUATION
IN THOSE SETS WHERE AVAILABLE, IS INDICATED BY THE SAME PUNCTUATION
CHARACTER BEING TYPED.   MAP AND RUSU RUSL HAVE SPECIAL CORRESPONDANCES
AVAILABLE FROM THE SYSTEMS PROGRAMMERS. DON'T USE '*' AS THE FIRST
CHARACTER, SINCE THE SYSTEM WILL THINK IT'S A COMMENT.

//   ********************************************************************** SOFTRO
SOFTROT

SYNTAX:          SOFTROT PIX

        SOFTROT TAKES THE VARIOUS HARDWARE MODIFICATIONS TO THE PIX
(SPECIFICALLY ROTATION, MOVING, AND SCALING) AND SOFTWARE ALTERS
THE DISPLAY LIST TO GIVE ACTUAL VECTORS REPRESENTING THESE CHANGES.
SOFTROT THEN DOES A RESET TO HALT FURTHER HARDWARE ROTATION, MOVING,
SCALING, UNTIL THE USER RE-ROTATES, ETC. A PUTDSK
WOULD THEN PUT THE TRANSFORMED LIST ONTO THE DISK.  TOO MANY SOFTROT'S
WILL SLOWLY DESTROY THE PICTURE DUE TO ROUNDOFF ERRORS.
SOFT DOES NOT DO THE CHANGES REPRESENTED IN CUTOFF, SETINT, AND SETCO.

EXAMPLE:         ROT GLOBE,Y,D0
                 SCALE GLOBE,D1
                 SOFT GLOBE

//   **********************************************************************
/*
 NO ONE TOLD ME ABOUT THAT COMMAND YET, SO HOW CAN I TELL YOU?  BY THE
 WAY, ARE YOU SURE YOU SPELLED IT RIGHT?
//   **********************************************************************
//   ********************************************************************** TRAC
TRACE

SYNTAX:          TRACE VAR1,VAR2,...          ESOTERIC/////

        TRACE SETS UP THE MECHANISM TO FOLLOW THE VALUES OF THE SPECFIED
VARIABLES AS THEY ARE CHANGED IN OTHER GRASS COMMANDS.  THIS IS A
USEFUL DEBUGGING AID.

SWITCHES:        /X - TURNS OFF TRACING FOR THE VARIABLES SPECFIED IN
                      THE REST OF THE COMMAND.

EXAMPLES:        TRACE A,X,VE,FR
                      THIS WILL TRACE THE VALUES OF A,X,VE,FR AND NOTIFY
                      THE USER EVERY TIME ONE OF THESE VARIABLES HAS
                      BEEN CHANGED.  THE USER IS TOLD THE MACRO
                      LEVEL, THE MACRO NAME, THE LINE # IN THE MACRO
                      AND THE VARIABLE'S NEW VALUE.
                       THE COMMAND THAT CAUSED THE VALUE TO
                      CHANGE IS ALSO PRINTED ON THE TERMINAL.

                 TRACE/X FR,X
                      THIS WILL TURN OFF THE TRACING OF FR & X.

                 TRACE/X
                      THIS WILL CANCEL TRACING OF ALL VARIABLES USED
```

NOTE: DIALS & ARRAYS CANNOT BE TRACED.
A MAXIMUM OF 20 VARIABLES CAN BE TRACED AT ONE TIME.
// ********** TEXT
TEXT

SYNTAX: TEXT PIX,X,Y,Z
+ ONE LINE OF TEXT TO BE DISPLAYED
WHERE X,Y,Z ARE EXPRS

TEXT DISPLAYS VECTOR GENERAL HARDWARE CHARACTERS ON THE VG SCREEN. THESE CHARACTERS (SPECFIED BY PIX) ARE NOT STANDARD PICTURES, THEY CANNOT BE ROTATED, PUTDSK'D, SCALED, ETC. AS SOFTEXT CHARACTERS CAN.
EACH PIX SIGNIFIES ONE LINE OF DISPLAYED TEXT.
X,Y,Z ARE EXPRS GIVING THE X,Y,Z COORDINATES OF WHERE PIX IS DISPLAYED ON VG SCREEN. THE EXPR CAN RANGE FROM 2047 TO -2048. DEFAULT PLACEMENT IS AT THE LEFT MARGIN OF THE VG SCREEN.

SWITCHES:

/1 THRU /4 - DISPLAYS PIX HORZONTALLY ON VG SCREEN WITH /1 BEING SMALLEST CHARACTER SIZE AND /4 THE LARGEST. DEFAULT IF NOT GIVEN IS /4.

/5 THRU /8 - DISPLAYS PIX VERTICALLY ON VG SCREEN WITH /5 SMALLEST CHARACTER SIZE AND /8 LARGEST.

SPECIAL SYNTAX:

IF A "|" IS IN DISPLAY LIST OF PIX THE NEXT CHARACTER DISPLAYED IS IN UPPER CASE.

A "||" WILL LOCK DISPLAY IN UPPER CASE CHARACTERS THIS IS NORMAL CHARACTER DISPLAY DEFAULT.

A " " WILL DISPLAY THE NEXT CHARACTER IN LOWER CASE TYPE.

A " " WILL LOCK DISPLAY IN LOWER CASE CHARACTERS.

A WIDE RANGE OF SPECIAL CHARACTERS ARE AVAILABLE WITH THIS COMMAND. SOME OF THESE INCLUDE MATHEMATICAL SYMBOLS AND THE GREEK ALPHABET. TO DISPLAY THESE SYMBOLS, THE OCTAL CHARACTER CODE OF THE DESIRED SYMBOL IS ENCLOSED IN ANGLE BRACKETS ("<",">"). THE OCTAL CODE FOR ALL THE SPECIAL CHARACTERS CAN BE FOUND IN THE VECTOR GENERAL GRAPHICS DISPLAY REFERENCE MANUAL OR IN THE VECTORGRAPHICS 11 PROGRAM REFERENCE CARD.
TO UNDERLINE CHARACTERS, FOR EXAMPLE, USE <10>_ FOR EACH CHARACTER:
I.E. HELLO<10><10><10><10><10>_____ WILL UNDERLINE HELLO.

EXAMPLES:

```
TEXT SAM,2000,2000,1000
+HELLO
      THIS WILL DISPLAY THE WORD 'HELLO'
      HORIZONTALLY AT POSITON 2000,2000,
      1000 IN  THE LARGEST CHARACTER SIZE.

TEXT/5 FAA,-2000,-1000
+<342> C  OMPUTER |GRAPHICS
      THIS WILL DISPLAY VERTICALLY, STARTING
      AT POSITION -2000,-1000,0. THE GREEK LETTER
      BETA (FROM THE OCTAL CODE <342>) FOLLOWED
      BY THE STRING 'COMPUTER GRAPHICS' WITH
      THE FIRST LETTER IN EACH WORD CAPITALIZED
      WHILE THE REST OF THE STRING IS IN LOWER
      CASE.   THE ENTIRE LINE WILL BE IN THE
      SMALLEST CHARACTER SIZE.

TEXT GA
+$A <275> <277>
        THIS WILL DISPLAY IN UPPER CASE WHATEVER IS
        IN STRING $A FOLLOWED BY THE SYMBOLS FOR
        'NOT EQUAL'  AND 'INFINITY'.
        NOTE:  $A CAN BE A MULTI-LINE STRING.
               DO NOT USE A '*' AS THE FIRST
               CHARACTER
```

// ********** TICK
TICK

SYNTAX: TICK EXPR

TICK IS THE SYSTEM TIMER. TICK 60 WILL WAIT ONE SECOND (OR SIXTY FRAMES ON THE MOVIE CAMERA), TICK 240 WILL WAIT 4 SECONDS, ETC. TICKS IN DOLOOPED MACROS ARE LOCAL TO THE

```
                    *ZAP:<A=A+1
                    +SKIP -1>
                IF IT IS VIP'D BECAUSE OF THE INFINITE LOOP.
        IF COMPILE PRINTS OUT AN ERROR MESSAGE STATING THAT
THE MACRO IS NOT VIP-ABLE, DO NOT VIP THE MACRO

//    ************************************************************************
/*
BOO-HOO, WE DON'T HAVE IT
//    ************************************************************************
//    ************************************************************************ WAITFOR
WAITFOR

SYNTAX:          WAITFOR MNAME1                    ESOTERIC/////

        THE DOLOOPED MACRO CONTAINING THE "WAITFOR MNAME1" WILL WAIT
FOR MNAME1 TO FINISH BEFORE IT CONTINUES.  THIS COMMAND IS USEFUL FOR
SYNCHRONIZING DOLOOPED MACROS.  IT CAN BE USED TO DOLOOP MACROS
WITHIN DOLOOPED MACROS TOO:
        SAM:< ....
        DOLOOP FRED
        WAITFOR FRED
        UNLOOP FRED
        ...>
SAM HAS TO BE DOLOOPED FOR THIS TO OCCUR.  IF THE DOLOOP/WAITFOR
PAIR ON FRED ARE REPLACED BY A DO FRED, THE WHOLE MACRO WILL
BE COUNTED AS A SINGLE LINE IN THE DOLOOPING STRUCTURE, WHICH MAY
OR MAY NOT BE MORE DESIRABLE THAN FRED BEING DOLOOPED AS WELL.
SEE HELP DOLOOP TOO.

//    ************************************************************************ WAIT
WAIT

SYNTAX:          WAIT                         ESOTERIC/////

        WAIT CAUSES THE MACRO TO WAIT FOR THE USER TO TYPE ON THE VT05.
IT INHIBITS THE EXECUTION OF THE MACRO UNTIL THE USER TYPES "RESUME".
WAIT IS USEFUL FOR INTERACTION WITH THE USER DURING THE USE OF A
LONG MACRO.  "CNTL S" EFFECTIVELY DOES A WAIT AND WAIT IS A
PROGRAMMED |S.
//    ************************************************************************ WINDOW
WINDOW

SYNTAX:          WINDOW PIX1,PIX2,SCALE,X,Y,Z
                        WINDOW/6 PIX1,PIX2,SCALE,X1,Y1,Z1,X2,Y2,Z2
                        WHERE SCALE, X,Y,Z,X1,Y1,Z1,X2,Y2 AND Z2 ARE EXPRS


        WINDOW IS A GENERALIZED SCALE WITH CLIPPING. OF POINTS
AND LINES OUTSIDE THE BOUNDS.  THE BOUNDS ARE
-X TO X, -Y TO Y, -Z TO Z, OR IN THE "/6" MODE BY X1 TO X2, Y1 TO Y2
Z1 TO Z2.  IF X,Y,Z ARE OMITTED, THE MAXIMUM SCREEN BOUNDARIES ARE
ASSUMED. THE FLOATING EXPRESSIOND REPRESENTS THE SCALING FACTOR
(2.5 MEANS TWO AND ONE HALF TIMES AS BIG, .33333 MEANS 1/3 THE SIZE).
EACH POINT OF THE IMAGE IS FIRST MULTIPLIED BY THE SCALE FACTOR, THEN
CLIPPED TO FIT INSIDE THE SPECIFIED BOUNDS. THE OLD IS IN NO WAY
DISTURBED    THE AUTHOR ASSUMES NO RESPONSIBILITY FOR ABSURD SCALE
FACTORS.

EXAMPLES:        WINDOW CILL,LEDGE,3.1415,100,200,300
                 WIN/6 SYLVIA,JUMP,SIN(FQ),-5,25,Q,17,DO,PO

//    ************************************************************************
/*
THAT ONE WAS A REAL WINNER, I NEVER HEARD OF IT... CHECK YOUR SPELLING
JUST TO MAKE SURE YOU DIDN'T MAKE A MISTAKE .....
//    ************************************************************************
//    ************************************************************************ XLIS
XLIST

SYNTAX:          XLIST

        THIS WONDERFUL COMMAND TURNS OFF LIST, SURPRISED????

//    ************************************************************************
/*
 OK, YOU TELL ME WHAT WENT WRONG,TURKEY..... IF YOU DON'T KNOW ASK THE
 NURD.....
//    ************************************************************************
//    ************************************************************************
/*
 YA REAL FINE, I THOUGHT EVERYBODY WHO USES GRASS KNOWS THAT THERE
ISN'T ANY COMMAND THAT STARTS WITH A "Y". CHECK-A-YOU SPELLIN'.
//    ************************************************************************
//    ************************************************************************ ZAPPO!
ZAPPO!
```

ONE TICK UNIT EQUALS 1/60 OF A SECOND.

EXAMPLE:

```
PUTL PIX
TICK 10
GETL PIX
TICK 10
SK -4
```

THIS WILL CAUSE PIX TO FLASH ON AND OFF 3 TIMES A SECOND.

// ********** TYPE
TYPE

SYNTAX: TYPE DNAME.EXT XX,XX

TYPE TYPES THE DNAME FROM THE AREA INDICATED BY " XX,XX "
THE DNAME IS TYPED ON THE VT05 AND MAY BE ABORTED BY TYPING:
"|C". IF NO " XX,XX " IS INDICATED, THE DEFAULT IS YOUR AREA.

EXAMPLES:

```
TYPE DRAW.MAC
TYPE PATH.DEC
```

// ********** TREE
TREE

SYNTAX: TREE

TREE GIVES THE USER A DIAGRAM OF HIS DATA STRUCTURE. IT LISTS
THE PICTURES AND GROUPS IN THE DISPLAYED STRUCTURE BY NAME,
INDICATING GROUPING LEVELS AND HIERARCHIES BY TABS.

// **********

/*
THAT IS STRANGE, I NEVER OF THAT COMMAND. YOU SURE YOU DIDN'T MESS
UP THE SPELLING??

// **********
// ********** UNFILM
UNFILM

SYNTAX: UNFILM ESOTERIC/////

RESETS TO NORMAL RUN MODE AFTER FILMING WAS USED. ("U" IS
ENOUGH TO TRIGGER THIS COMMAND.)

// **********UNLOOP
UNLOOP

SYNTAX: UNLOOP MNAME1,MNAME2,..... ESOTERIC/////

UNLOOP TAKES A MACRO OUT OF THE DOLOOP STRUCTURE.
SEE HELP DOLOOP FOR DETAILS.

// **********

/*
YOU ARE JUST PLAIN OUT OF LUCK IF YOU WANTED HELP WITH THAT COMMAND
NAME, BECUASE I DON'T HAVE ANY. CHECK YOUR SPELLING.....

// **********
// ********** VIP
VIP

SYNTAX: VIP CPLNAME ESOTERIC/////

VIP (VERY IMPORTANT PROGRAM) TAKES .CPL (COMPILED
MACROS) AND EXECUTES THEM AT A HIGH PRIORITY AT 60 TIMES
A SECOND OR ONCE PER TICK (IF YOU'RE FILMING, ETC).
THE .CPL SHOULD NOT CONTAIN ANY NON-TERMINATING LOOPS UNLESS
YOU REALLY WANT THE SYSTEM TO CEASE NORMAL OPERATION QUICKLY
(INFINITE LOOPS AT HIGH PRIORITY LOCK OUT EVERYTHING ELSE).
VIP IS FOR CERTAIN ANIMATION USES AND TO ELIMINATE SOME
TYPES OF LOOPING AT '*' LEVEL. |C AND |S DO NOT WORK ON
VIP'D MACROS.
UP TO 30 .CPL MACROS MAY BE SET UP USING VIP. ONE MAY
BE SET UP SEVERAL TIMES IF 60 TIMES A SECOND IS NOT FAST ENOUGH.
DELETING THE .CPL MACRO WILL ALSO REMOVE IT FROM THE VIP
LIST.
IF VERY COMPLEX .CPL MACROS ARE SETUP THIS WAY, THE
UPDATE RATE OF ROTATIONS, ETC., WILL SLOW DOWN DUE TO THE
EXTRA TIME IT TAKES.

EXAMPLES:

IF THE MACRO FILE LOOKS LIKE:

```
*SSS:<A=D0+D1/256
+B=D2*4>
```

THIS WILL HAPPEN BY ITSELF EVERY 1/60 OF A SECOND
IF YOU:

```
*COMPILE SSS.BIF
*VIP BIF
```

SYNTAX: ZAPPOI PIX,N,X,Y,Z,K
WHERE N,X,Y,Z,K ARE EXPRS

ZAPPOI MOVES A POINT, GIVEN BY N, TO THE NEW COORDINATES SPECIFIED BY X,Y,Z. THE VALUE OF N CAN RANGE FROM 1, THE FIRST POINT IN THE PICTURE, TO THE PICTURE'S LAST POINT.
K IS A VARIABLE IN WHICH THE FOLLOWING IS INDICATED:

K=0	DRAWN VECTOR
K=1	NON-DRAWN VECTOR (JUMP)
K=-1	END OF LIST (LAST VECTOR)

SWITCHES: /N - WORKS IN CONJUNCTION WITH GETPOI/N. ZAPPS THE LAST POINT THAT WAS GETP'D.

BE CAREFUL NOT TO SET K TO -1 UNLESS YOU'RE SURE BECAUSE IT TERMINATES THE LIST. ZAPPOI IS EFFECTIVELY THE REVERSE OF GETPOIN.

EXAMPLE:

```
ZAPPOI LAMP,25,-1500,-1600,500,0
  THIS WILL ZAP THE 25TH POINT OF LAMP TO THE
  NEW COORDINATES SHOWN.
```

NOTE: ZAP CAN, IN FACT, BLANK THE LAST VECTOR (K=1). HOWEVER, GETPOINT WILL STILL READ IT AS K=-1 SINCE IT IS THE END OF THE LIST. PUTDSK WILL ALSO GET CONFUSED IF THE LAST VECTOR IS BLANKED SO DON'T PUTDSK A PICTURE WITHOUT RE-ZAPPING THE LAST VECTOR TO K=-1.

AN EXAMPLE OF HOW TO USE GETP AND ZAP TOGETHER:

```
N=0
%FOO N=N+1
GETP PIX,N,X,Y,Z,K
IF K#-1,ZAP PIX,N,X*2,Y*2,Z*2,K
SK %FOO
```

THIS WILL DOUBLE THE SIZE OF THE PIX.

```
//   ********** ZAPTEX
```

ZAPTEXT

SYNTAX: ZAPTEXT PIX,$VAR1,$VAR2

ZAPTEXT SEARCHES THE PIX FOR $VAR1, DISPLAYED AS TEXT AND REPLACES IT WITH $VAR2. IF IT IS NOT FOUND, THE COMMAND EXITS WITH AN ERROR. THIS COMMAND DOES NOT ALWAYS WORK--COMPLAIN AND WE MAY FIX IT.

```
//   **********
/*
```

NO DESCRIPTION IS AVAILABLE FOR THIS COMMAND AT THIS TIME.

```
//   **********
1,E.PIX------------------------------------------>>>     ERROR 1
THIS NAME CANNOT BE FOUND--CHECK SPELLING OR DIRCOR
2,E.CMD------------------------------------------>>>     ERROR 2
THIS COMMAND NAME CANNOT BE FOUND--CHECK SPELLING AND HELP
3,E.SWH------------------------------------------>>>     ERROR 3
THIS SWITCH IS NOT DEFINED FOR THIS COMMAND
4,E.SYN------------------------------------------>>>     ERROR 4
UNDIAGNOSIABLE SYNTAX ERROR--SEE HELP COMMAND
5,E.UNF------------------------------------------>>>     ERROR 5
MORE INFORMATION NEEDED FOR THIS COMMAND--SEE HELP COMMAND
6,F.OVR------------------------------------------>>>     ERROR 6
TOO MUCH INFORMATION GIVEN--PLEASE LOG THIS ONE AS A BUG
7,F.NID ------------------------------------------>>>     ERROR 7
INPUT DEVICE NOT AVAILABLE
10,E.NOD------------------------------------------>>>     ERROR 8
OUTPUT DEVICE IS NOT AVAILABLE
11,E.NIF------------------------------------------>>>     ERROR 9
THIS NAME DOES NOT EXIST ON THE DISK--CHECK DIRDSK
12,E.NOF------------------------------------------>>>     ERROR 10
THIS NAME IS ALREADY ON DISK OR NOT LOGGED INTO  1,1
13,E.CAR------------------------------------------>>>     ERROR 11
COMMAND GOT CONFUSED WITH FUNNY CHARACTER--CHECK TYPING
14,E.NCS------------------------------------------>>>     ERROR 12
THIS FILE IS NOT A CHARACTER SET
15,E.CCR------------------------------------------>>>     ERROR 13
SYSTEM SPACED OUT--NO STORAGE LEFT--DELETE SOMETHING
16,E.TYP------------------------------------------>>>     ERROR 14
THIS NAME EXTENSION IS NOT VALID--SEE HELP GETDSK
17,E.MAX------------------------------------------>>>     ERROR 15
TOO MANY NAMES USED--DELETE SOMETHING
20,E.DEV------------------------------------------>>>     ERROR 16
A DEVICE OR VARIABLE NAME IS EXPECTED HERE
21,E.NFX------------------------------------------>>>     ERROR 17
THE PICTURE/MACRO DUALITY HAS BEEN CONFUSED--CHECK DIRCOR
22,E.STR------------------------------------------>>>     ERROR 18
THE DISK HAS NOT BEEN INITIALIZED FOR YOUR AREA-SEE TOM
23,E.PRV------------------------------------------>>>     ERROR 19
THIS PICTURE NAME ALREADY IN USE--CHECK DIRCOR OR RENAME
```

```
ONLY DISPLAYED PICTURES MAY BE PUTLIB'D
25,E.GLB----------------------------------------->>>     ERROR 21
ONLY NON-DISPLAYED (PUTLIB'D) PICTURES MAY BE GETLIB'D
26,E.MIS----------------------------------------->>>     ERROR 22
THERE IS A COORDINATE MISSING IN THE INPUT DATA
27,E.OFL----------------------------------------->>>     ERROR 23
X,Y,Z MAX/MIN ARE 2047/-2048 RESPECTIVELY
30,E.LTM----------------------------------------->>>     ERROR 24
THERE IS A LINE TERMINATOR MISSING--CHECK INPUT CAREFULLY
31,E.MAC----------------------------------------->>>     ERROR 25
THIS MACRO HAS GONE ASTRAY--CHECK LOGIC AND SYNTAX
32,E.WOW----------------------------------------->>>     ERROR 26
THIS COMMAND DOES NOT WORK WITH PICTURES
33,E.N3D----------------------------------------->>>     ERROR 27
DATA NOT IN 3D SHADE FORMAT
34,E.BSP----------------------------------------->>>     ERROR 28
SOFTWARE CHARACTER SET JUMBLED
35,E.NPT----------------------------------------->>>     ERROR 29
LINE OVER 80 CHARACTERS LONG ASKED TO EXECUTE--SHORTEN IT
36,E.CPT----------------------------------------->>>     ERROR 30
THERE IS A MISSING ='S SIGN HERE
37,E.DMT----------------------------------------->>>     ERROR 31
THIS DEVICE DOES NOT EXIST AS SPELLED ABOVE
40,E.BAD----------------------------------------->>>     ERROR 32
THIS NUMBER HAS INVALID CHARACTERS IN IT--CHECK TYPING
41,E.UGP----------------------------------------->>>     ERROR 33
THIS COMMAND IS FOR PIX WITH VECTORS ONLY (NO COPIES, NO GROUPS).
42,E.UFN----------------------------------------->>>     ERROR 34
THIS FEATURE DOES NOT WORK IN THIS COMMAND YET.
43,E.TXT----------------------------------------->>>     ERROR 35
ILLEGAL VG CHARACTER SPECIFIED IN <XXX>.
44,E.NGS----------------------------------------->>>     ERROR 36
A STRING IN SINGLE QUOTES (') OR A $VAR IS EXPECTED HERE
45,E.NDS----------------------------------------->>>     ERROR 37
ONLY DISPLAYED PICTURES MAY BE GROUPED.
46,E.WWO----------------------------------------->>>     ERROR 38
THIS COMMAND WORKS ONLY WITH PICTURES
47,E.AXS----------------------------------------->>>     ERROR 39
ROTATE REQUIRES AN AXIS  (EXCEPT /X,/Y,/Z MODES)
50,E.TXG----------------------------------------->>>     ERROR 40
CLOSING ANGLE BRACKET (>) NOT FOUND.
51,E.OVF----------------------------------------->>>     ERROR 41
YOU CAN'T EXECUTE A NULL STRING
52,E.GRB----------------------------------------->>>     ERROR 42
BINARY FILE IS NOT A PICTURE OR IS GARBLED
53,E.PRO----------------------------------------->>>     ERROR 43
ONLY GOD CAN WRITE ON HERE. (UNLESS YOU KNOW HIS PASSWORD)
54,E.CNT----------------------------------------->>>     ERROR 44
ONLY YOUR OWN FILES CAN BE DELETED OR RENAMED
55,E.CAL----------------------------------------->>>     ERROR 45
PICTURE CONTAINS SUBROUTINE CALLS & CANNOT BE WRITTEN
56,E.UNP----------------------------------------->>>     ERROR 46
THIS PICTURE IS SOMEHOW IN NON-STANDARD FORMAT
57,E.NMC----------------------------------------->>>     ERROR 47
MACRO MUST BE IN PROGRESS FOR THIS COMMAND TO WORK
60,E.IVN----------------------------------------->>>     ERROR 48
THIS NAME HAS AN INVALID CHARACTER IN IT (A-Z & 0-9 ONLY)
61,E.FOR----------------------------------------->>>     ERROR 49
ONLY VARIABLE NAMES ARE ALLOWED LEFT OF THE "=" SIGN
62,E.ARG----------------------------------------->>>     ERROR 50
NO ARGUMENT DELIMITER SEEN WHERE EXPECTED--CHECK SPELLING
63,E.GEZ----------------------------------------->>>     ERROR 51
ONLY VARIABLES CAN BE WRITTEN TO--DEVICES CAN ONLY BE READ
64,E.FUK----------------------------------------->>>     ERROR 52
GETPOI OR ZAPPOI INDEX OUT OF BOUNDS
65,E.OPP----------------------------------------->>>     ERROR 53
ATTEMPT TO CREATE NEW 'OPENO' BEFORE CLOSING THE OLD ONE
66,E.NOO----------------------------------------->>>     ERROR 54
ATTEMPT TO GENERATE A POINT WHEN NO 'OPENO' IS IN PROGRESS
67,E.IVG----------------------------------------->>>     ERROR 55
CAN'T OUTPUT THIS PIX AS .DEC--HAS TEXT OR IS NOT CLOSED
70,E.FSE----------------------------------------->>>     ERROR 56
FUNCTION SWITCH NUMBER GREATER THAN 15
71,E.ONA----------------------------------------->>>     ERROR 57
$A TO $Z ONLY ALLOWABLE NAMES VARIABLES
72,E.SWP----------------------------------------->>>     ERROR 58
GROUPING ERROR--RESTART AND CHECK TREE NEXT TIME
73,E.XXX----------------------------------------->>>     ERROR 59
PSEUDO-LABEL ERROR DETECTED BY COMPILE COMMAND
74,E.XXX----------------------------------------->>>     ERROR 60
THE ABOVE PSEUDO-LABEL IS NOT FOUND
75,E.XXX----------------------------------------->>>     ERROR 61
*****LOG THIS ONE--CAN'T FIGURE IT OUT
76,E.XXX----------------------------------------->>>     ERROR 62
EDITOR FILE ALREADY OPEN (TYPE REST)
77,E.XXX----------------------------------------->>>     ERROR 63
NO EDITOR FILE HAS BE OPENE'D
```

MAXIMUM NUMBER (20) OF PUSHES EXCEEDED--POP SOMETHING
101,XXX----------->>> ERROR 65
ONLY EDITOR FILES MAY BE CLOSED'D
102,XXX----------->>> ERROR 66
DIRECTORY CANNOT BE DECODED ON THIS DEVICE
103,XXX----------->>> ERROR 67
FILE EXTENSION AND DATA TYPE DO NOT MATCH
104,XXX----------->>> ERROR 68
ONLY SWAP MODULES MAY BE MADE CORE-RESIDENT
105,XXX----------->>> ERROR 69
ILLEGAL VG CHARACTER SIZE 1 TO 8 ONLY ALLOWED
106,XXX----------->>> ERROR 70
POPPING WITHOUT PUSHING (ENOUGH) IS NOT MAKING SENSE
107,XXX----------->>> ERROR 71
A $VAR MUST BE SPECIFIED HERE
110,XXX----------->>> ERROR 72
FPP: ILLEGAL OPERATOR IN EXPRESSION
111,XXX----------->>> ERROR 73
FPP: INCOMPLETE EXPRESSION
112,XXX----------->>> ERROR 74
FPP: UNBALENCED PARENTHESIS
113,XXX----------->>> ERROR 75
FPP: ILLEGAL FLOATING CONSTANT
114,XXX----------->>> ERROR 76
FPP: INVALID OPERAND IN EXPRESSION
115,XXX----------->>> ERROR 77
FPP: NO ARGUMENT SPECIFIED ON FUNCTION CALL
116,XXX----------->>> ERROR 78
FPP: DIVISION BY ZERO ATTEMPTED
117,XXX----------->>> ERROR 79
FPP: FLOATING TO INTEGER CONVERSION ERROR
120,XXX----------->>> ERROR 80
FPP: ATTEMPT TO TAKE THE LN OF ZERO
121,XXX----------->>> ERROR 81
FPP: SQR OF A NEGATIVE NUMBER IS IMAGINARY
122,XXX----------->>> ERROR 82
THIS NAME DOES NOT EXIST ON TAPE
123,XXX----------->>> ERROR 83
THIS NAME CANNOT NOW BE USED ON TAPE (MAY ALREADY EXIST)
124,XXX----------->>> ERROR 84
THE TAPE HAS NOT BEEN INITIALIZED FOR YOUR AREA - SEE TOM
125,F.CPL----------->>> ERROR 85
ONLY CPL FILES CAN BE 'EXECUTED'
126,XXX----------->>> ERROR 86
COMPILE: INVALID STATEMENT SYNTAX
127,XXX----------->>> ERROR 87
COMPILE: INVALID LINE END OR LINE SYNTAX, CHECK MACRO
130,XXX----------->>> ERROR 88
COMPILE GOOFED, PLEASE LOG THIS BUG
131,XXX----------->>> ERROR 89
COMPILE WILL NOT PROCESS ALL OF YOUR MACRO - SEE TOM
132,XXX----------->>> ERROR 90
COMPILE: MISSING OPERATOR IN PUTPOI,GETPOI,OR ZAPPOI
133,XXX----------->>> ERROR 91
COMPILE: ILLEGAL OPERATOR IN FLOATING EXPRESSION
134,XXX----------->>> ERROR 92
COMPILE: INCOMPLETE FLOATING EXPRESSION
135,XXX----------->>> ERROR 93
COMPILE: UNBALENCED PARENTHESIS IN FLOATING EXPRESSION
136,XXX----------->>> ERROR 94
COMPILE: ILLEGAL FLOATING CONSTANT
137,XXX----------->>> ERROR 95
COMPILE: INVALID OPERAND IN FLOATING EXPRESSION
140,XXX----------->>> ERROR 96
MAX STRING LENGTH (8000 CHARACTERS) EXCEEDED
141,XXX----------->>> ERROR 97
ATTEMPT TO ACCESS NEGATIVE LOCATION IN STRING
142,XXX----------->>> ERROR 98
TARGET STRING IS NULL
143,XXX----------->>> ERROR 99
MATCH STRING IS NULL
144,XXX----------->>> ERROR 100
BUMP ONLY WORKS IF $A IS A-Z,VA-VZ,WA-WY,FA-FY,AA-AY,LA-LY
145,XXX----------->>> ERROR 101
TARGET IS NOT A 'LINED' STRING
156,XXX----------->>> ERROR 1110
ERROR IN POPPING---SEE HELP POPVAR
160,XXX----------->>> ERROR 112
COMPILE: NO AGRUMENT SPECIFIED ON FUNCTION CALL
161,XXX----------->>> ERROR 113
ARRAYS ARE AA-AZ ONLY
162,XXX----------->>> ERROR 114
UPPER BOUND OF THE ARRAY IS TOO LARGE OR SOMETHING
163,XXX----------->>> ERROR 115
YOU CANNOT EXECUTE STRINGS SHORTER THAN 8 CHARACTERS, SORRY.
164,XXX----------->>> ERROR 116
ARRAY SUBSCRIPTS ARE TERMINATED WITH ')' OR ','

```
166,XXX-------------------------------------------->>>     ERROR 118
COMPILE: NESTED MACROS ARE NOT ALLOWED IN CPL FILES
167,XXX-------------------------------------------->>>     ERROR 119
TOO MANY SUBSCRIPTS HAVE BEEN SPECIFIED FOR THIS ARRAY
170,XXX-------------------------------------------->>>     ERROR 120
THE SYSTEM THINKS THIS IS AN ARRAY, BUT IT'S NOT
171,XXX-------------------------------------------->>>     ERROR 121
NOT ENOUGH SUBSCRIPTS ARE SPECIFIED FOR THIS ARRAY
172,XXX-------------------------------------------->>>     ERROR 122
ARRAY SUBSCRIPT OUT OF BOUNDS
173,CCC-------------------------------------------->>>     ERROR 123
THIS ARRAY HAS ALREADY BEEN DIMENSIONED
174,XXX-------------------------------------------->>>     ERROR 124
ARRAY SPECIFIED ISN'T BEEN DIMENSIONED--USE ARRAY COMMAND
176,XXX-------------------------------------------->>>     ERROR 126
THE VALUES OF DIALS AND ARRAYS CANNOT BE TRACED.
177,XXX-------------------------------------------->>>     ERROR 127
LOCAL VARIABLES CAN ONLY BE USED WITHIN MACROS.
200,XXX-------------------------------------------->>>     ERROR 128
THE MAXIMUM NUMBER OF TRACES HAS BEEN EXCEEDED.
201,XXX-------------------------------------------->>>     ERROR 129
YOU CAN'T DOLOOP THE SAME MACRO TWICE
202,XXX-------------------------------------------->>>     ERROR 130
MAXIMUM NUMBER OF THINGS TO DOLOOP IS 10
203,XXX-------------------------------------------->>>     ERROR 131
CANNOT UNLOOP SOMETHING THAT ISN'T DOLOOPED
204,XXX-------------------------------------------->>>     ERROR 132
CANNOT WAITFOR SOMETHING THAT ISN'T DOLOOPED
205,DDD-------------------------------------------->>>     ERROR 133
THIS ERROR IS IMPOSSIBLE
210,XXX-------------------------------------------->>>     ERROR 136
PUTDSK OF ARRAYS NEEDS SECOND ARGUMENT -- SEE HELP PUTDSK
211,XXX-------------------------------------------->>>     ERROR 137
CORE ALLOCATION IS CONFUSED BY THIS DELETION--RESTART
206,XXX-------------------------------------------->>>     ERROR 134
SOMEHOW THE NAME TO BE DELETED WAS LOST--TRY RESTARTING
212,XXX-------------------------------------------->>>     ERROR 138
FIRST PICTURE IN GROUP COMMAND NOT DISPLAYED
213,XXX-------------------------------------------->>>     ERROR 139
SECOND PICTURE IN GROUP COMMAND NOT DISPLAYED
215,XXX-------------------------------------------->>>     ERROR 141
PIX TO BE GROUPED MUST BE ON SAME LEVEL (CONSULT TREE)
216,000-------------------------------------------->>>     ERROR 142
I FORGOT
217,000-------------------------------------------->>>     ERROR 143
I FORGOT
220,000-------------------------------------------->>>     ERROR 144
YOU CAN'T GET A LOCAL VARIABLE THAT DOESN'T EXIST YET
221,000-------------------------------------------->>>     ERROR 145
NOT ENOUGH SPACE FOR COPY/V OR IS NOT COPIABLE
222,000-------------------------------------------->>>     ERROR 146
YOU CANNOT COPY COPIES--COPY THE ORIGINAL
223,000-------------------------------------------->>>     ERROR 147
PIX IS NOT PUTLIB'D BUT ISN'T DISPLAYED EITHER--GIVE UP
224,000-------------------------------------------->>>     ERROR 148
CHECK THIS DISK FILE CAREFULLY--ERROR DETECTED IN WRITING
225,000-------------------------------------------->>>     ERROR 149
THE DISK IS FULL--FILE MAY BE INCOMPLETELY WRITTEN OUT
226,000-------------------------------------------->>>     ERROR 150
DELETE THINKS THIS PIX IS PUTLIB'D BUT IT'S NOT--GIVE UP
227,000-------------------------------------------->>>     ERROR 151
GETPOINT IS CONFUSED BY PIX THAT AREN'T CLOSED--CLOSE IT
230,PPP-------------------------------------------->>>     ERROR 152
GRASSB DOESN'T HAVE FLOATING POINT VARIABLES-USE GRASS/GRASS2
300,XXX-------------------------------------------->>>     ERROR 192
SOFTWARE POINTERS CRAPPED UP--RESTART (GOON)
301,XXX-------------------------------------------->>>     ERROR 193
SOFTWARE CRAPPED UP POINTERS--GIVE UP (UPGON)
302,XXX-------------------------------------------->>>     ERROR 194
SOFTWARE CRAPPED UP POINTERS--GIVE UP (UPBAD)
303,XXX-------------------------------------------->>>     ERROR 195
SOFTWARE CRAPPED UP POINTERS--GIVE UP (NEAR UPDOIT)
304,XXX-------------------------------------------->>>     ERROR 196
GROUPING SCREWED UP--GIVE UP (UPDOIT)
305,XXX-------------------------------------------->>>     ERROR 197
BAD CHAINING--GIVE UP (UPDOIT)
313,XXX-------------------------------------------->>>     ERROR 203
SYSTEM CAN'T HARDWARE DECHAIN--GIVE UP (HCHNOK)
0
PLEASE LOG THIS ERROR CAREFULLY IN THE BUG LOG--TOM
```