

# Computer Programming Then and Now?

*Alan German*  
*Ottawa PC Users' Group*

# Elliott 803B



# Teletypewriter



# 5-Hole Paper Tape





# Elliott Autocode

A SPECIFICATION OF THE  
MARK 3 AUTOCODE  
FOR THE 803  
ELECTRONIC DIGITAL COMPUTER

**ELLIOTT**

SCIENTIFIC COMPUTING DIVISION

## SUMMARY OF INSTRUCTIONS FOR AUTOCODE

In these examples

A, B, C and D	represent	Floating-point variables
I, J, K and L	represent	Integer variables
i, m and n	represent	Positive integer constants
p, q and r	represent	Any integer constants
x, y and z	represent	Floating-point constants

Any variable except the one before the = sign may be replaced by a constant.

### Arithmetic

A=B	A=-B	I=J	I=-J
A=B+C	A=-B+C	I=J+K	I=-J+K
A=B-C	A=-B-C	I=J-K	I=-J-K
A=B*C	A=-B*C	I=J*K	I=-J*K
A=B/C	A=-B/C		

### Function

A=SIN B	A=LOG B	A=FRAC B	-
A=COS B	A=EXP B	A=INT B	I=INT A
A=TAN B	A=SQRT B	A=STAND I	-
A=ARCTAN B		A=MOD B	I=MOD J

### Jump

JUMP @K  
JUMP IF A=B@K  
JUMP UNLESS A=B@K  
JUMP IF I=J@K  
JUMP UNLESS I=J@K  
(K may not have any form of suffix).  
Any permitted arithmetical instruction or function instruction may replace A=B or I=J, and > (%) or < (\$) may replace =

### Other Controls

SUBR n    EXIT    STOP    WAIT

### Vary and Cycle

VARY A=B:C:L  
CYCLE A=B:C:D  
CYCLE A=x, y, z, ...  
REPEAT A  
(B, C, D, J, K, and L may have simple suffixes only)

VARY I=J:K:L  
CYCLE I=J:K:L  
CYCLE I=p, q, r, ...  
REPEAT I

### Input

READ A    READ I    INPUT I

### Output

PRINT A, n: m PRINT A, n    PRINT A, n/    PRINT A  
PRINT I, n    PRINT I    OUTPUT I  
(In OUTPUT I, I may have a numerical suffix only).  
LINE    LINES I    SPACES I    TITLE  
CHECK A    CHECK I

### Setting and Start

SETS (Integer variables).  
SETV (Floating-point variables).  
SETF (Functions).  
SETR n (Maximum reference number).  
START m (Starting reference number)  
In SETF (i) TRIG covers SIN, COS and TAN.  
(ii) MOD and STAND need not be mentioned.  
(iii) FILM allows use of film instructions.  
(iv) CARD and PAR allow use of card reader instructions

### Film

FILM(I) SEARCHJ(K)  
FILM(I) TO J(K) or FILM(I) TO A(K)  
FILM(I) FROM J(K) or FILM(I) FROM A(K)  
JUMP IF FILM(I) SEARCHING @ L  
JUMP UNLESS FILM(I) SEARCHING @ L  
FILM(I) BLOCK NUMBER TO J(K)  
FILM(I) ALLOW WRITE  
FILM(I) PREVENT WRITE  
K is any form of suffix  
L cannot have any form of suffix

### Card

J=PAR i, m, n  
i, m, n may be replaced by integer variables having numerical suffixes only.  
A=CARD 1, A, J, K or I =CARD 1, I, J, K  
A=CARD 2, A, J, K or I =CARD 2, I, J, K  
K may be replaced by an integer constant.

# Elliott Autocode

A SPECIFICATION OF THE  
MARK 3 AUTOCODE  
FOR THE 803  
ELECTRONIC DIGITAL COMPUTER

**ELLIOTT**  
SCIENTIFIC COMPUTING DIVISION

## SUMMARY OF INSTRUCTIONS FOR AUTOCODE

In these examples

A, B, C and D	represent	Floating-point variables
I, J, K and L	represent	Integer variables
i, m and n	represent	Positive integer constants
p, q and r	represent	Any integer constants
x, y and z	represent	Floating-point constants

Any variable except the one before the = sign may be replaced by a constant.

### Arithmetic

### Input

READ A      READ I      INPUT I

### Output

PRINT A, n: m PRINT A, n      PRINT A, n/      PRINT A  
PRINT I, n      PRINT I      OUTPUT I  
(In OUTPUT I, I may have a numerical suffix only).  
LINE      LINES I      SPACES I      TITLE  
CHECK A      CHECK I

### Setting and Start

NS (Integer variables).  
FV (Floating-point variables).  
TF (Functions).  
TR n (Maximum reference number).  
URT m (Starting reference number).  
SIG covers SIN, COS and TAN.  
OD and STAND need not be mentioned.  
LM allows use of film instructions.  
JRD and PAR allow use of card reader actions

# JUMP IF A=B@K

### Jump

JUMP @K      JUMP IF I=J@K  
JUMP IF A=B@K      JUMP UNLESS I=J@K  
JUMP UNLESS A=B@K      JUMP UNLESS I=J@K  
(K may not have any form of suffix).  
Any permitted arithmetical instruction or function instruction may replace A=B or I=J, and > (%) or < (\$) may replace =

### Other Controls

SUBR n      EXIT      STOP      WAIT

### Vary and Cycle

VARY A=B: C: L      VARY I=J: K: L  
CYCLE A=B: C: D      CYCLE I=J: K: L  
CYCLE A=x, y, z, ...      CYCLE I=p, q, r, ...  
REPEAT A      REPEAT I  
(B, C, D, J, K, and L may have simple suffixes only)

FILM(I) TO J(K) or FILM(I) TO A(K)  
FILM(I) FROM J(K) or FILM(I) FROM A(K)  
JUMP IF FILM(I) SEARCHING @ L  
JUMP UNLESS FILM(I) SEARCHING @ L  
FILM(I) BLOCK NUMBER TO J(K)  
FILM(I) ALLOW WRITE  
FILM(I) PREVENT WRITE  
K is any form of suffix  
L cannot have any form of suffix

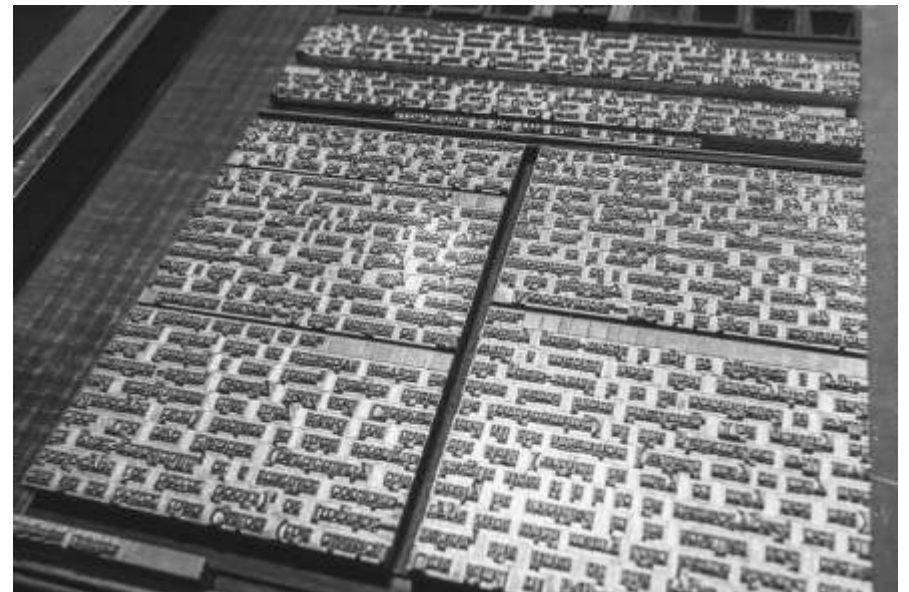
### Card

J=PAR i, m, n  
i, m, n may be replaced by integer variables having numerical suffixes only.  
A=CARD 1, A, J, K or I =CARD 1, I, J, K  
A=CARD 2, A, J, K or I =CARD 2, I, J, K  
K may be replaced by an integer constant.





# Linotype Machine





# Type and horizontal spacing

12 points  
equal 1 pica

6 picas  
(72 points)  
equal 1 inch



## 60-POINT SCALE

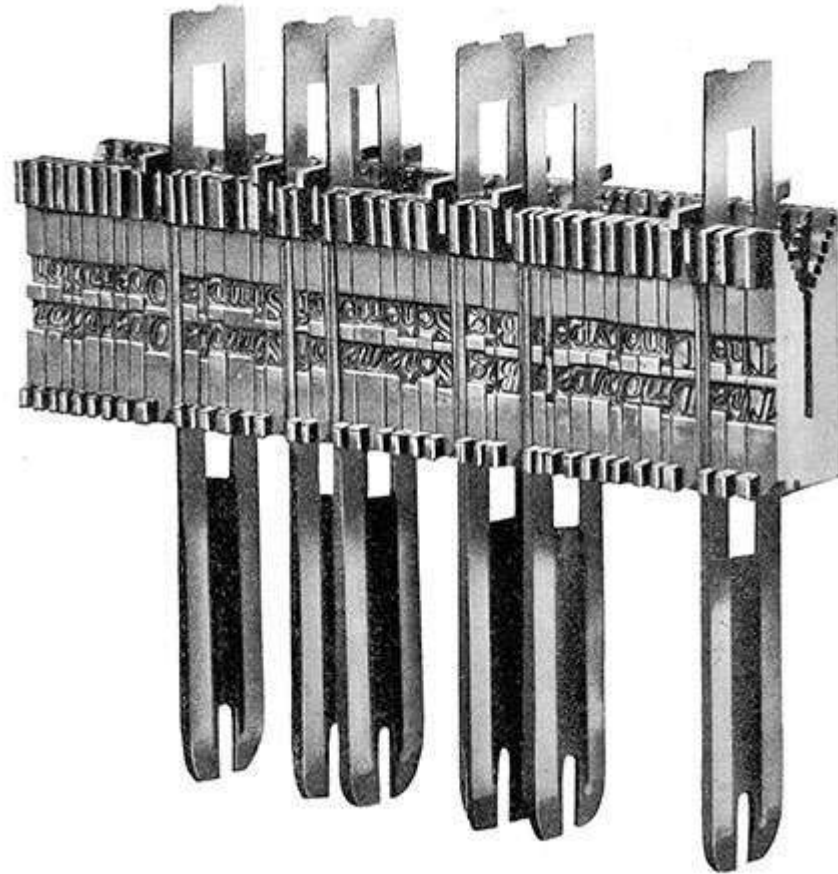
A typeface is measured from the top of the capital letter to the bottom of the lowest descender, plus a small buffer space.



In metal type, the point size is the height of the type slug.

- Letters in a proportional font have different widths (e.g. i and w)
- Lines can be ragged (left justified),  
they can be fully justified  
or right justified
- Inter-word spacing can be variable, thin, thick, or some combination

# Variable Spaces



The spaceband is basically a wedge that expands the spaces between words

# Computer Typesetting

- Set the width of the line
- Add up the widths of the characters in each word
- Add the widths of the spaces
- Determine if the variable spaces will justify the line
- Add thin/thick spaces
- Letter space (thin spaces between letters)
- If all else fails - hyphenate!



# **Elliott Autocode**

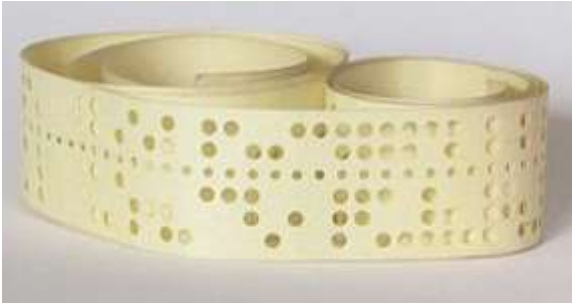
**M = 633**

**L = 0**

**L = L + C**

**JUMP IF L>M @ Z**

# But, there's a problem...



# But, there's a problem...



00000 = 0

11111 = 31



# But, there's a problem...



a...z

A...Z

. , ' @ ! ...etc.

00000 = 0

11111 = 31

# But, there's a problem...



a...z

A...Z

. , ' @ ! ...etc.

00000 = 0

11111 = 31

\* = Shift

@ = Unshift

\*A@LAN = Alan

LINE WIDTH IN UNITS = 200

\$\$\$\$\*A@T \$ THE  
MOMENT  
N\$A\$T\$I\$O\$N\$A\$L  
NEWSPAPERS ARE  
PRODUCED IN  
EITHER TWO OR  
THREE \$MAIN-  
CENTRES--I.E.  
LONDON,  
\*M@ANCHESTER,  
OR \$\*S@COTLAND  
(EITHER  
\*G@LASGOW \$OR  
\*E@DINBURGH).=

\$\$\$\$ = Tab

\* @ = Cap

Space = Var Sp

\$ = Thin space



# Left justified Linotype output

2122-2240

7053-7284

typesetting by computer

phase d

line width in units=633

Information Sheet from the School of  
Advanced Studies.

## A COMPUTER TYPESETTING PROGRAM

A Project carried out jointly by the  
Manchester College of Art and Design and  
John Dalton College of Technology,  
Manchester (Faculties in the proposed  
Manchester Polytechnic).

This information paper describes the  
co-ordinating work done by Colin Nield, a  
Fellow of the School of Advanced Studies  
Manchester College of Art and Design on the  
writing of a computer typesetting program and  
the achievement of a production capability.

The program was written by Alan German, a  
technical assistant in the Physics and  
Mathematics Department of John Dalton  
College.

Nield has now completed his Fellowship  
year and is soon to join P.I.R.A as a composition  
consultant.

It is hoped that work on the program will  
continue at John Dalton College under the

# Algol 60

```
procedure Absmax(a) Size:(n, m) Result:(y) Subscripts:(i, k);  
  value n, m; array a; integer n, m, i, k; real y;  
  comment The absolute greatest element of the matrix a, of size n by m  
    is transferred to y, and the subscripts of this element to i and k;  
begin  
  integer p, q;  
  y := 0; i := k := 1;  
  for p := 1 step 1 until n do  
    for q := 1 step 1 until m do  
      if abs(a[p, q]) > y then  
        begin y := abs(a[p, q]);  
          i := p; k := q  
        end  
      end  
    end  
  end  
end Absmax
```

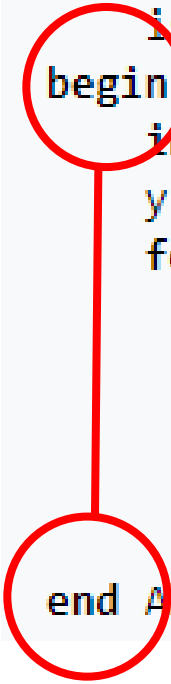
# Algol 60

```
procedure Absmax(a) Size:(n, m) Result:(y) Subscripts:(i, k);  
  value n, m; array a; integer n, m, i, k; real y;  
  comment The absolute greatest element of the matrix a, of size n by m  
    is transferred to y, and the subscripts of this element to i and k;  
begin  
  integer p, q;  
  y := 0; i := k := 1;  
  for p := 1 step 1 until n do  
    for q := 1 step 1 until m do  
      if abs(a[p, q]) > y then  
        begin y := abs(a[p, q]);  
          i := p; k := q  
        end  
      end  
    end  
  end  
end Absmax
```



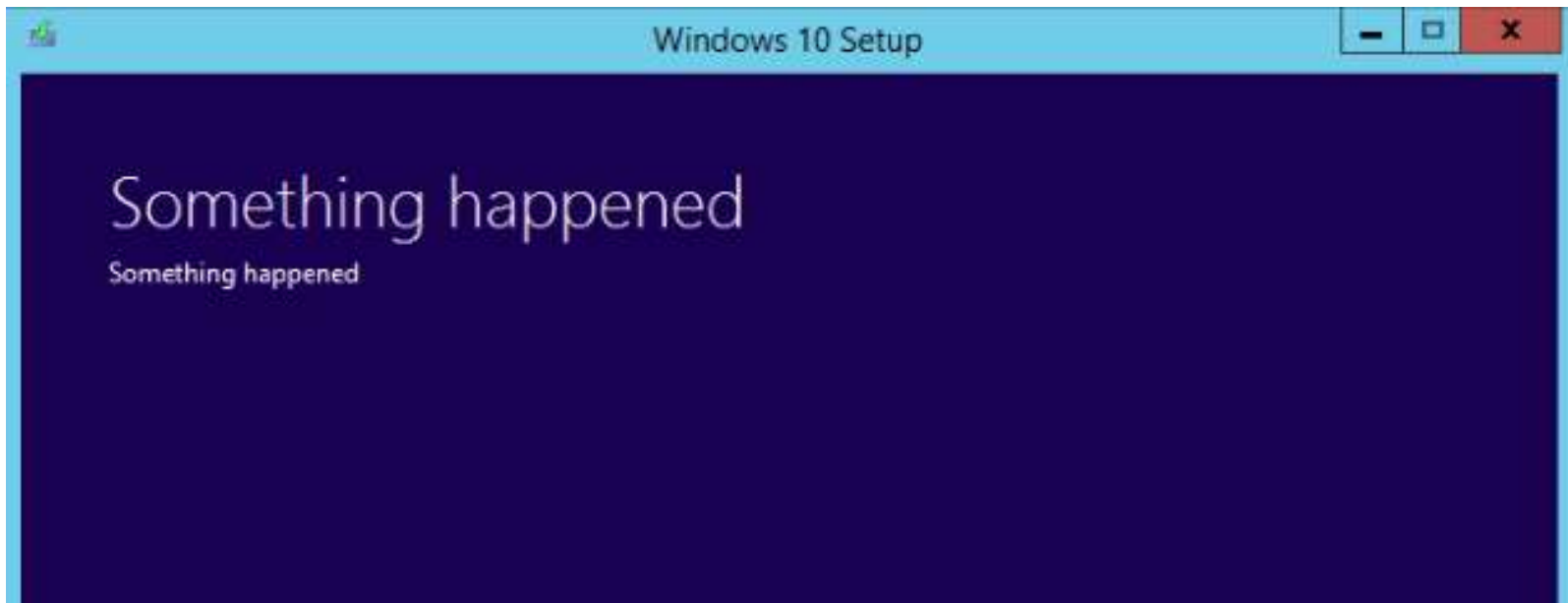
# Algol 60

```
procedure Absmax(a) Size:(n, m) Result:(y) Subscripts:(i, k);  
  value n, m; array a; integer n, m, i, k; real y;  
  comment The absolute greatest element of the matrix a, of size n by m  
    is transferred to y, and the subscripts of this element to i and k;  
begin  
  integer p, q;  
  y := 0; i := k := 1;  
  for p := 1 step 1 until n do  
    for q := 1 step 1 until m do  
      if abs(a[p, q]) > y then  
        begin y := abs(a[p, q]);  
          i := p; k := q  
        end  
      end  
end Absmax
```



# Programmers!

Who says programmers don't have a sense of humour?



# Whetstone Algol

## THE WHETSTONE KDF9 ALGOL TRANSLATOR

B. RANDELL

*The English Electric Company Ltd., Atomic Power Division, Whetstone,  
England*

### 1. Introduction

Past experience with computers and translation schemes at the Atomic Power Division has shown that users' requirements of an automatic programming scheme are to some extent conflicting. On the one hand the price paid for ease of writing and testing in a convenient language must be small, and, particularly for large or frequently used programs, the final running efficiency must be high. The

**My favourite compiler error  
message... ever!**

**No**

# VAX Mainframe

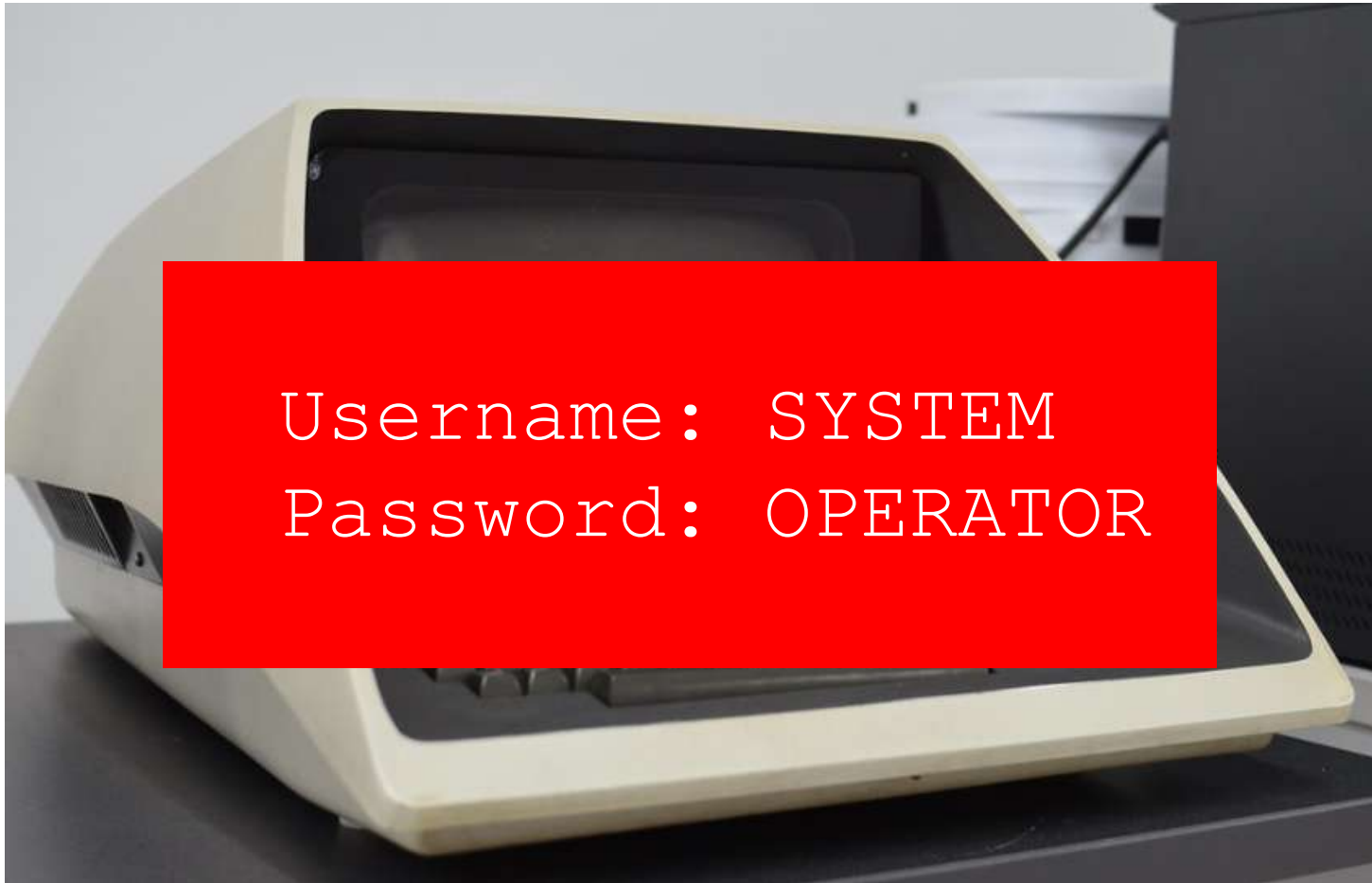


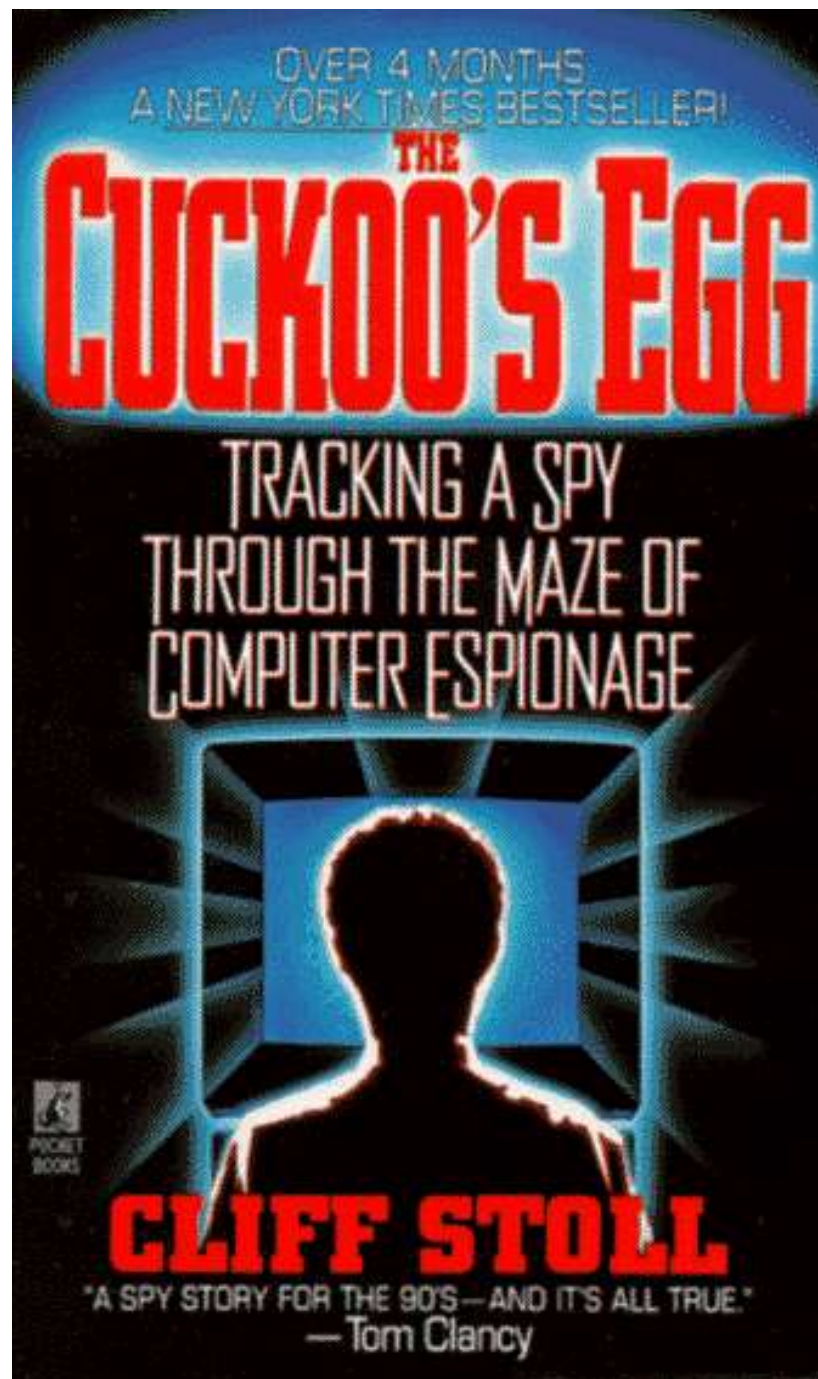


# VT05 Video Terminal



# VT05 Video Terminal





Until he tried the Air Force Systems Command, Space Division.

He first twisted on their doorknob by trying their System account, with the password of "Manager." No luck.

Until he tried the Air Force Systems Command, Space Division.

He first twisted on their doorknob by trying their System account, with the password of "Manager." No luck.

Then Guest, password of "Guest." No effect.



Until he tried the Air Force Systems Command, Space Division.

He first twisted on their doorknob by trying their System account, with the password of "Manager." No luck.

Then Guest, password of "Guest." No effect.

Then Field, password "Service":

Until he tried the Air Force Systems Command, Space Division.

He first twisted on their doorknob by trying their System account, with the password of "Manager." No luck.

Then Guest, password of "Guest." No effect.

Then Field, password "Service":

```
Username: FIELD
```

```
Password: SERVICE
```

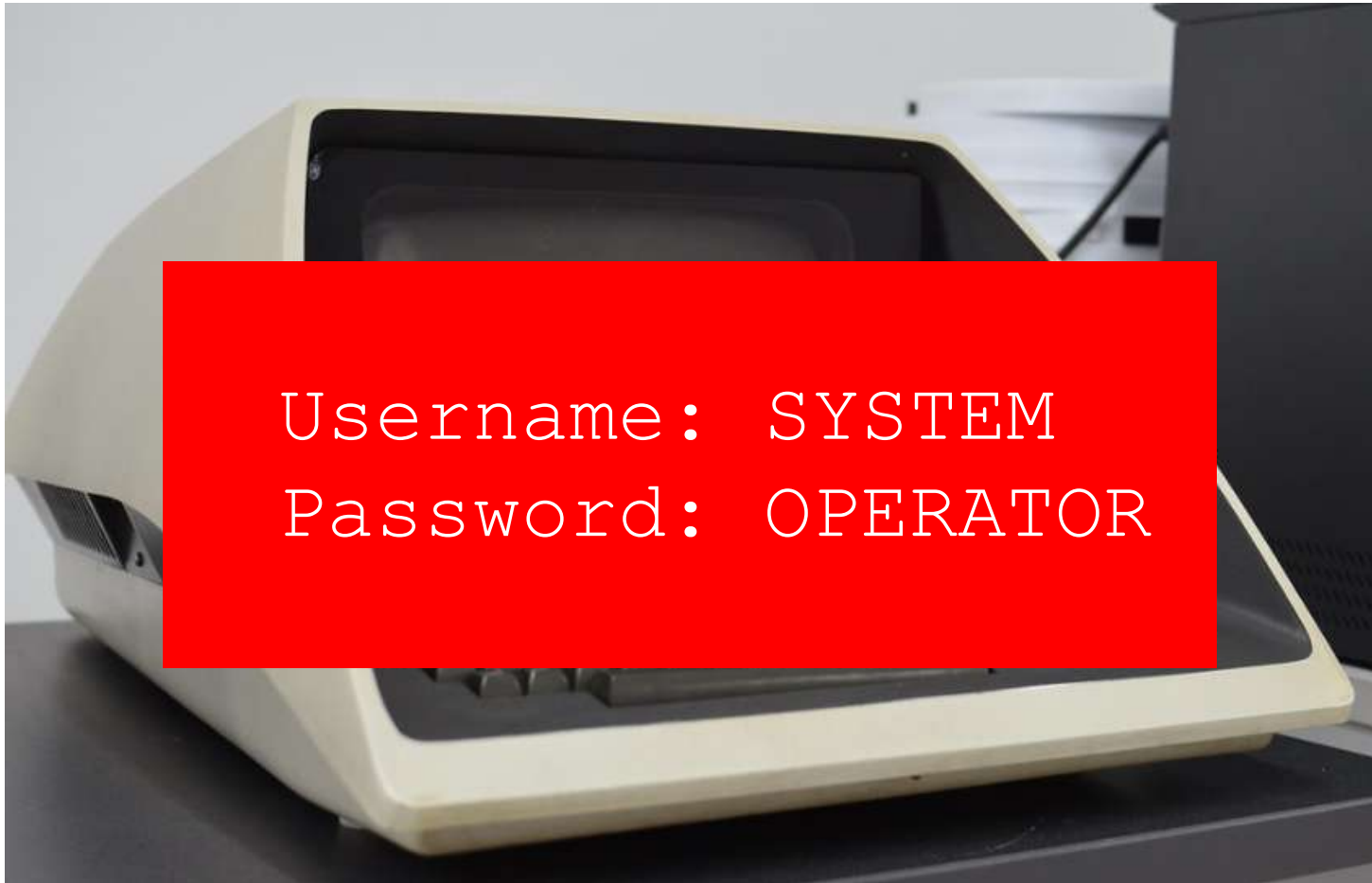
```
WELCOME TO THE AIR FORCE SYSTEM COMMAND-SPACE DIVISION
```

```
VAX/VMS 4.4
```

```
IMPORTANT NOTICE
```

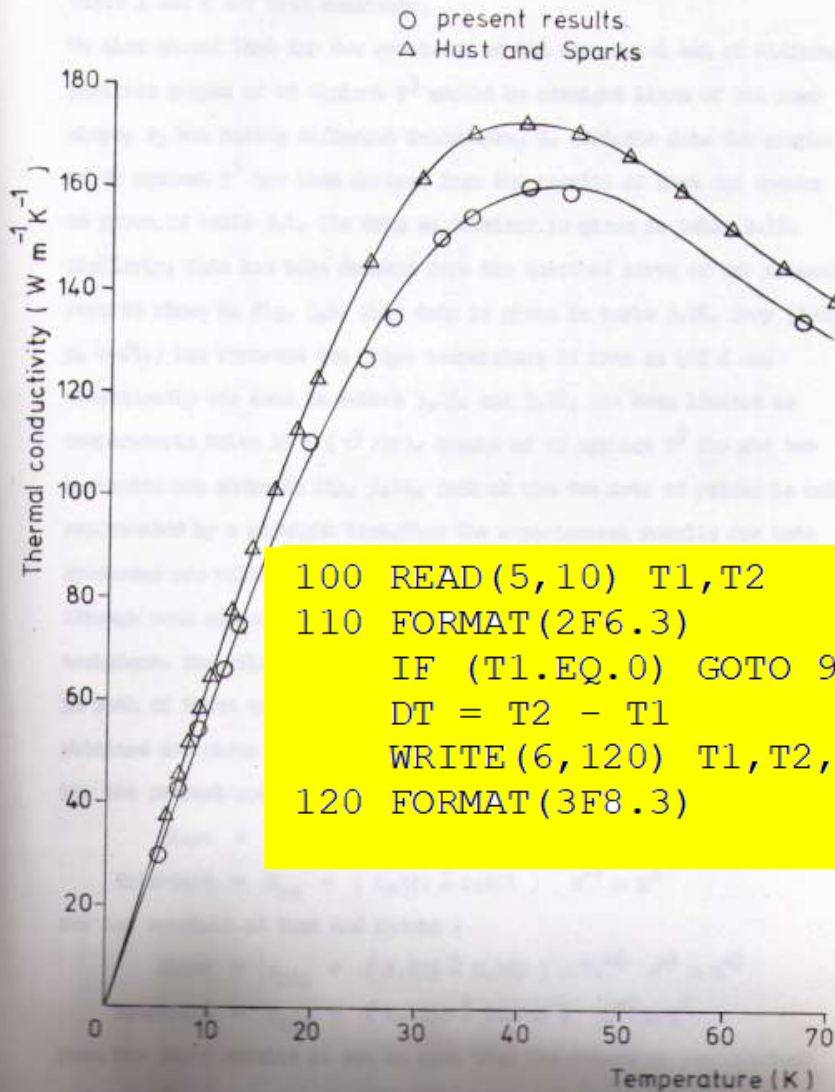
```
Computer System problems should be directed to the Information  
Systems Customer Service Section located in building 130, room  
2369.
```

# VT05 Video Terminal



# FORTRAN

Fig. 3.9. The thermal conductivity of specimen 2.



```

100 READ(5,10) T1,T2
110 FORMAT(2F6.3)
    IF (T1.EQ.0) GOTO 900
    DT = T2 - T1
    WRITE(6,120) T1,T2,DT
120 FORMAT(3F8.3)
    
```

Table 3.21. Experimental observations for the determination of the thermal conductivity of specimen 2.

R(CCG173) (ohms)	R(CCG172) (ohms)	Heat I (volts)	Heat V (volts)
741.50	700.00	0.00 549	0.13 573
467.60	413.55	0.00 548	0.13 574
303.80	266.80	0.00 723	0.17 983
206.53	178.74	0.00 896	0.22 371
181.49	154.21	0.00 895	0.22 371
105.49	86.540	0.01 175	0.29 689

Table 3.22. The thermal conductivity of specimen 2 as calculated from the data given in Table 3.21.

$T_2$ (K)	$T_1$ (K)	$T_2 - T_1$ (K)	$\dot{Q}$ (W)	$T$ (K)	$k$ ( $\text{W m}^{-1} \text{K}^{-1}$ )
5.384	5.005	0.379	0.00 799	5.195	29.76
7.022	6.760	0.262	0.00 797	6.891	42.95
9.079	8.717	0.362	0.01 395	8.898	54.50
11.647	11.186	0.460	0.02 150	11.416	65.94
12.732	12.328	0.404	0.02 146	12.530	74.97
19.469	18.992	0.478	0.03 740	19.230	110.6

# DEC PDP-11





# Some serious programming!

```
1100    WRITE(7,809)
809     FORMAT(//16X,'***** MASTERMIND GAME *****'//)
        WRITE(7,810)
810     FORMAT(' THE COMPUTER WILL CHOOSE A CODE CONSISTING
1 OF A NUMBER OF NUMERICAL'// DIGITS ARRANGED IN A CERTAIN
2 ORDER,'//' THE OBJECT OF THE GAME IS FOR THE PLAYER TO
3 DISCOVER THE CODE.')

C----- CHOOSE LEVEL OF SKILL

        WRITE(7,819)
819     FORMAT(//' THE GAME CAN BE PLAYED AT THE FOLLOWING LEVELS
1 OF SKILL : '//6X,' LEVEL 1 = MASTERMIND (EASIEST)'
2 //6X,' LEVEL 2 = SUPER-MASTERMIND (MORE DIFFICULT)')
        WRITE(7,811)
811     FORMAT (//16X,32('*'))
```



# Some serious programming!

```
1100    WRITE(7,809)
809     FORMAT(//16X,'***MASTERMIND GAME ***'//)
        WRITE(7,810)
810     FORMAT(' THE COMPUTER WILL CHOOSE A CODE CONSISTING
1 OF A NUMBER OF NUMERICAL'// DIGITS ARRANGED IN A CERTAIN
2 ORDER,'//' THE OBJECT OF THE GAME IS FOR THE PLAYER TO
3 DISCOVER THE CODE.')

C----- CHOOSE LEVEL OF SKILL

        WRITE(7,819)
819     FORMAT(//' THE GAME CAN BE PLAYED AT THE FOLLOWING LEVELS
1 OF SKILL : '//6X,' LEVEL 1 = MASTERMIND (EASIEST)'
2 //6X,' LEVEL 2 = SUPER-MASTERMIND (MORE DIFFICULT)')
        WRITE(7,811)
811     FORMAT (//16X,32('*'))
```



. R MASTER

DO YOU REQUIRE INSTRUCTIONS ?  
TYPE <YES> OR <NO> AND <RETURN>    N

I HAVE CHOSEN A CODE

YOUR GUESS ?    1234

1            1 2 3 4            X 0

YOUR GUESS ?    1256

2            1 2 5 6            0 0

YOUR GUESS ?    1537

3            1 5 3 7            0 0

YOUR GUESS ?    5184

4            5 1 8 4            X X 0 0

YOUR GUESS ?    8154

5            8 1 5 4            X 0 0 0

YOUR GUESS ?    5814

SUCCESS - THE CODE WAS :    5 8 1 4

# FORTRAN

C----- READ NEXT CARD

```
300  READ(5,902) ICASE,ICOLL,ICODE(1),IDAY,  
      1  IMONTH,IYEAR,NVEH,NOCC,IVEH,IOCC
```

```
902  FORMAT(2I4,I1,3I2,4I1)
```

C----- IF THE FIRST NUMBER ON THE CARD IS

C----- NEGATIVE, ALL THE DATA HAS BEEN READ

IF (ICASE.LT.0) GO TO 500

# First PC



# First PC



Introduced 1981

16-256K RAM

5¼" floppies (optional)

BASIC

US \$1,500

(≈\$4,500 today)

# First PC







# 300 Baud Acoustic Coupler



# 300 Baud Acoustic Coupler



# BASIC

```
LOCATE 9,21,0: PRINT "          AGCOMM - VERSION 5.7"
LOCATE 12,21,0: PRINT "A communications program for the IBM-PC"
LOCATE 13,21,0: PRINT "    and the UNB mainframe computer"
LOCATE 16,21,0: PRINT "          Author : Alan German"
LOCATE 24,20,1: PRINT "...do you require instructions (y/n)? ";
B$=INKEY$: IF B$="" THEN 1230
IF B$="y" OR B$="Y" THEN GOSUB 2070
WIDTH "com2:",255
OPEN "com2:300,e,7,1" AS 1
OPEN "lpt1:" FOR OUTPUT AS #2
GOSUB 3322
LOCATE 8,25,0: PRINT "Automatic dial-up to DATAPAC..."
LOCATE 12,25,0: PRINT "Dialling...    9"
PRINT #1, "AT FO X1 DT9,;"
FOR JJ=1 TO 6000: NEXT JJ
A$=INPUT$(LOC(1),#1)
IF INSTR(A$,"OK")=0 THEN PRINT "***** NO DIAL TONE *****"
LOCATE 16,25,0: PRINT "Dialling...    679-7500"
PRINT #1, "AT DT679-7500"
FOR JJ=1 TO 16000: NEXT JJ
A$=INPUT$(LOC(1),#1)
IF INSTR(A$,"CONNECT")=0 THEN PRINT "***** NO CARRIER *****"
CLS: GOSUB 5210
LOCATE YSTART+1,25,0: PRINT "Automatic LOGON sequence..."
LOCATE CSRLIN+2,,1
```



# Batch Files (DOS)

AUTOEXEC.BAT

and

CONFIG.SYS



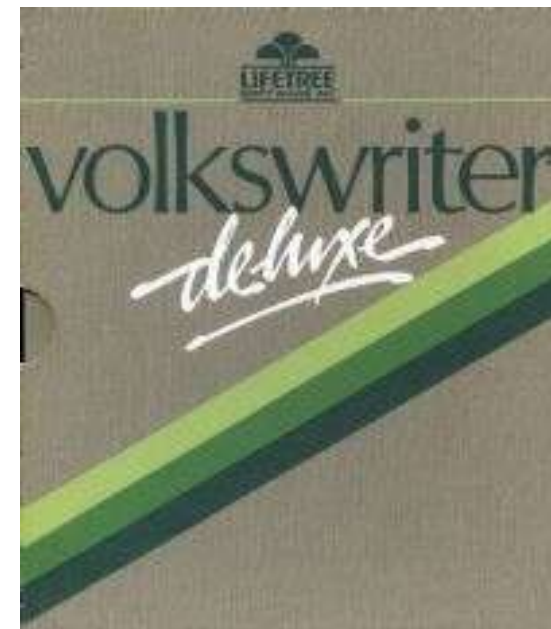
device=vdisk.sys 360

BUFFERS=20

FILES=20

# AUTOEXEC.BAT

```
mode lpt1:
superspl lpt1:=com1:/rate=2400,n,8,1/on=xon/m=32
kbbuff
zenidate
copy a:COMMAND.COM c:
copy b:VW3.OVL c:
copy b:VW3.MES c:
copy b:VW3.SYS c:
copy b:VW3.KEY c:
copy b:VW3.A?? c:
copy b:VWSTYLE.LYT c:
copy b:VWPR3.TBL c:
copy a:V.BAT c:
copy a:SUPERSPL.COM c:
copy a:ABORT.BAT c:
c:
a:vw3
```



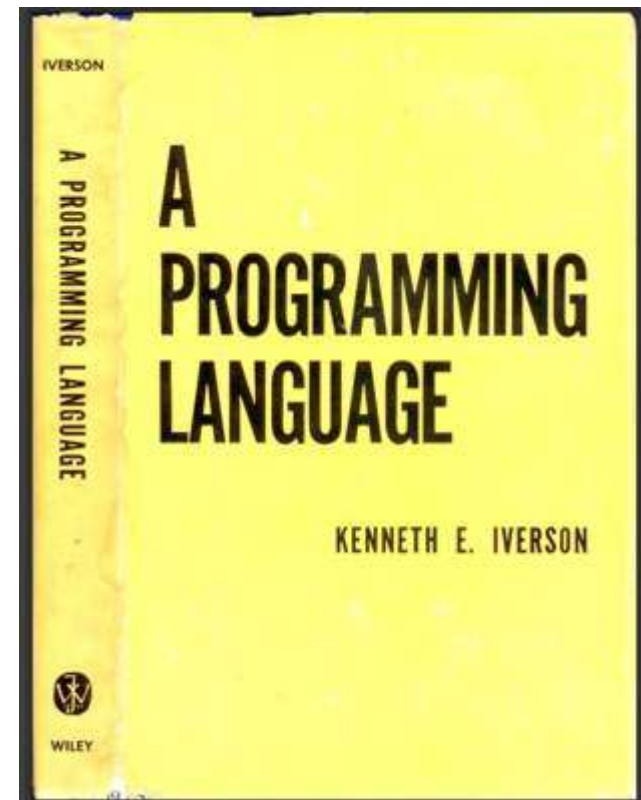
Could you write  
a function to  
sum the first  
hundred  
thousand  
integers in just  
9 keypresses?





Could you write  
a function to  
sum the first  
hundred  
thousand  
integers in just  
9 keypresses?

**APL**



# Powerful – but cryptic

```

∇FNS[□]∇
[0] FNS;N;RN;F;NI;AV;FN;F;NL;L;FD;FL;LF;T;I;□IO
[1] RN←1↑ρN←□NL 4+NI←-1+□IO←0
[2] N←N[Δ65↓(' ',AV←□AV[(, 65 97 ∘.+↓26), 145 241 ,48+↓10])↓N;]
[3] F0:→↓RN=NI←1+NI
[4] →(∧/('FNS'=3↑FN),3=ρFN←(' '≠FN)/FN←,N[NI;])ρF0
[5] L←0×NL←1↑ρF←□CR FN
[6] ' ∇',FN,'[□]∇',
[7] →(0=NL)ρF4
[8] ' ∇', (1-(LF=' ')↓1)↓LF←F[0;]
[9] F1:→(NL=L←1+L)ρF3
[10] →('R'=' 'ρLF←(1-(LF=' ')↓1)↓LF←,F[L;])ρF2,ρT←' '
[11] →(∨/(0,ρLF)=I←' 'ρLF↓':')ρF2,ρT←' '
[12] →(∨/(-10↑AV)∈' 'ρLF)ρF2
[13] →(∧/(I↑LF)∈AV)ρF2
[14] T←' '
[15] F2:LF←((↓+/I)∈-1+↓\I←1+6×0,-1↓LF=□AV[10])\LF
[16] '[',FL,']',((3-ρFL←ϕL)ρ' '),T,LF
[17] →F1
[18] F3:' ∇',□AV[10]
[19] →F0
[20] F4:'DEFN ERROR'
[21] ' ∇',FN
[22] ((7+ρFN)ρ' '),'^'
[23] →F0

```

# More comments than code!

```
[24]  # Loop through elements of Record string converting
[25]  # each lower case character to upper case
[26]  #
[27]  LOOP:I←I+1
[28]  →(I>IMAX)/HELL
[29]  #
[30]  # Create LOGICAL array to flag if current array element is lower case.
[31]  # The value of POSITION thus flags the position of the character in the
[32]  # arrays LOWER and UPPER. If POSITION is zero, the element is not a lower
[33]  # character. Otherwise, the original lower case character is replaced by
[34]  # the equivalent upper case character from the UPPER array.
[35]  #
[36]  LOGICAL←LOWER←RECORD[I]
[37]  POSITION←(+/^~LOGICAL)+1
[38]  →(POSITION>26)/LOOP
[39]  RECORD[I]←UPPER[POSITION]
[40]  →LOOP
```

# Assembler (Debug)

## MICRO-UTILITIES

Alan German

London IBM PC and Compatibles Users Group

In this article one method of producing such utilities will be reviewed. The mini-assembler contained in the DOS 2.0 version of DEBUG will be used to produce the utility program. Don't stop reading just because "assembly language" and "DEBUG" have been mentioned. You will not be required to understand how either of these work, nor even how to use them. You will merely have to follow the instructions given; all will be explained in due course...



# The Data Bus

This is the Editorial section of the newsletter so I can't resist following up on that theme of information exchange. It strikes me that this is the usual purpose of a Users' Group -- **users helping other users** by exchanging information -- with tips on good and bad software packages, help with hardware problems, advice on how to get the biggest bang for the buck, and hints on how to drive one of these computer machines most efficiently.



# PAGE.COM

The program can be used to throw a new page between files which are being printed consecutively. This feature can be invaluable if you are using a ram-spooler as a print buffer. The page "command" can also be included in batch files:

```
copy file1 prn
page
copy file2 prn
page
```

All this and yet PAGE.COM requires only 8 bytes of your disk's storage capacity!

# PAGE.COM

The program can be used to throw a new page between files which are being printed consecutively. This feature can be invaluable if you are using a ram-spooler as a print buffer. The page "command" can also be included in batch files:

```
copy file1 prn
page
copy file2 prn
page
```

All this and yet PAGE.COM requires only 8 bytes of your disk's storage capacity!



If you have followed the instructions given above precisely, your input screen should look like the following:

```
A> debug page.com
File not found
-a
XXXX:0100 mov ah,5
XXXX:0102 mov dl,0c
XXXX:0104 int 21
XXXX:0106 int 20
XXXX:0108
-rbx
BX 0000
:0
-rcx
CX 0000
:8
-w
Writing 0008 bytes
-q

A>
```

=====

October 1988 -- Volume 5, Number 9

T H E   D A T A   B U S

The Official "On-Disk" Newsletter of the  
London IBM PC & Compatibles User's Group

=====

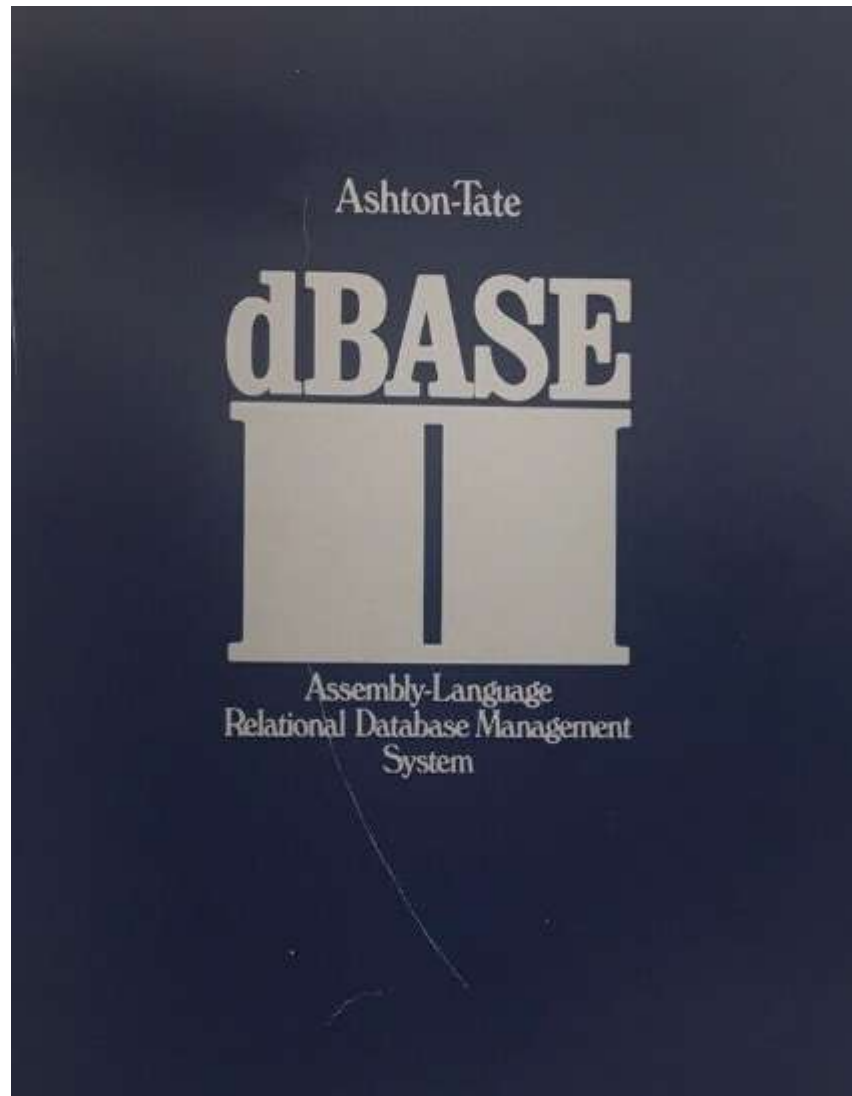
This month we have offerings across a wide range of computing applications. What do you think about structuring the newsletter in this way? What software would you like to see included in future issues. Would you like to read reviews on specific hardware and/or software products? Would you like to contribute some original material or an item of public domain software? This is YOUR newsletter, please give us some input into its format, if not its content! Call, write, or leave a message on the BBS (as soon as the sleeping Bell Canada giant wakes up!)

DATABASE MANAGEMENT

=====

TRAPDOOR.TXT   [5K]

A few hints for new users of dBASE III who wish to produce customized menus. The special feature is the inclusion of a "secret" trapdoor!

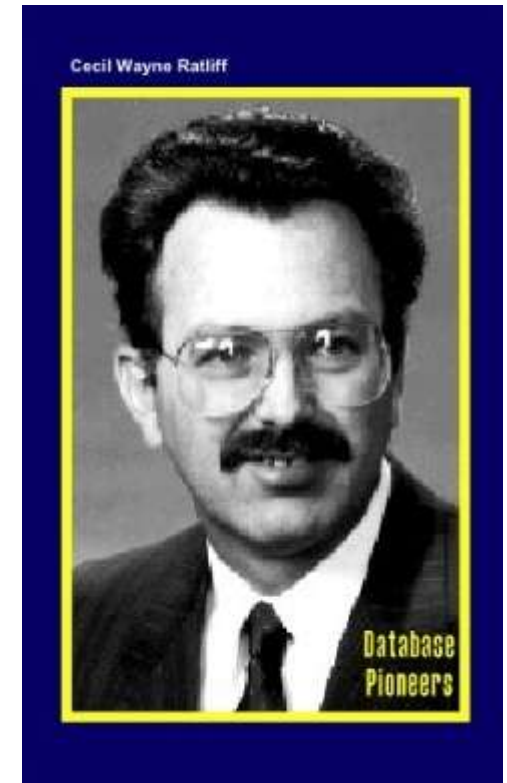


What version  
pre-dated  
dBASE II?

Wayne Ratliff  
Jet Propulsion Laboratories  
Vulcan (Mr. Spock)

George Tate  
Ratliff-Tate => Ashton-Tate

Vulcan => dBASE II



## A TRAPDOOR TO dBASE III

Alan German  
London IBM-PC and Compatibles Users Group  
P.O. Box 1141, Station B, London, Ontario, N6A 5K2

Recently, I have been busy writing a custom program in dBASE to handle memberships for a non-profit organization. The program design features a multiple-menu system which makes it easy to use for the Membership Secretary who is not too familiar with computers.

Skimming the dBASE manual, a few assorted text books, plus a lot of trial and error, produced a basic menu structure. New users of dBASE might find some of the techniques adopted to be useful, so the full code for the main menu module is provided below.

```
set color to w+
@ 6,36 say 'MAIN MENU'
set color to w
@ 8,20 say '1-->  ADD a new membership record'
@ 9,20 say '2-->  UPDATE an existing membership record'
@ 10,20 say '3-->  CHANGE a name and address'
@ 11,20 say '4-->  PRINT reports'
@ 12,20 say '5-->  UTILITY routines'
@ 14,20 say '0-->  QUIT database program'
set color to w+
option='0'
@ 18,20 say 'Option Number ?  ' get option
set color to w
read
```

GENERAL LEDGER  
Ottawa PC Users' Group

Version 4.6

MAIN MENU

- 1--> REVENUE transaction
- 2--> EXPENSE transaction
- 3--> OTHER transaction
- 4--> UPDATE transaction
- 5--> PRINT reports
- 6--> UTILITY routines

0--> QUIT database program

Option Number ?



```
do case

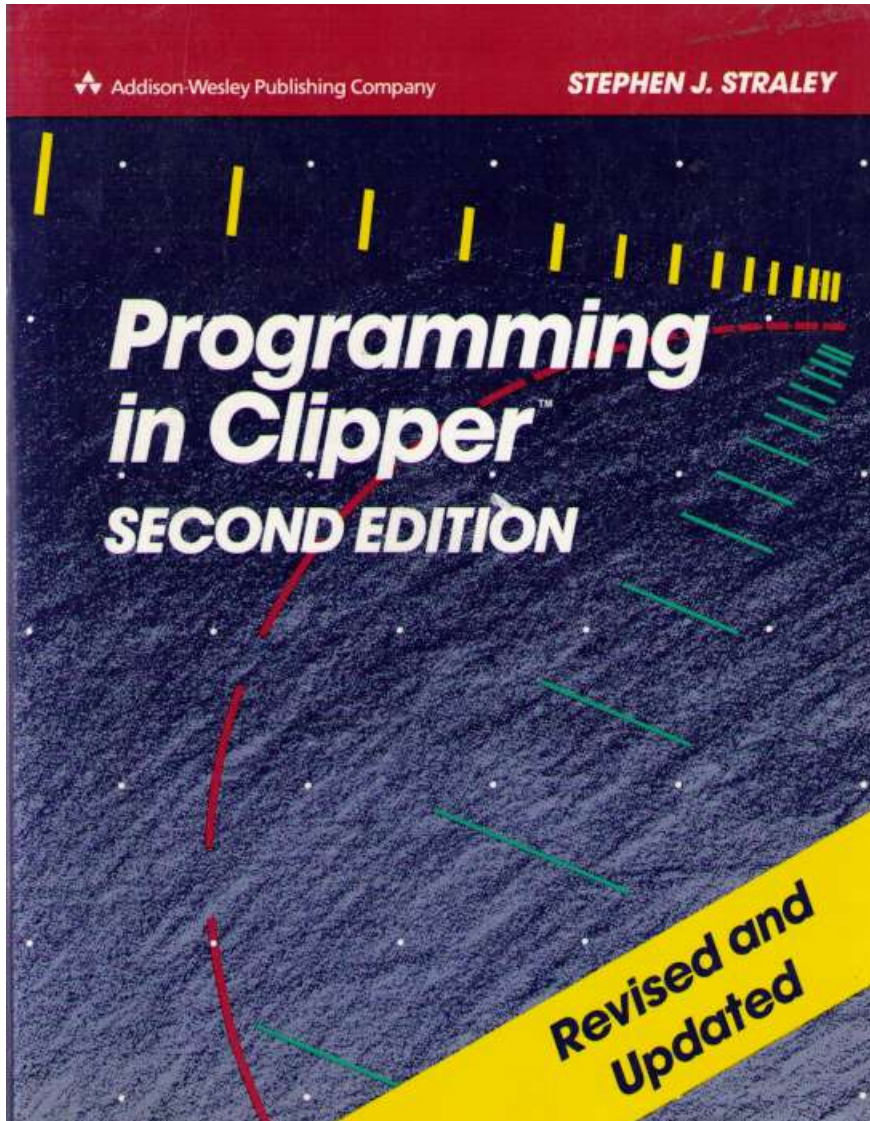
* add a new membership record
  case option='1'
    do add
* update an existing membership record
  case option='2'
    do update
* change an existing name and address
  case option='3'
    do change
* print reports
  case option='4'
    do print
* utility routines
  case option='5'
    do utility
* secret trapdoor back to dBASE system
  case option='9'
    return
* exit from dBASE for 0 or return
  case option='0' .or. len(option)=0
    quit

endcase
```



```
*:*****
*:
*:      Program:  ADD_TRAN.PRG
*:
*:      System:   AGL
*:      Author:   Alan German
*:      Copyright (c) 1991, Alan German
*:      Last modified: 12/14/91      19:43
*:
*:      Called by: REVENUE.PRG
*:                  : EXPENSE.PRG
*:                  : OTHER.PRG
*:
*:      Calls:    LO_WINDO.PRG
*:                  : SHOW_CHT.PRG
*:                  : ADD_SCR.PRG
*:                  : TRAN_ERR.PRG
*:                  : SAV_OPT.PRG
*:                  : EDI_OPT.PRG
*:                  : ABO_OPT.PRG
*:                  : GET_OPT.PRG
*:                  : UPD_BAL.PRG
*:
*:      Uses:     AGTEMP.DBF
*:                  : GL.DBF
*:                  Alias: LEDGER
*:
*:      Indexes:  DATE.NDX
*:
*:      Documented 12/22/91 at 11:11
*:
*:      SNAP! version 4.01
*:*****
```

# Clipper



Nantucket Corporation

Replacement programming language for dBASE III

Compiler/linker

Create stand-alone MS-DOS program

DOSBox 0.74-2, Cpu speed: 3000 cycles, Frameskip 0, Program: CLIPPER

```
E:\DB>cl main
E:\DB>set lib=x:\clipper\lib

E:\DB>set include=x:\clipper\include

E:\DB>set pll=x:\clipper\pll

E:\DB>x:\clipper\bin\clipper main
Clipper (R) Version 5.2
Copyright (c) 1985-1993, Computer Associates International, Inc.
Microsoft C Floating Point Support Routines
Copyright (c) Microsoft Corp 1984-1987. All Rights Reserved.
366K available
Compiling MAIN.PRG
Compiling INU_TITL.PRG
```

'P\_INTCOL'

MAIN.OBJ

'P\_INTCHK'

MAIN.OBJ

```
warning wrt0022: .EXE may not execute properly -- undefined symbols
260K
1 warning message(s)
```

DOSBox 0.74-2, Cpu speed: 3000 cycles, Frameskip 0, Program: CLIPPER

```
E:\DB>cl main
E:\DB>set lib=x:\clipper\lib

E:\DB>set include=x:\clipper\include

E:\DB>set pll=x:\clipper\pll

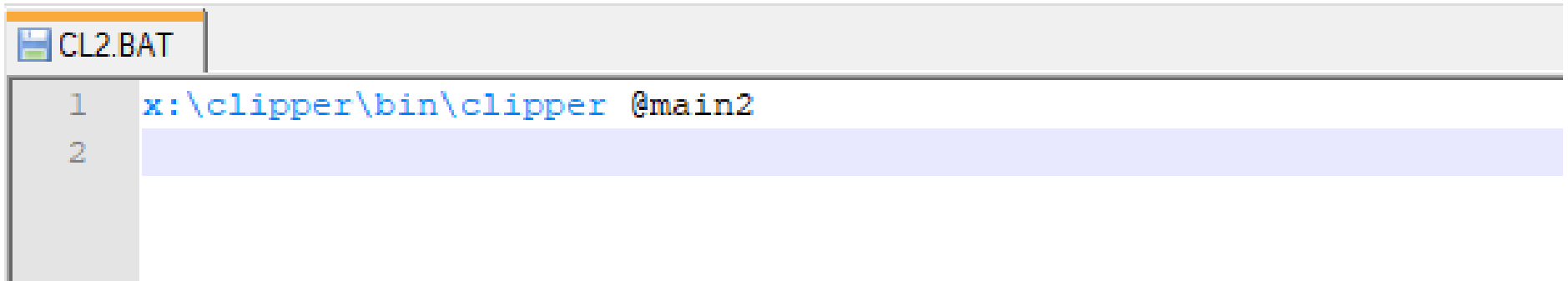
E:\DB>x:\clipper\bin\clipper main
Clipper (R) Version 5.2
Copyright (c) 1985-1993, Computer Associates International, Inc.
Microsoft C Floating Point Support Routines
Cop
366
Com
Com
```

**EXE may not execute properly - -  
undefined symbols**

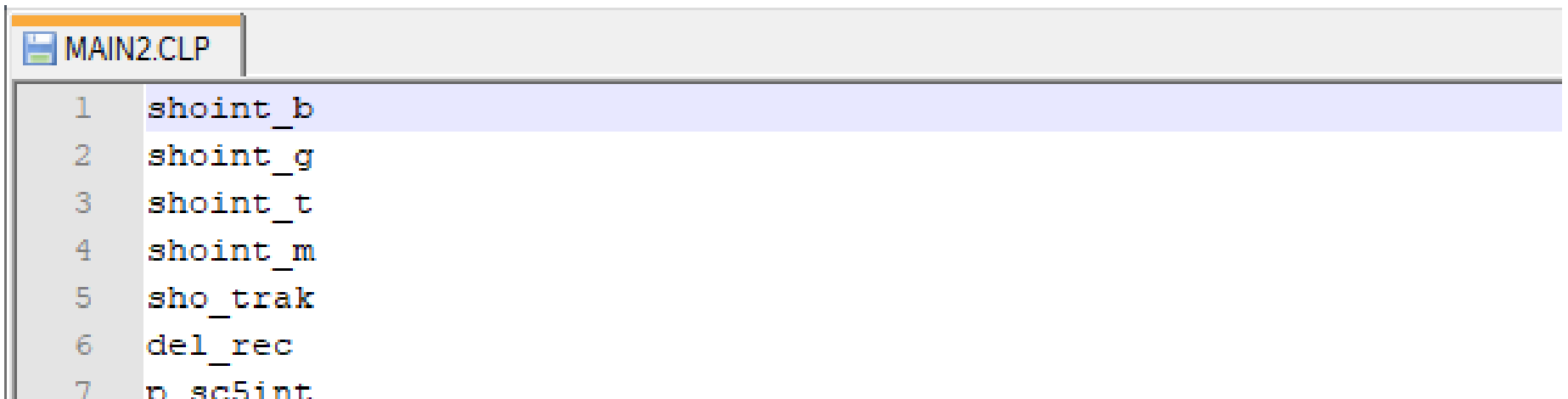
```
'P_INTCHK'                                MAIN.OBJ

warning wrt0022: .EXE may not execute properly -- undefined symbols
260K
1 warning message(s)
```

# Batch Files



```
CL2.BAT
1 x:\clipper\bin\clipper @main2
2
```



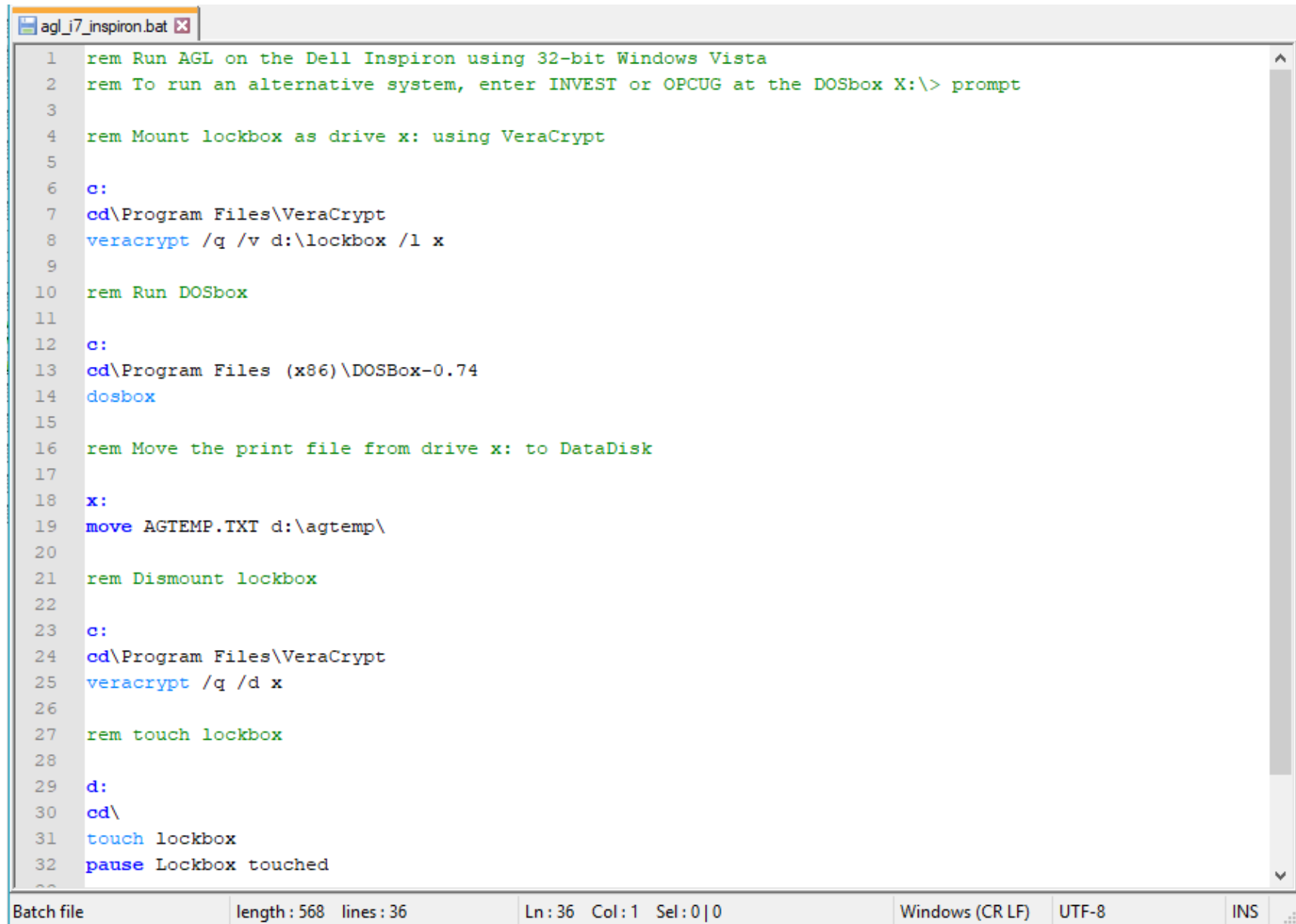
```
MAIN2.CLP
1 shoint_b
2 shoint_g
3 shoint_t
4 shoint_m
5 sho_trak
6 del_rec
7 n sc5int
```



CL.BAT

```
1 cd\develop\inv_code
2 set lib=x:\clipper\lib
3 set include=x:\clipper\include
4 set pl1=x:\clipper\pl1
5 x:\clipper\bin\clipper %1
6 if not errorlevel 1 x:\clipper\bin\rtlink file %1,main2
7
```

# Batch Files – Windows 10



```
1 rem Run AGL on the Dell Inspiron using 32-bit Windows Vista
2 rem To run an alternative system, enter INVEST or OPCUG at the DOSbox X:\> prompt
3
4 rem Mount lockbox as drive x: using VeraCrypt
5
6 c:
7 cd\Program Files\VeraCrypt
8 veracrypt /q /v d:\lockbox /l x
9
10 rem Run DOSbox
11
12 c:
13 cd\Program Files (x86)\DOSBox-0.74
14 dosbox
15
16 rem Move the print file from drive x: to DataDisk
17
18 x:
19 move AGTEMP.TXT d:\agtemp\
20
21 rem Dismount lockbox
22
23 c:
24 cd\Program Files\VeraCrypt
25 veracrypt /q /d x
26
27 rem touch lockbox
28
29 d:
30 cd\
31 touch lockbox
32 pause Lockbox touched
```

Batch file      length: 568 lines: 36      Ln: 36 Col: 1 Sel: 0 | 0      Windows (CR LF)      UTF-8      INS

```
agl_i7_inspiron.bat x
1 rem Run AGL on the Dell Inspiron using 32-bit Windows Vista
2 rem To run an alternative system, enter INVEST or OPCUG at the DOSbox X:\> prompt
3
4 rem Mount lockbox as drive x: using VeraCrypt
5
6 c:
7 cd\Program Files\VeraCrypt
8 veracrypt /q /v d:\lockbox /l x
9
10 rem Run DOSbox
11
12 c:
13 cd\Program Files (x86)\DOSBox-0.74
14 dosbox
15
16 rem Move the print file from drive x: to DataDisk
17
18 x:
19 move AGTEMP.TXT d:\agtemp\
20
21 rem Dismount lockbox
22
23 c:
24 cd\Program Files\VeraCrypt
25 veracrypt /q /d x
26
27 rem touch lockbox
28
29 d:
30 cd\
31 touch lockbox
32 pause Lockbox touched
33
```

Batch file      length : 568   lines : 36      Ln : 36   Col : 1   Sel : 0 | 0      Windows (CR LF)   UTF-8   INS

Double-click on  
one batch file -  
14 commands!



Borland International  
(Philippe Kahn)

Integrated Development  
Environment (IDE)

Edit/Compile/Run

Turbo = Fast!

\$49.95

# MenuGen

This program is designed to assist TURBO Pascal programmers to develop routines which include menu-driven input screens. In particular, the program will automatically generate a segment of source code which will produce a data entry screen in the form:

Menu Title

1. Option one
2. Option two

Enter selection (1-2)

# MenuGen

This program is designed to assist TURBO Pascal programmers to develop routines which include menu-driven input screens. In particular, the program will automatically generate a segment of source code which will produce a data entry screen in the form:

Menu Title

1. Option one
2. Option two

Enter selection (1-2)

```
{write menu title to output file}  
  
{compute x and y screen coordinates required for title  
  line to be centered horizontally, and for the whole  
  menu to be centered vertically, on the display screen}  
  
xValue:=(80-Length(titleText)) div 2 + 1;  
yValue:=(25-4-count) div 2 + 1;
```



# FileList

Print a file, or a group of files, in a paginated format.

A header specifies the file name and page number

```
FileList Version 1.0
Listing of file <FILNAMIN.PAS>
Page 1

-----

procedure FilNaminvar filename: str14; extension: str3;
(
  (Author : Alan Berman)
  (Version : 1.0)
)
{
  This procedure reads in a file name and provides a default
  extension if no extension is supplied as input from the
  keyboard.

  The default extension is supplied as a parameter in the call
  for the procedure.

  The filename is returned as a text string which includes the
  drive (if specified) and the extension.

  Using the example of a default extension of .DAT, the
  procedure call takes the form:
      FilNamin(filename,'DAT');
}

var
  len: integer; (length of filename string if required)

begin
  (read in the filename string)
  read(filename);

  (if there is no period in the filename string then insert a
  period and the default extension after the filename)

  if Pos('.',filename)=0 then
  begin
    len:=Length(filename);
    Insert('.'+extension,filename,len+1);
  end;

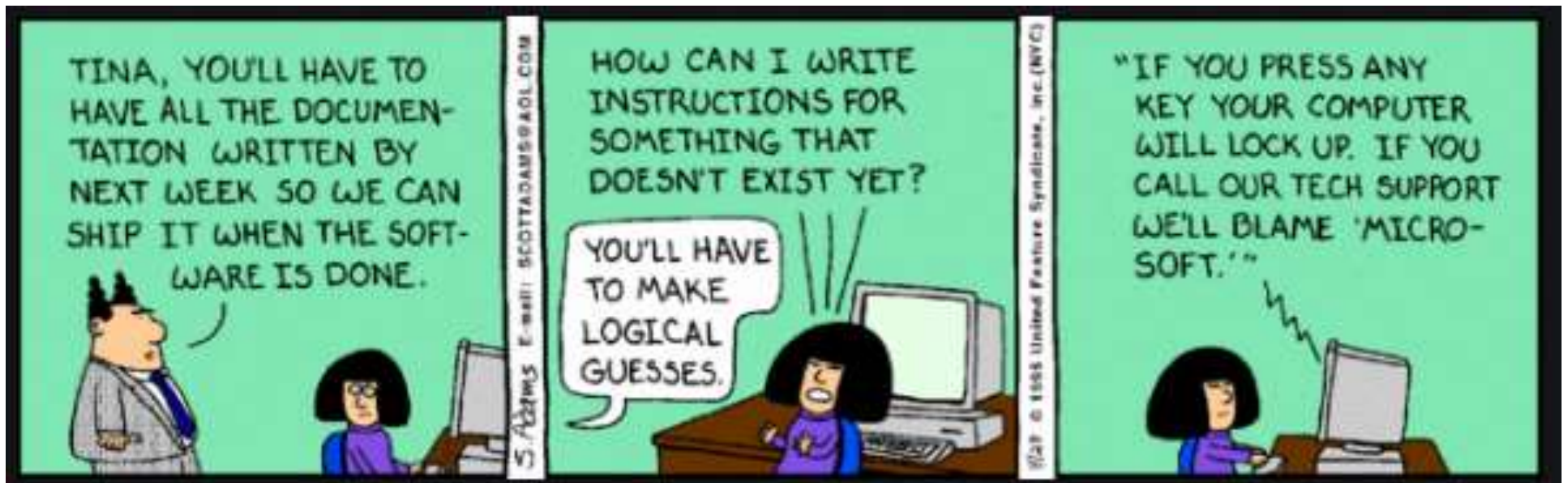
end;
```

# Documentation

Who says documentation writers  
don't have a sense of humour?

# Documentation

Who says documentation writers don't have a sense of humour?





Michel Renaud 

Sep 26 '19 

... from the index of the dBASE III Plus manual back in the '80s (when software came with printed manuals so big you could kill someone with them):

- Endless Loop: See Loop, Endless ... ..
- Loop, Endless: See Endless Loop



[REPLY](#)

## dBASE III Plus Reference

PC1

### Table of Contents

1	Introduction.....	3
2	Entering and Exiting dBASE III PLUS .....	3

# FileList

Filelist is a program which will print a file, or a group of files, in a paginated format. Each page contains a header specifying the file name and page number, so it is easy to keep track of long file or multiple file listings.

Note the restriction that input from text files is limited to 80 characters per line (because I don't use lines any longer than that!)

## NEW FEATURES

For users who are familiar with earlier versions of this program, Version 2.0 has the following new features:

- menu driven interface
- colour screens (with the option of setting up for monochrome)
- specify a default file extension
- specify the drive and sub-directory for data files
- customize the output page format
- select a data file containing a list of files to be printed

# FileList

Filelist is a program which will print a file, or a group of files, in a paginated format. Each page contains a header specifying the file name and page number, so it is easy to keep track of long file or multiple file listings.

Note the restriction that input from text files is limited to 80 characters per line (because I don't use lines any longer than that!)

## NEW FEATURES

For users who are familiar with earlier versions of this program, Version 2.0 has the following new features:

- menu driven interface
- colour screens (with the option of setting up for monochrome)
- specify a default file extension
- specify the drive and sub-directory for data files
- customize the output page format
- select a data file containing a list of files to be printed



# FileList

Filelist is a program which will print a file, or a group of files, in a paginated format. Each page contains a header specifying the file name and page number, so it is easy to keep track of long file or multiple file listings.

Note the restriction that input from text files is limited to 80 characters per line (because I don't use lines any longer than that!)

## NEW FEATURES

For users who are familiar with earlier versions of this program, Version 2.0 has the following new features:

- menu driven interface
- colour screens (with the option of setting up for monochrome)
- specify a default file extension
- specify the drive and sub-directory for data files
- customize the output page format
- select a data file containing a list of files to be printed

Filelist

Version 2.0

MAIN MENU

1. Print a file
2. Print a list of files
3. Change default drive\path
4. Change default extension
5. Utilities
  
0. Quit to DOS

Enter selection (0-5)

Defaults

List of files:	FILELIST.LST
Drive/Path:	C:\
Extension:	PAS

---

```
procedure FilNamIn(var filename: str14; extension: str3);  
  
(Author :   Alan German)  
(Version : 1.0      )  
  
<  
  This procedure reads in a file name and provides a default  
  extension if no extension is supplied as input from the  
  keyboard.  
  
  The default extension is supplied as a parameter in the call  
  for the procedure.  
  
  The filename is returned as a text string which includes the  
  drive (if specified) and the extension.  
  
  Using the example of a default extension of .DAT, the  
  procedure call takes the form:  
      FilNamIn(filename,'DAT');  
>  
  
var  
  len: integer;  (length of filename string if required)  
  
begin  
  (read in the filename string)  
  
  read(filename);  
  
  (if there is no period in the filename string then insert a  
  period and the default extension after the filename)  
  
  if Pos('.',filename)=0 then  
  begin  
    len:=Length(filename);  
    Insert('.'+extension,filename,len+1);  
  end;  
  
end;
```

# Visual Basic for Applications (VBA)

- Applies to MS Office (Word, Excel...)
- Steep learning curve!
- Find a good book
- Use Dr. Google
  - => vba excel write number to cell
  - => vba excel list worksheets



[Home](#)
[PUBLIC](#)
[Stack Overflow](#)
[Tags](#)
[Users](#)
[Jobs](#)
[TEAMS](#)
[+ Create Team](#)

# Writing an input integer into a cell



7



I am writing a quick application myself - first project, however I am trying to find the VBA code for writing the result of an input string to a named cell in Excel.

For example, a input box asks the question "Which job number would you like to add to the list?"... the user would then enter a reference number such as "FX1234356". The macro then needs to write that information into a cell, which I can then use to finish the macro (basically a search in some data).

[excel](#)
[vba](#)
[share](#) [improve this question](#)
[edited Jun 12 at 19:17](#)


[braX](#)

5,066 ● 4 ● 11 ● 28

[asked Nov 2 '08 at 19:16](#)


[David Max](#)

470 ● 5 ● 10 ● 15

2 Question title is misleading. Your input is not an integer. – [Jason Z](#) Nov 2 '08 at 19:37

[add a comment](#)

## 4 Answers

[active](#)
[oldest](#)
[votes](#)


10



You can use the Range object in VBA to set the value of a named cell, just like any other cell.

```
Range("C1").Value = Inputbox("Which job number would you like to add to the list?")
```

Where "C1" is the name of the cell you want to update.

My Excel VBA is a little bit old and crusty, so there may be a better way to do this in newer versions of Excel.

[share](#) [improve this answer](#)
[answered Nov 2 '08 at 19:33](#)


[Bill the Lizard](#)

289k ● 156 ● 493 ● 785

[Home](#)
[PUBLIC](#)
[Stack Overflow](#)
[Tags](#)
[Users](#)
[Jobs](#)
[TEAMS](#)
[+ Create Team](#)

## Writing an input integer into a cell



I am writing a quick application myself - first project, however I am trying to find the VBA code for writing the result of an input string to a named cell in Excel.

7



For example, a input box asks the question "Which job number would you like to add to the list?"... the user would then enter a reference number such as "FX1234356". The macro then needs to write that information into a cell, which I can then use to finish the macro (basically a search in some data).


[excel](#)
[vba](#)

**Range("C1").Value = 21**

[add a comment](#)

4 Answers

[active](#)
[oldest](#)
[votes](#)


You can use the Range object in VBA to set the value of a named cell, just like any other cell.

10

```
Range("C1").Value = Inputbox("Which job number would you like to add to the list?")
```



Where "C1" is the name of the cell you want to update.



My Excel VBA is a little bit old and crusty, so there may be a better way to do this in newer versions of Excel.

[share](#) [improve this answer](#)

answered Nov 2 '08 at 19:33



[Bill the Lizard](#)

289k ● 156 ● 493 ● 785



## VBA – Macro to List all Sheets in a Workbook

The following macro loops through every sheet in a workbook and writes the tab name of each sheet sequentially to a sheet you choose. This could be handy for a quick list of every sheet in a workbook with many sheets.

### List all Worksheets in a Workbook

To use the macro just replace the word Sheet1(it appears twice) in the code with the tab name where you would like the results. Make sure there isn't any important information on the output tab because it clears the data there before writing to it.

```
Sub ListSheets()  
  
Dim ws As Worksheet  
Dim x As Integer  
  
x = 1  
  
Sheets("Sheet1").Range("A:A").Clear  
  
For Each ws In Worksheets  
    Sheets("Sheet1").Cells(x, 1) = ws.Name  
    x = x + 1  
Next ws  
  
End Sub
```

## VBA – Macro to List all Sheets in a Workbook

The following macro loops through every sheet in a workbook and writes the tab name of each sheet sequentially to a sheet you choose. This could be handy for a quick list of every sheet in a workbook with many sheets.

**For Each ws In Worksheets  
Sheets("Sheet1").Cells(x, 1) = ws.Name**

```
Sub ListSheets()  
  
Dim ws As Worksheet  
Dim x As Integer  
  
x = 1  
  
Sheets("Sheet1").Range("A:A").Clear  
  
For Each ws In Worksheets  
    Sheets("Sheet1").Cells(x, 1) = ws.Name  
    x = x + 1  
Next ws  
  
End Sub
```

# Excel

NASS\_create\_hyperlinks\_2018\_05\_28.xlsm - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW DEVELOPER

Clipboard Font Alignment

Control button with Underlying macro

Create Hyperlinks Remove Hyperlinks Remove Case Numbers

Create Hyperlinks macro uses NHTSA's updated web server (<https://crashviewer.nhtsa.dot.gov/>)

Checkpoint (Set N pause the display every N rows) 100

NASS Case Number	NASS Case ID	URL	View Case
2009-02-001	760011613	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011613">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011613</a>	<a href="#">2009-02-001</a>
2009-02-002	760011633	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011633">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011633</a>	<a href="#">2009-02-002</a>
2009-02-003	760011634	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011634">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011634</a>	<a href="#">2009-02-003</a>
2009-02-004	760011652	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011652">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011652</a>	<a href="#">2009-02-004</a>
2009-02-005	760011653	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011653">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011653</a>	<a href="#">2009-02-005</a>
2009-02-006	760011654	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011654">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011654</a>	<a href="#">2009-02-006</a>
2009-02-007	760011671	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011671">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011671</a>	<a href="#">2009-02-007</a>
2009-02-008	760011672	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011672">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011672</a>	<a href="#">2009-02-008</a>
2009-02-009	760011673	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011673">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011673</a>	<a href="#">2009-02-009</a>
2009-02-010	760011691	<a href="https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011691">https://crashviewer.nhtsa.dot.gov/nass-cds/CaseForm.aspx?xsl=main.xsl&amp;CaseID=760011691</a>	<a href="#">2009-02-010</a>

1997-2003 2004-2008 2009-2014 2015 Updated\_URL\_Generator

READY 100%

# Add Hyperlink

```
'URL base for 2004-2010

address_base = "https://crashviewer.nhtsa.dot.gov/nass-cds/"
CaseForm.aspx?xsl=main.xsl&CaseID="

' Get the Case ID number from the column B in the current row
' and append this to the base address

case_id = Range("B" + CStr(N)).Value
address_text = address_base & case_id

' Add a hyperlink to column D

Range("D" + CStr(N)).Select
ActiveSheet.Hyperlinks.Add Anchor:=Selection, Address:=address_text, _
TextToDisplay:=case_number
```

# Add Hyperlink

```
'URL base for 2004-2010
```

```
address_base = "https://crashviewer.nhtsa.dot.gov/nass-cds/"  
CaseForm.aspx?xsl=main.xsl&CaseID="
```

```
' Get the Case ID number from the column B in the current row  
' and append this to the base address
```

```
case_id = Range("B" + CStr(N)).Value  
address_text = address_base & case_id
```

```
' Add a hyperlink to column D
```

```
Range("D" + CStr(N)).Select  
ActiveSheet.Hyperlinks.Add Anchor:=Selection, Address:=address_text, _  
TextToDisplay:=case_number
```

# Add Hyperlink

```
'URL base for 2004-2010
```

```
address_base = "https://crashviewer.nhtsa.dot.gov/nass-cds/"  
CaseForm.aspx?xsl=main.xsl&CaseID="
```

```
' Get the Case ID number from the column B in the current row  
' and append this to the base address
```

```
case_id = Range("B" + CStr(N)).Value  
address_text = address_base & case_id
```

```
' Add a hyperlink to column D
```

```
Range("D" + CStr(N)).Select  
ActiveSheet.Hyperlinks.Add Anchor:=Selection, Address:=address_text, _  
TextToDisplay:=case_number
```



# Add Hyperlink

```
'URL base for 2004-2010
```

```
address_base = "https://crashviewer.nhtsa.dot.gov/nass-cds/"  
CaseForm.aspx?xsl=main.xsl&CaseID="
```

```
' Get the Case ID number from the column B in the current row  
' and append this to the base address
```

```
case_id = Range("B" + CStr(N)).Value  
address_text = address_base & case_id
```

```
' Add a hyperlink to column D
```

```
Range("D" + CStr(N)).Select  
ActiveSheet.Hyperlinks.Add Anchor:=Selection, Address:=address_text, _  
TextToDisplay:=case_number
```



NASS\_VEHS\_W\_RSR\_EDR V01A\_2017-09-03.xlsm - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW DEVELOPER

Clipboard Font Alignment Number Styles Cells Editing

C6 : 2015-72-041-V01

1		NASS VEHICLES FITTED WITH HEAD CURTAINS AND EDR REPORT IS AVAILABLE MY: 2009 - 2014 W/ CURTAIN DEPLOYMENT				
2	NASS_VEH	NASS_CASE	NASS Case ID	VEHNO	View Case	View Web EDR Report
3	2015-41-095-V02	2015-41-095	773018614	2	<a href="#">2015-41-095</a>	<a href="#">2015-41-095-V02</a>
4	2015-74-003-V02	2015-74-003	688018802	2	<a href="#">2015-74-003</a>	<a href="#">2015-74-003-V02</a>
5	2015-41-060-V01	2015-41-060	773018354	1	<a href="#">2015-41-060</a>	<a href="#">2015-41-060-V01</a>
6	2015-72-041-V01	2015-72-041	896019026	1	<a href="#">2015-72-041</a>	<a href="#">2015-72-041-V01</a>
7	2015-41-126-V01	2015-41-126	773018815	1	<a href="#">2015-41-126</a>	<a href="#">2015-41-126-V01</a>
8	2015-48-103-V01	2015-48-103	208019220	1	<a href="#">2015-48-103</a>	<a href="#">2015-48-103-V01</a>
9	2015-48-103-V01	2015-48-103	208019220	1	<a href="#">2015-48-103</a>	<a href="#">2015-48-103-V01</a>
10	2015-08-122-V01	2015-08-122	972018889	1	<a href="#">2015-08-122</a>	<a href="#">2015-08-122-V01</a>
11	2015-49-134-V02	2015-49-134	554019337	2	<a href="#">2015-49-134</a>	<a href="#">2015-49-134-V02</a>
12	2015-76-084-V01	2015-76-084	703018879	1	<a href="#">2015-76-084</a>	<a href="#">2015-76-084-V01</a>
13	2015-82-008-V02	2015-82-008	209017998	2	<a href="#">2015-82-008</a>	<a href="#">2015-82-008-V02</a>
14	2015-05-062-V01	2015-05-062	211017500	1	<a href="#">2015-05-062</a>	<a href="#">2015-05-062-V01</a>
15	2015-49-112-V01	2015-49-112	554019157	1	<a href="#">2015-49-112</a>	<a href="#">2015-49-112-V01</a>
16	2015-79-146-V01	2015-79-146	727019754	1	<a href="#">2015-79-146</a>	<a href="#">2015-79-146-V01</a>

WORKSHEET Strip\_space\_kmh Conversion (dV no accel) Conversion (dV) ...

READY 100%


**NHTSA**  
NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION  
Print and Braille Only

NASS Case		
Crash Overview		
Vehicle 1		
Vehicle 2		
Vehicle 3		
Crash		
POL	72	
Case Number	041	
Status	K	
Weight	4,000	
Crash Date	04/07/00	
Day Of Week	Saturday	
Crash Time	05:50	
Crash Summary		
Crash Type	Multi-vehicle	
Configuration	Head-on	
Summary	Vehicles #1, #2 and #3 were ahead. The front of V1 impacted	
Vehicles		
Vehicle	Year	Make
1	2000	CADILLAC
2	2000	MERCUY
3	2010	FORD
Persons		
Vehicle Occupant Role	Seat	Restraints
1 1 Driver	Front-Left	Air Bag Male
2 1 Driver	Front-Left	Manual-Lock
2 2 Passenger	Front-Right	Manual-Lock

Images may not be to scale.

Vehicle 1 - Frontleftoblique  
Front/Left Oblique

A photograph showing a dark-colored sedan from a front-left oblique perspective. The vehicle is parked on a gravel surface. The front end is heavily damaged, with the hood crumpled and the front left headlight area crushed. A white measuring tape is stretched across the front of the car, indicating the extent of the damage. The background shows a chain-link fence and some trees under a clear blue sky.

Model	Damage Plane	Severity
QTR	Front	Moderate
MBAR	Back	Unknown
MUSTANG/MUSTANG II	Back	Unknown

	Max Injury	Max Severity	Injury Source
Head Unknown		Unknown	
Chest		Unknown	
Lower	Ulna fracture, shaft, complex comminuted, segmental	Moderate	Right instrument Panel

[illegible]

# LibreOffice Basic

gl\_test.ods - LibreOffice Calc

File Edit View Insert Format Styles Sheet Data Tools Window Help

Times New Roman 11 B I U A

Q44 fx Σ =

	A	B	C	D	E	F	G	H	I
1									
2									
3		<b>AGL – A General Ledger</b>							
4									
5									
6		Income transaction	<b>Add an income transaction</b>						
7			<i>(Transaction number and debit account are assigned automatically)</i>						
8									
9									
10		Expense transaction	<b>Add an expense transaction</b>						
11			<i>(Transaction number and credit account are assigned automatically)</i>						
12									
13									
14									
15		Other transaction	<b>Add a new transaction</b>						
16			<i>(Transaction number is assigned automatically)</i>						
17									
18									
19									
20		Income Statement and Balance Sheet	<b>Produce an Income Statement and Balance Sheet</b>						
21									
22									
23									
24									
25		BOD Report	<b>Produce Monthly BOD Report</b>						
26			<i>(Includes producing a balance sheet and income statement)</i>						
27									
28									
29									
30									

< Main Menu Data Entry Chart of Accounts General Ledger Cash Investment Account (Tangerine) Mem

Find Find All ☐ Formatted Display ☐ Match Case

Sheet 1 of 37 Default En

[illegible]



D8

 $f_x$   $\Sigma$  =

	A	B	C	D	E	F	G	H
1								
2		<b>Transaction Entry</b>						
3								
4								
5		<b>Transaction No.</b>		136				
6								
7		<b>Date</b>		30-Nov-2018				
8		<b>Credit Account</b>						
9		<b>Debit Account</b>		1000		Cash		
10		<b>Amount</b>						
11		<b>Description</b>						
12								
13								
14								
15								
16								
17				Save				
18								

gl\_test.ods.Standard - LibreOffice Basic

File Edit View Run Dialog Tools Window Help

[gl\_test.ods].Standard

### Object Catalog

- My Macros & Dialogs
  - Standard
    - Module1
- LibreOffice Macros & Dialogs
  - gl\_test.ods
    - Standard
      - AGL\_Macros**
        - IncomeTransaction
        - ExpenseTransaction
        - OtherTransaction
        - ReadTransactionData
        - GetAccountName
        - GetLastBalance
        - IncomeStatement
        - BalanceSheet
        - ConsistencyCheck
        - BODReport

```

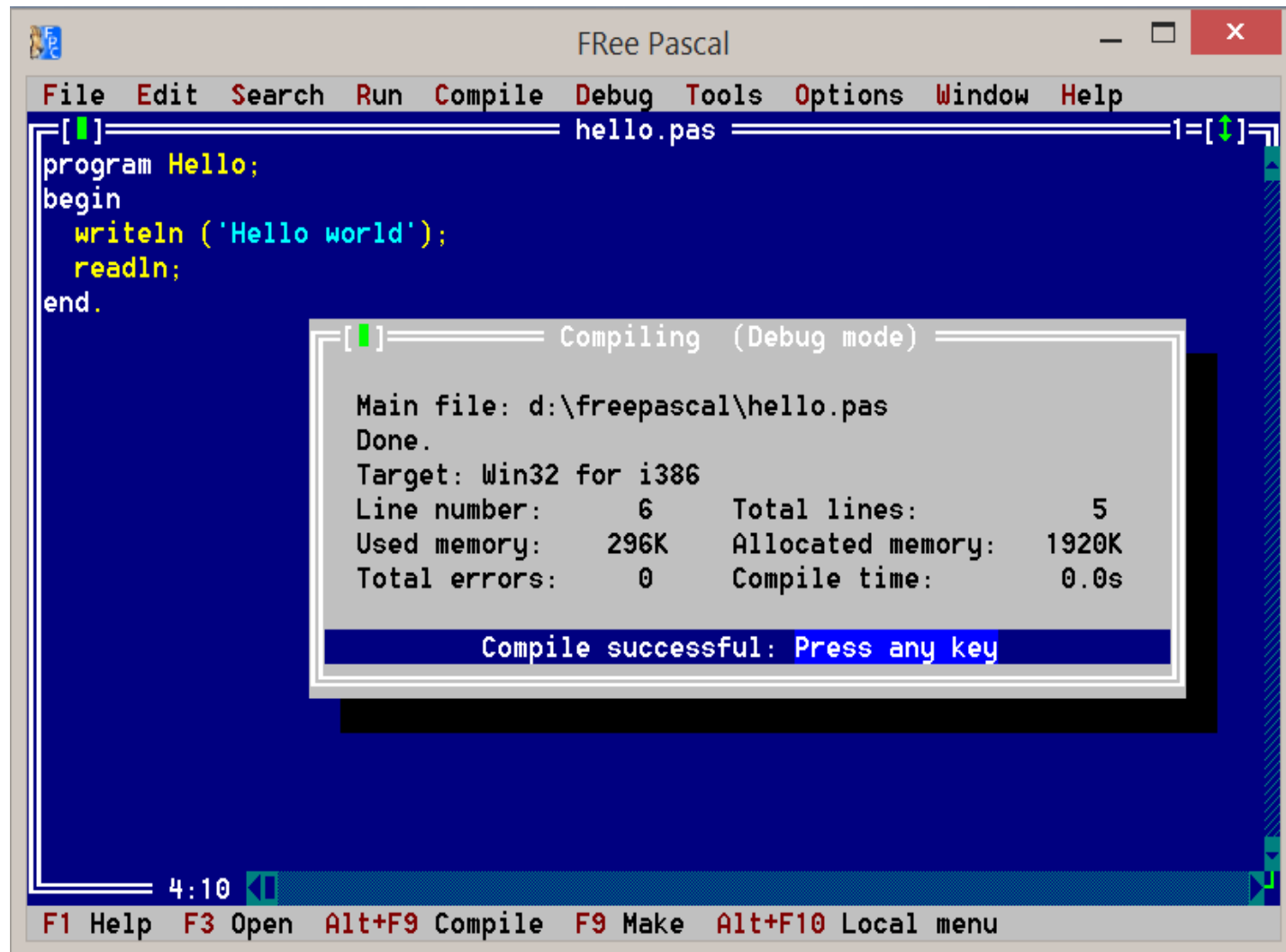
11 Sub IncomeTransaction
12
13 Dim cfgSheet As Object
14 Dim trans_number As Double
15 Dim last_date as Date
16 Dim dataSheet As Object
17
18 ' =====
19 ' Read configuration data
20 ' =====
21
22 cfgSheet = ThisComponent.Sheets.getByName("Configuration")
23 ThisComponent.CurrentController.setActiveSheet(cfgSheet)
24
25 ' Obtain last transaction number and index it by one
26
27 trans_number = cfgSheet.getCellRangeByName("B9").value
28 trans_number = trans_number + 1
29
30 ' Obtain the date of the last transaction
31
32 last_date = cfgSheet.getCellRangeByName("B13").value
33
34 ' Write the current transaction number, the date of the last transaction,
35 ' and 1000 (Cash) as the debit account to the Data Entry worksheet
36
37 dataSheet = ThisComponent.Sheets.getByName("Data Entry")
38 ThisComponent.CurrentController.setActiveSheet(dataSheet)

```

Watch:

Variable	Value	Type

# Free Pascal (FRP)



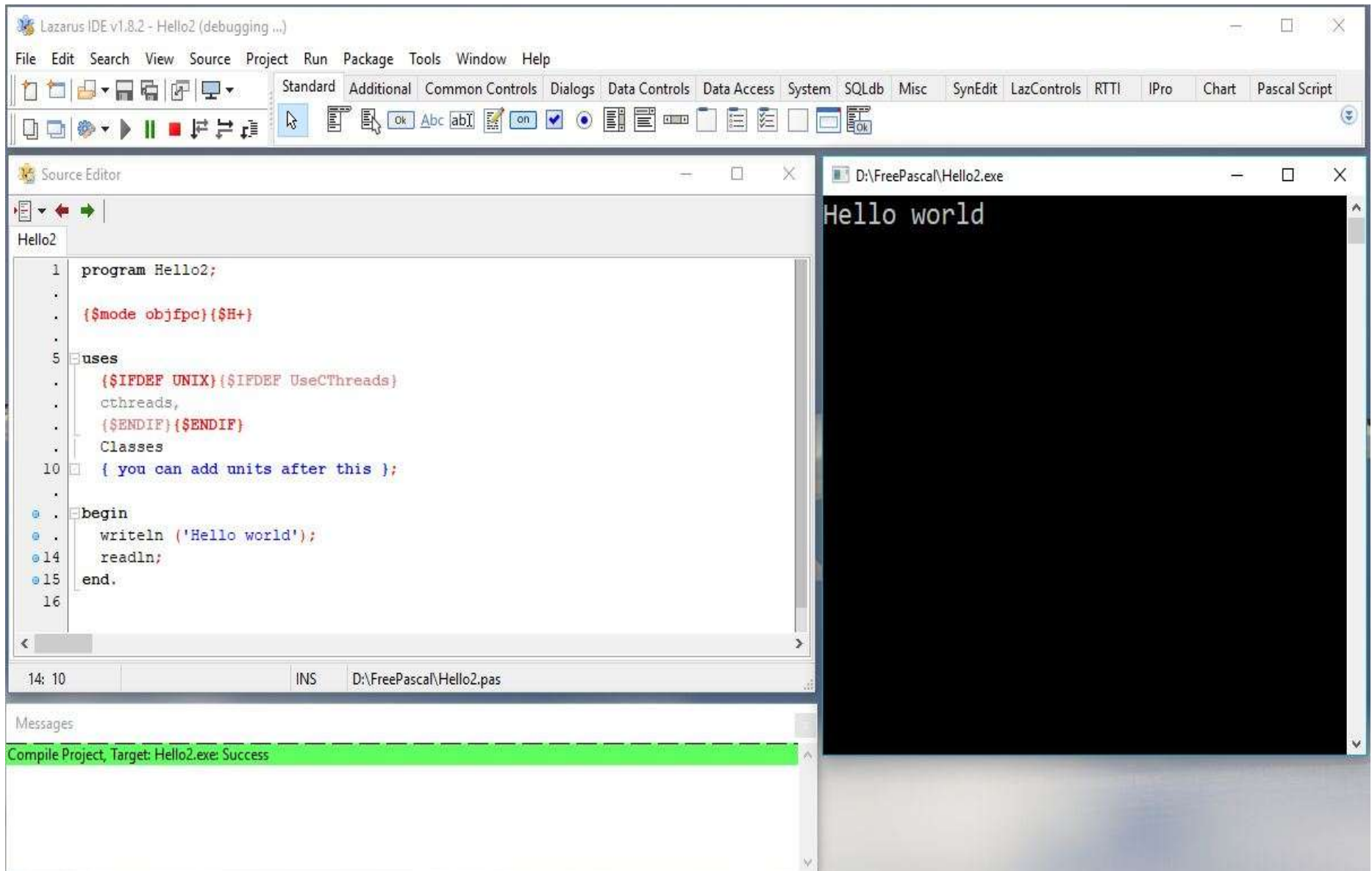
# Output in Command Window



The screenshot shows a window titled "Free Pascal" with a standard Windows title bar (minimize, maximize, close buttons). The window contains a black command window with white text. The text displays the Free Pascal IDE version (1.0.12), compiler version (3.0.4), and GDB version (7.4). It also shows the configuration files path and the execution of a program named "hello.exe" which outputs "Hello world".

```
Free Pascal  
■ Free Pascal IDE Version 1.0.12 [2017/10/06]  
■ Compiler Version 3.0.4  
■ GDB Version GDB 7.4  
■ Using configuration files from: C:\FPC\3.0.4\bin\i386-win32\  
Running "d:\freepascal\hello.exe "  
Hello world
```

# Lazarus - IDE



# Turbo Pascal vs. Free Pascal

```
{print a page of the file}
while (nlines<62) and (not EOF(Fil)) do
begin
  readln(Fil,line);
  writeln(Lst,line);
  nlines:=nlines+1;
end;
```

```
aLines.Free;
aLines := TStringList.Create;
aLines.Add(' ');
aLines.Add(' ');
aLines.Add('Filelist Version 2.0');
aLines.Add('Listing of file');
aLines.Add('_____');
aLines.Add(' ');
aLines.Add(' ');
```

**Page header**

```
bLines := TStringList.Create;
bLines.LoadFromFile('test.pas');
```

**Body text**

```
For J := FirstJ to LastJ do
begin
  aLines.Add(bLines[J-1]);
  I := I + 1;
  count := I;
end;
```

**Line counters**

```
PrintStrings(aLines);
```

**Print page!**



# C

- GNU C Compiler (GCC) is built into Linux Mint
- Create source code file in a text editor (e.g. hello.c)
- Compile the program  
`gcc hello.c -o hello`
- Run the program  
`./hello`

# hello.c - source code

```
#include <stdio.h>
```

```
int main(void)
```

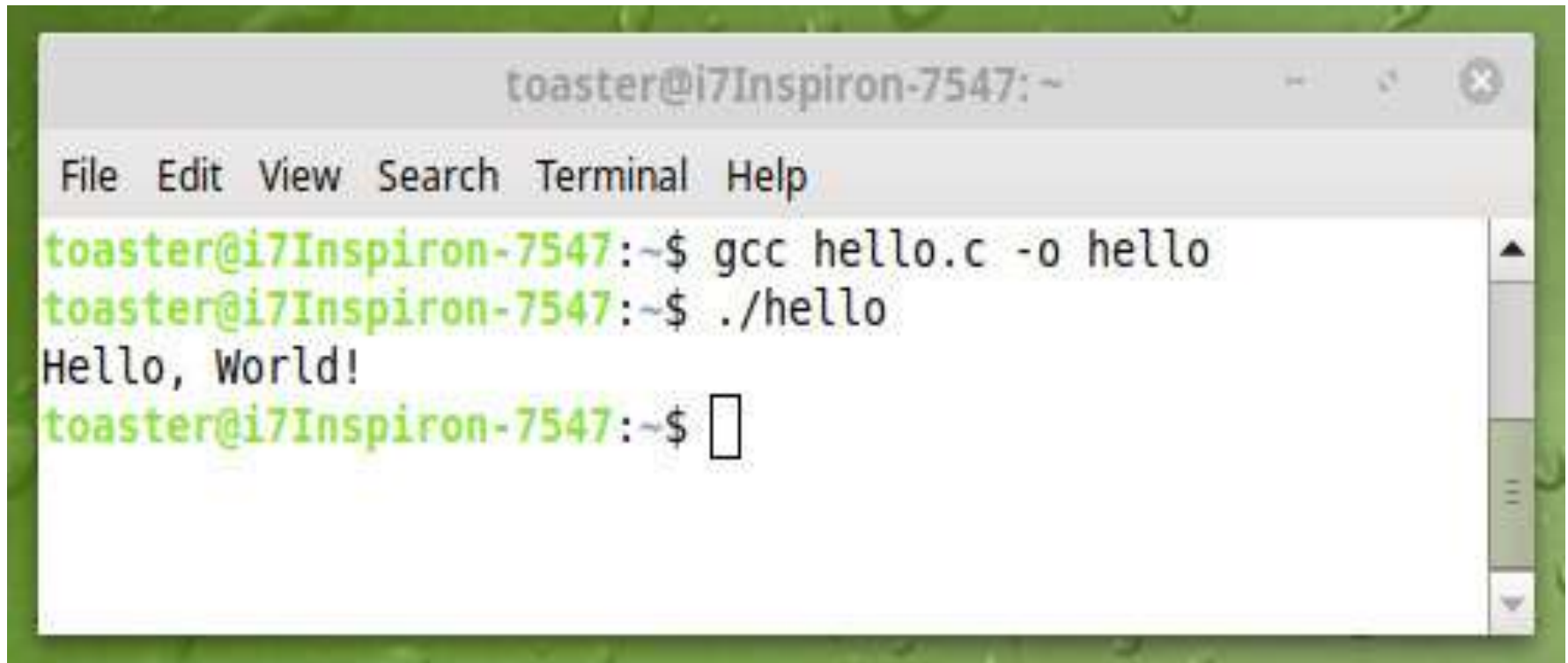
```
{
```

```
printf("Hello, World!\n");
```

```
return 0;
```

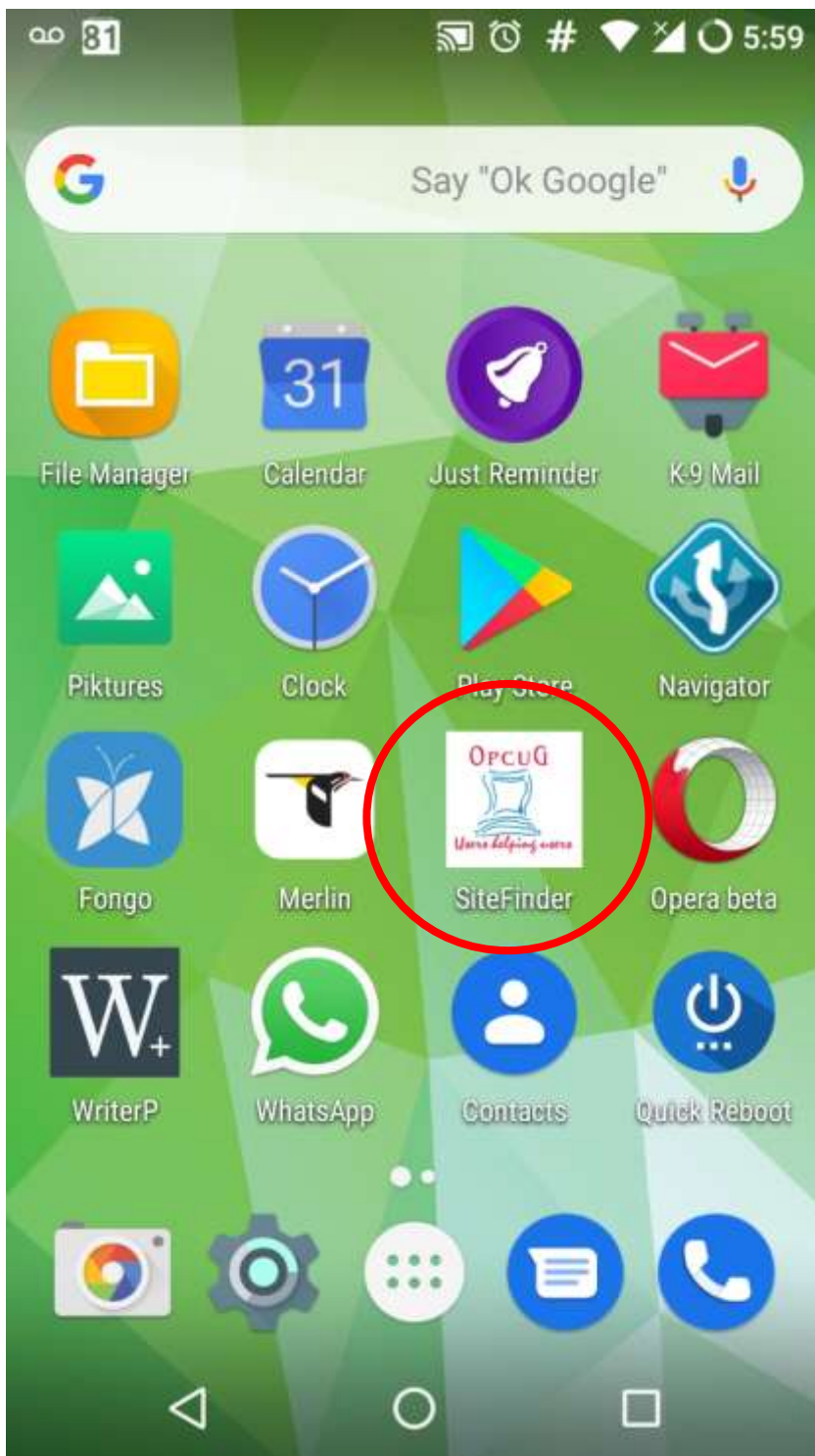
```
}
```

# Compile and Run

A screenshot of a terminal window titled 'toaster@i7Inspiron-7547: ~'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the following sequence of commands and output:

```
toaster@i7Inspiron-7547:~$ gcc hello.c -o hello
toaster@i7Inspiron-7547:~$ ./hello
Hello, World!
toaster@i7Inspiron-7547:~$
```

The prompt 'toaster@i7Inspiron-7547:~\$' is shown in green. The output 'Hello, World!' is shown in black. The cursor is at the end of the last command line.



# Site Finder

URL “wrapper”

All the app does is  
point to [opcug.ca](http://opcug.ca)  
and load the site  
in a browser



Load web site →



Back arrow →



# Android Cordova

- Apache Software Foundation
- Develop Android apps
- HTML, CSS  
and JavaScript
- Android, IOS  
and Windows Phone
- Node.js, npm, cordova,  
JDK, Android Studio (SDK)



## SiteFinder

File Edit View Go Bookmarks Help



/home/toaster/cordova/SiteFinder



## ▼ My Computer

- Home
- Deskt...
- Docu...
- Music
- Pictures
- Videos
- Downl...
- File Sy...
- Trash

## ▼ Devices

- DataD...
- Windo...

## ▼ Network

- Netwo...

Name

Size

Type

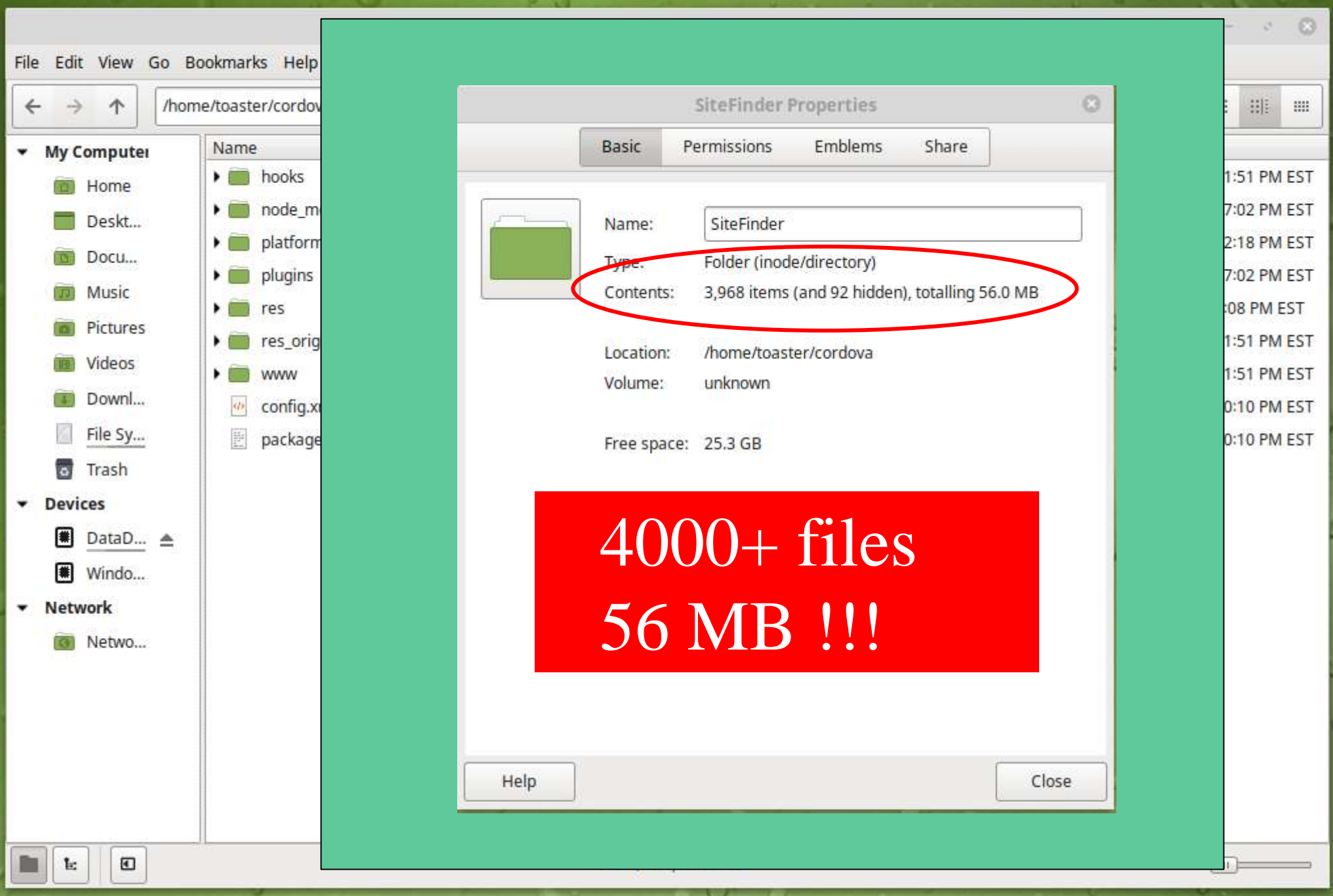
Date Modified

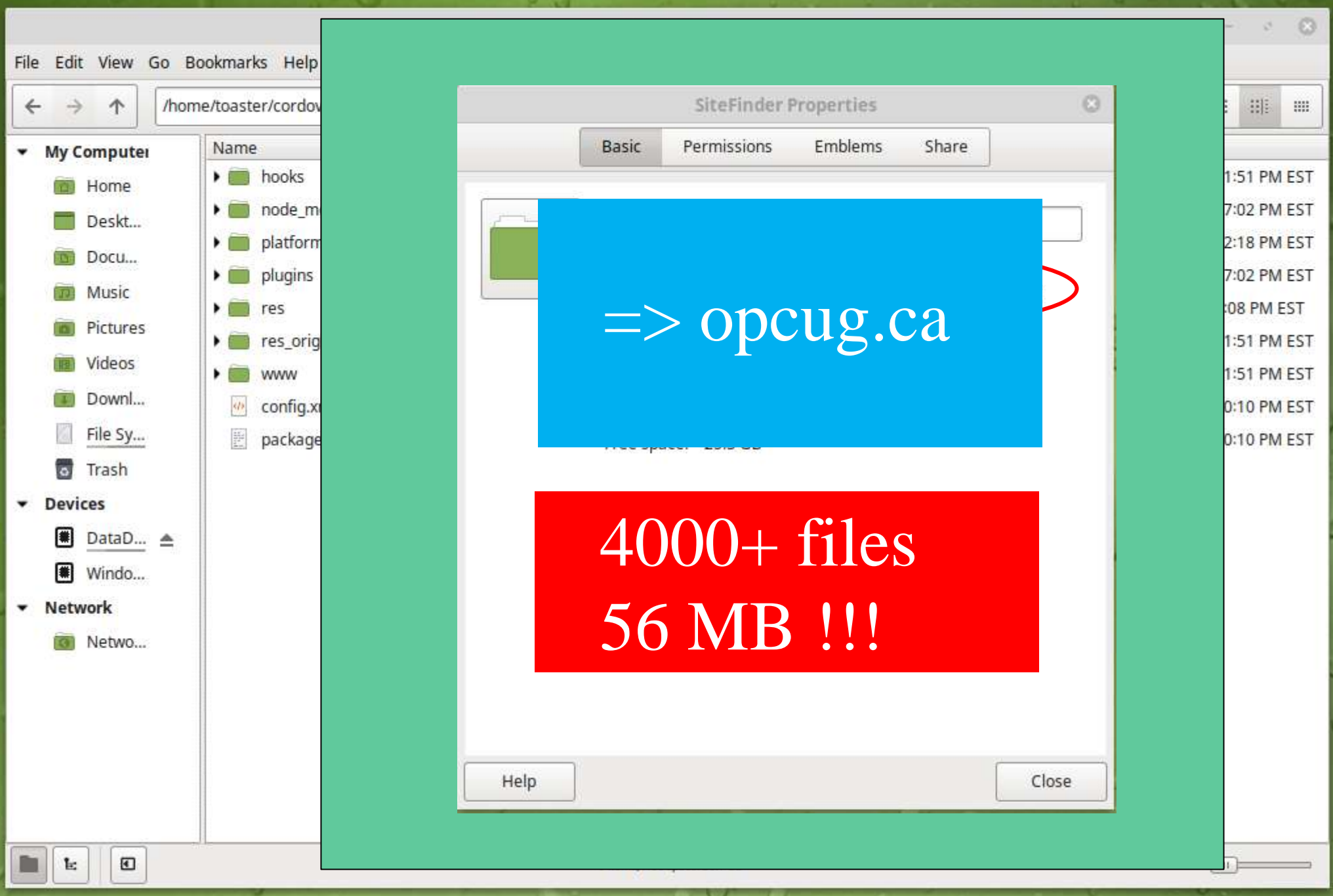
▶ hooks	1 item	Folder	Sun 25 Nov 2018 04:21:51 PM EST
▶ node_modules	3 items	Folder	Sun 25 Nov 2018 05:17:02 PM EST
▶ platforms	1 item	Folder	Sun 25 Nov 2018 04:22:18 PM EST
▶ plugins	4 items	Folder	Sun 25 Nov 2018 05:17:02 PM EST
▶ res	2 items	Folder	Fri 09 Nov 2018 01:46:08 PM EST
▶ res_original	3 items	Folder	Sun 25 Nov 2018 04:21:51 PM EST
▶ www	4 items	Folder	Sun 25 Nov 2018 04:21:51 PM EST
config.xml	2.7 kB	Markup	Sun 25 Nov 2018 05:20:10 PM EST
package.json	653 bytes	Program	Sun 25 Nov 2018 05:20:10 PM EST

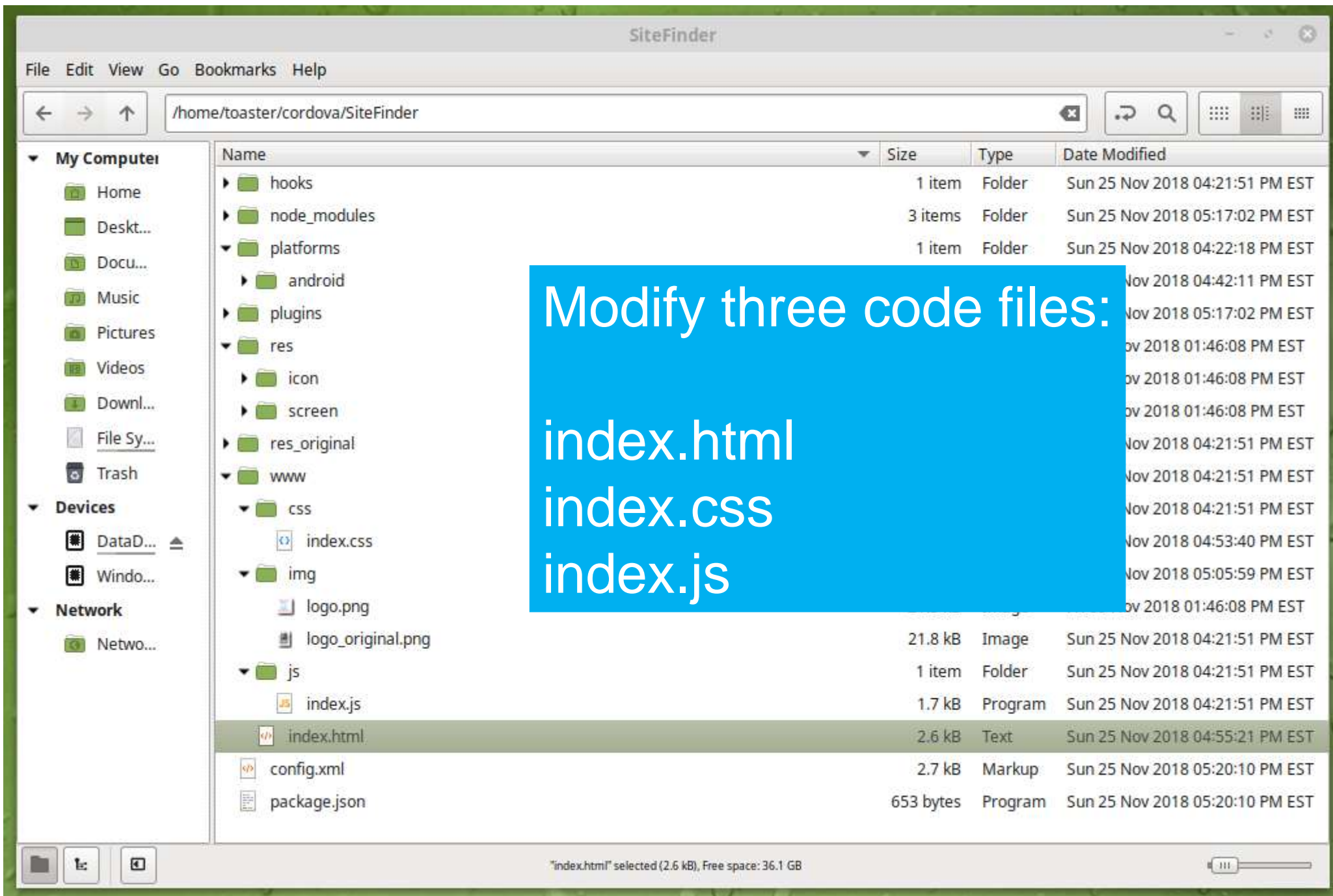
9 items, Free space: 36.1 GB









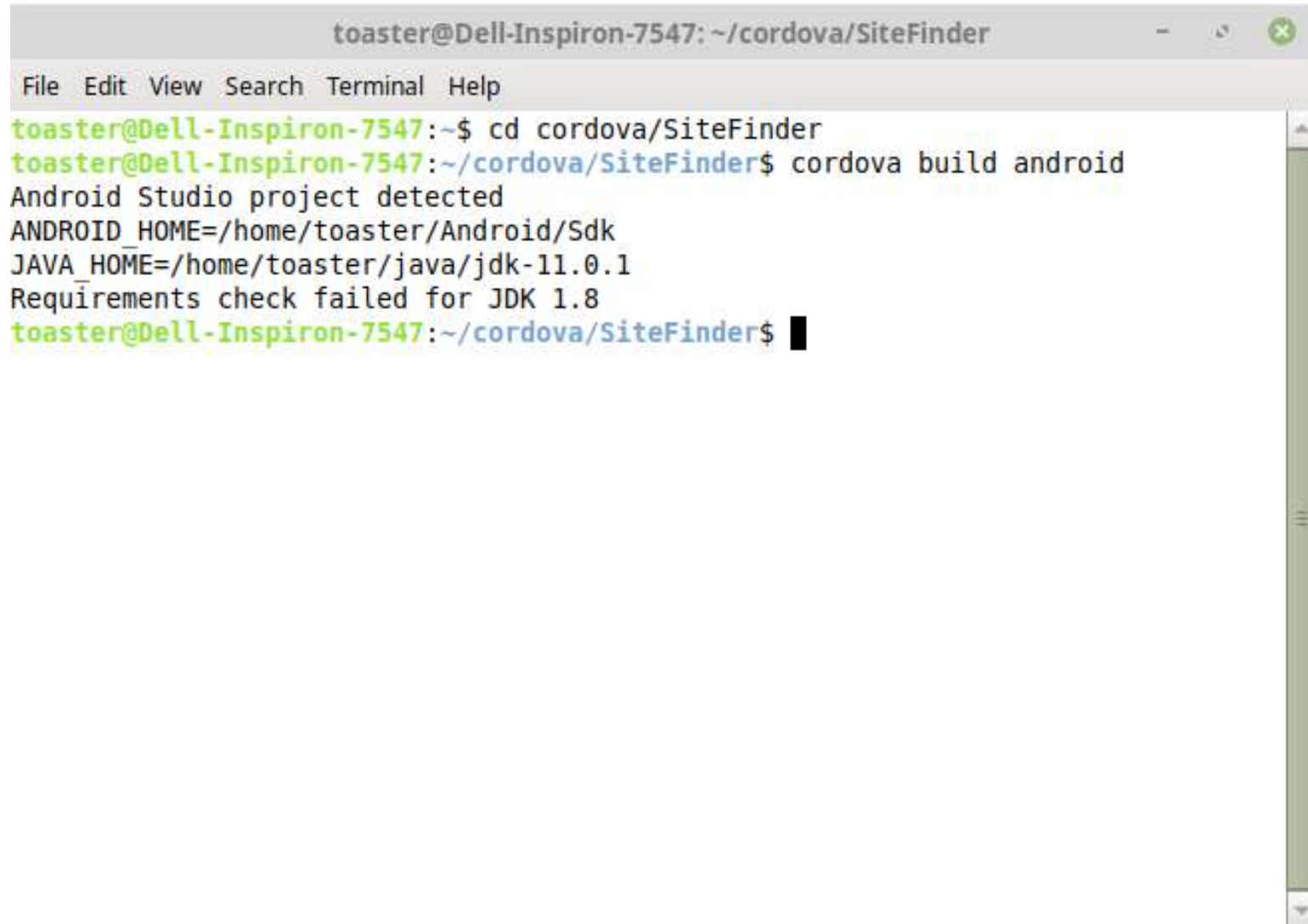






```
<body>
  <div class="app">
    <p>&nbsp;</p>
    <h2>Ottawa PC Users' Group</h2>
    <div id="deviceready">
      <p class="event listening">Connecting to Device</p>
      <p>
        <p class="event received"><a href="http://opcug.ca">
          <h1>http://opcug.ca</h1></a></p>
      </div>
    </div>
    <script type="text/javascript" src="cordova.js"></script>
    <script type="text/javascript" src="js/index.js"></script>
  </body>
```

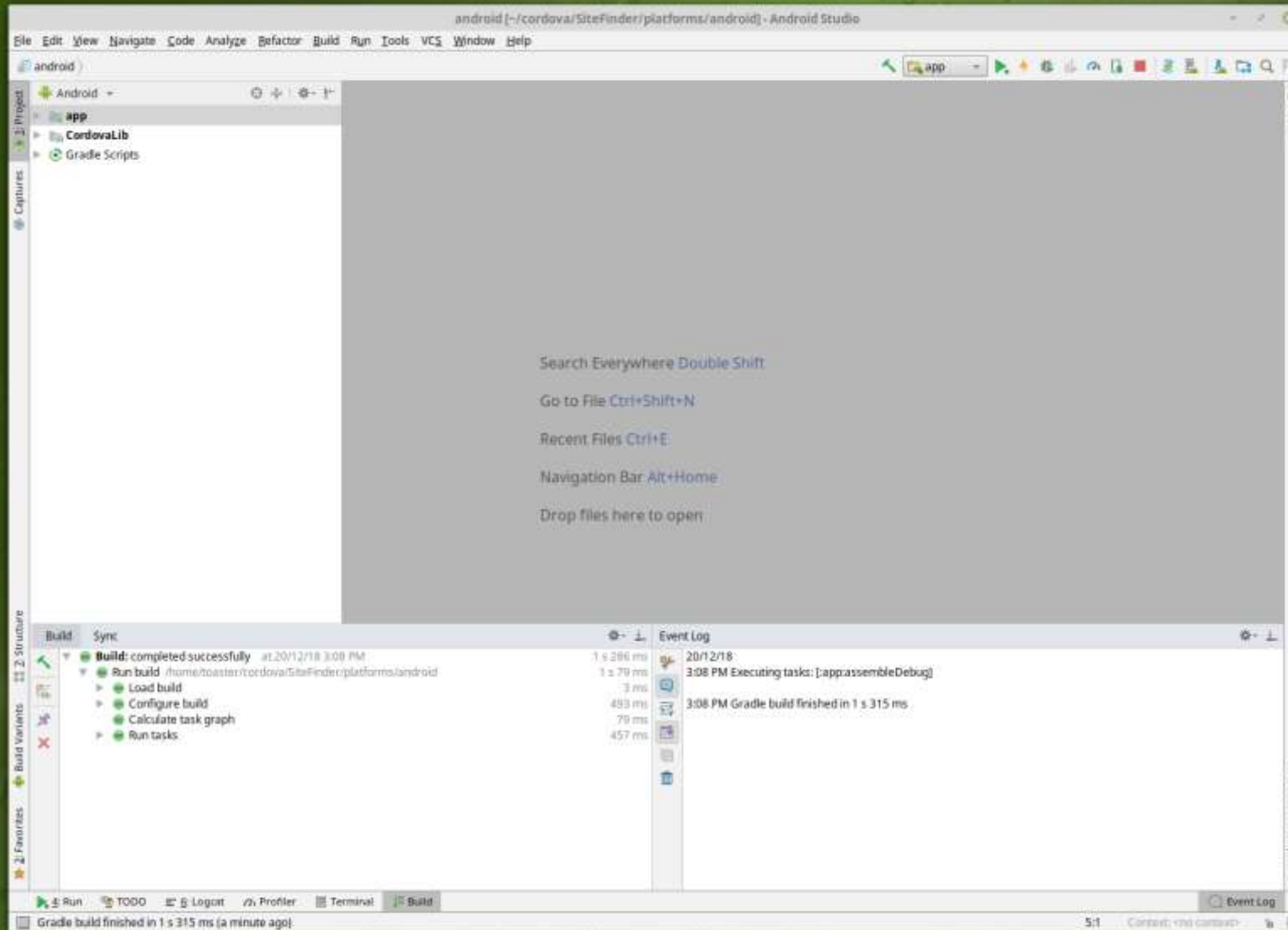
# Build 1



```
toaster@Dell-Inspiron-7547: ~/cordova/SiteFinder
File Edit View Search Terminal Help
toaster@Dell-Inspiron-7547:~$ cd cordova/SiteFinder
toaster@Dell-Inspiron-7547:~/cordova/SiteFinder$ cordova build android
Android Studio project detected
ANDROID_HOME=/home/toaster/Android/Sdk
JAVA_HOME=/home/toaster/java/jdk-11.0.1
Requirements check failed for JDK 1.8
toaster@Dell-Inspiron-7547:~/cordova/SiteFinder$
```

The image shows a terminal window titled "toaster@Dell-Inspiron-7547: ~/cordova/SiteFinder". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal content shows the user navigating to the "cordova/SiteFinder" directory and running the command "cordova build android". The output indicates that an Android Studio project was detected, and the system variables ANDROID\_HOME and JAVA\_HOME were set. However, a requirements check for JDK 1.8 failed. The prompt returns to the user's shell.

# Build 2







ANDROID DEVELOPMENT

NEWS

October 14, 2016

# How to build your own custom Android ROM



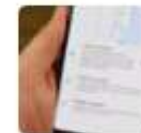
3.2K Shares



Gary Sims / @garyexplains

One of the key features of Android is that it is open source. The source code for the full operating system, including the kernel, UI, libraries and key apps, is available for free. This means that anyone (with the right technical skills) can build Android from source and flash it onto a compatible device. This flexibility has allowed various groups, some commercial and some hobbyist, to develop alternative distributions of Android. These are commonly referred to as “custom ROMs” however a better name would be “custom firmware.”

## YOU MIGHT LIKE

**Don't like /  
can custom**

by Phillip Prad

**Here's how  
buttonless**

by Gary Sims

# Prerequisites

- Know about makefile and shell commands
- Linux or Mac computer
- 130 GB of disk space; 8 GB+ of RAM
- 24 hours to synchronize the source repository with the local machine
- 10 to 20 minutes to build after a minor change
- Several hours for a clean build



# Bash Scripting (Linux)

- Problem – “Safely remove” a  
USB flash drive
- Red light on drive still flashing!
- Solution – Flush the drive buffer  
Unmount the drive  
Power down the drive
- Use a bash script (`#!/bin/bash`)

# Device ID

```
usblongname =  
$(lsblk -l | grep SILICON16GB)
```

```
usbname = "${usblongname:0:4}"
```

```
Device Name = /dev/sdg1  
usbname = sdg1
```



EjectUSB.sh

```
1  #!/bin/bash
2
3  # Eject USB drive once buffer transfer has completed
4
5  echo "Flushing USB drive buffer"
6  sync
7
8  echo "sync complete"
9
10 # Identify the device name for the SILICON16GB USB drive
11
12 usblongname=$(lsblk -l | grep SILICON16GB)
13 usbname="${usblongname:0:4}"
14
15 # Unmount the USB drive
16
17 udiskctl unmount -b /dev/$usbname
18
19 echo "unmount complete"
20
21 # Power off the USB drive
22
23 udiskctl power-off -b /dev/$usbname
24
25 echo "power off complete"
26
27 # Script complete
28
29 echo "Shell command complete"
30 read
31
32
```

EjectUSB.sh

```
1  #!/bin/bash
2
3  # Eject USB drive once buffer transfer has completed
4
5  echo "Flushing USB drive buffer"
6  sync
7
8  echo "sync complete"
9
10 # Identify the device name for the SILICON16GB USB drive
11
12 usblongname=$(lsblk -l | grep SILICON16GB)
13 usbname="${usblongname:0:4}"
14
15 # Unmount the USB drive
16
17 udiskctl unmount -b /dev/$usbname
18
19 echo "unmount complete"
20
21 # Power off the USB drive
22
23 udiskctl power-off -b /dev/$usbname
24
25 echo "power off complete"
26
27 # Script complete
28
29 echo "Shell command complete"
30 read
31
32
```



sync



```
1  #! /bin/bash
2
3  # Eject USB drive once buffer transfer has completed
4
5  echo "Flushing USB drive buffer"
6  sync
7
8  echo "sync complete"
9
10 # Identify the device name for the SILICON16GB USB drive
11
12 usblongname=$(lsblk -l | grep SILICON16GB)
13 usbname="${usblongname:0:4}"
14
15 # Unmount the USB drive
16
17 udiskctl unmount -b /dev/$usbname
18
19 echo "unmount complete"
20
21 # Power off the USB drive
22
23 udiskctl power-off -b /dev/$usbname
24
25 echo "power off complete"
26
27 # Script complete
28
29 echo "Shell command complete"
30 read
31
32
```

usbname = sdg1

EjectUSB.sh

```
1  #!/bin/bash
2
3  # Eject USB drive once buffer transfer has completed
4
5  echo "Flushing USB drive buffer"
6  sync
7
8  echo "sync complete"
9
10 # Identify the device name for the SILICON16GB USB drive
11
12 usblongname=$(lsblk -l | grep SILICON16GB)
13 usename="${usblongname:0:4}"
14
15 # Unmount the USB drive
16
17 udisksctl unmount -b /dev/$usename
18
19 echo "unmount complete"
20
21 # Power off the USB drive
22
23 udisksctl power-off -b /dev/$usename
24
25 echo "power off complete"
26
27 # Script complete
28
29 echo "Shell command complete"
30 read
31
32
```

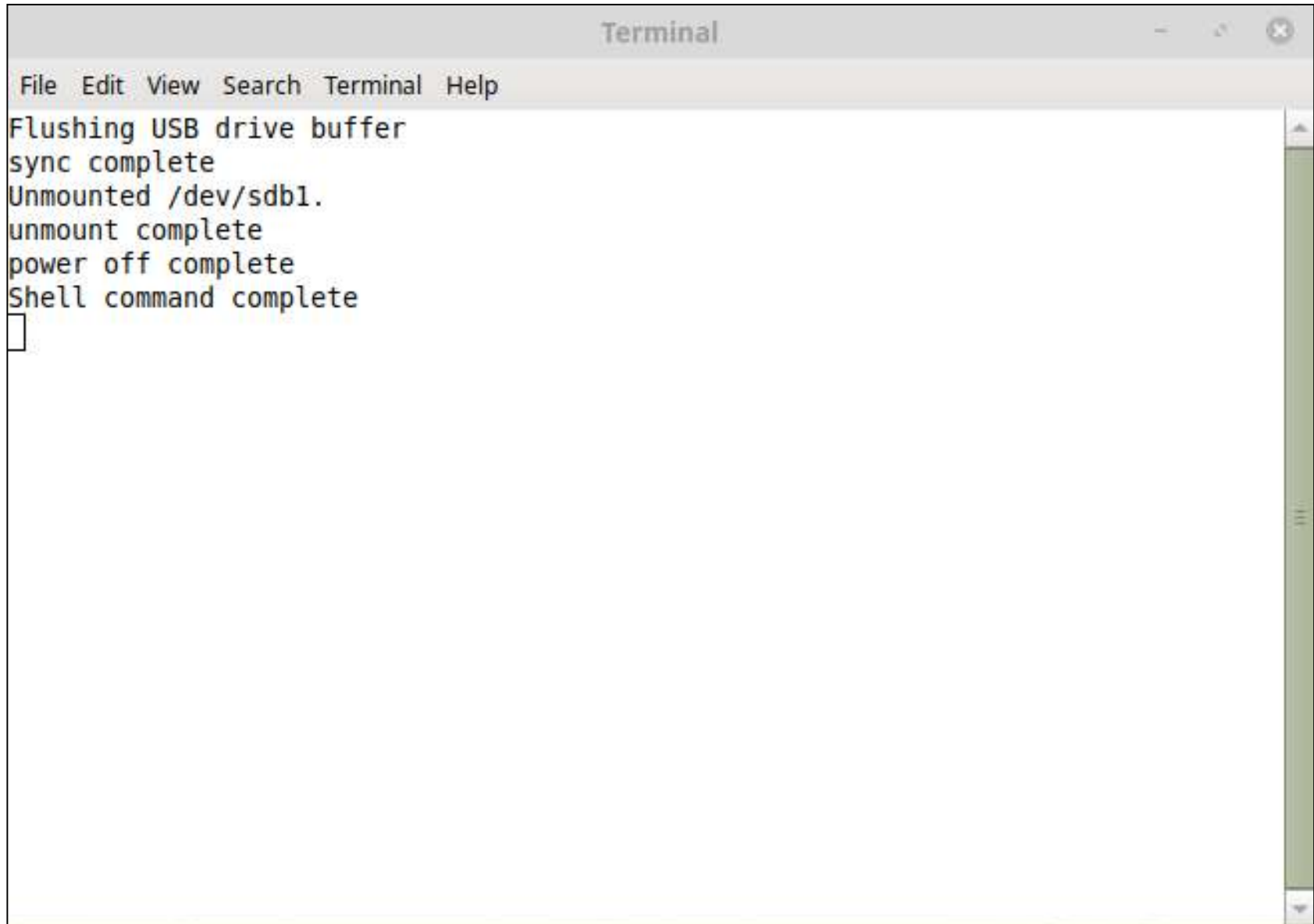
udisksctl  
unmount -b  
/dev/\$usename

EjectUSB.sh

```
1  #!/bin/bash
2
3  # Eject USB drive once buffer transfer has completed
4
5  echo "Flushing USB drive buffer"
6  sync
7
8  echo "sync complete"
9
10 # Identify the device name for the SILICON16GB USB drive
11
12 usblongname=$(lsblk -l | grep SILICON16GB)
13 usbname="${usblongname:0:4}"
14
15 # Unmount the USB drive
16
17 udiskctl unmount -b /dev/$usbname
18
19 echo "unmount complete"
20
21 # Power off the USB drive
22
23 udiskctl power-off -b /dev/$usbname
24
25 echo "power off complete"
26
27 # Script complete
28
29 echo "Shell command complete"
30 read
31
32
```

udiskctl  
power-off -b  
/dev/\$usbname

# Success!

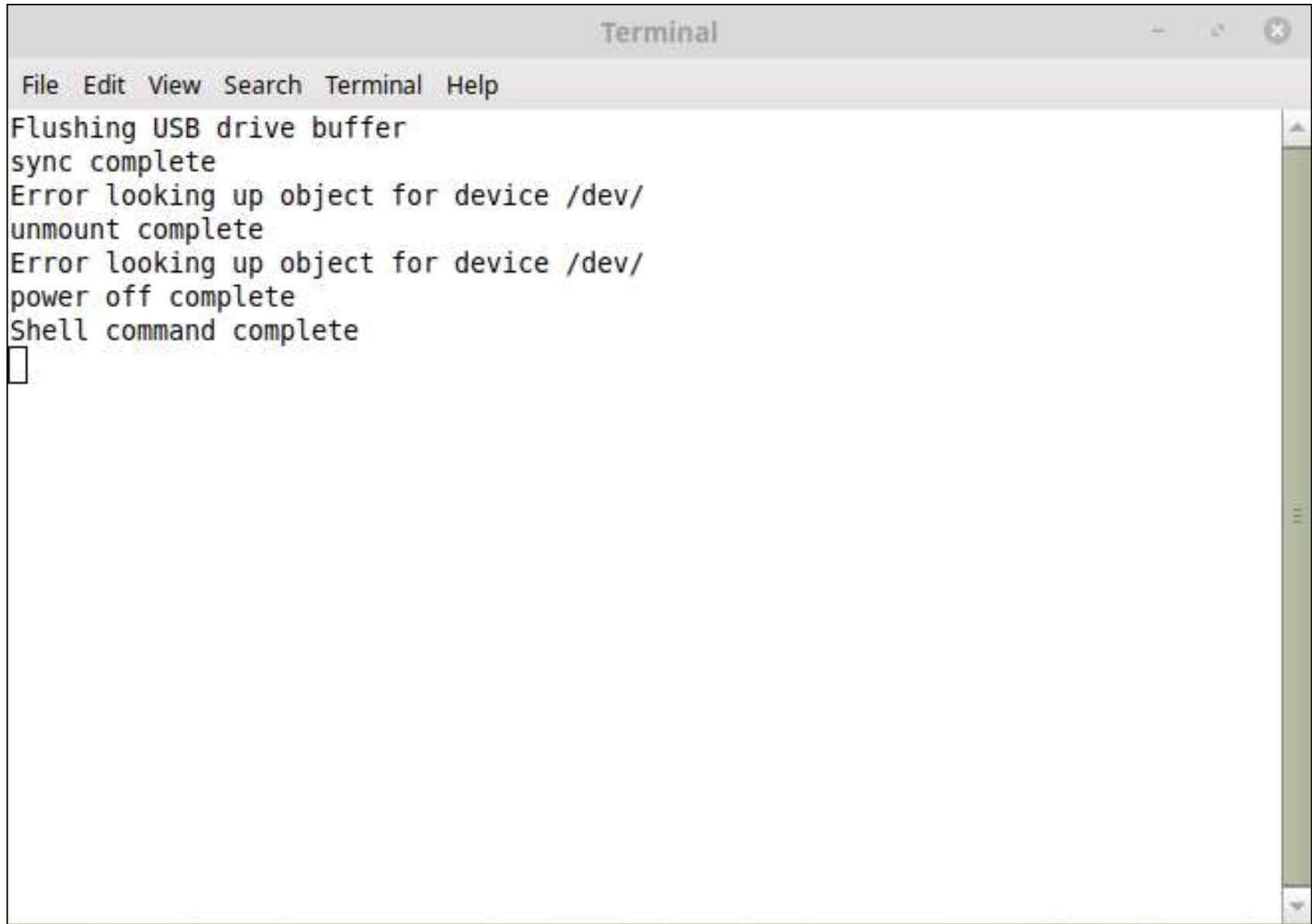


A screenshot of a macOS Terminal window titled "Terminal". The window has a standard macOS title bar with minimize, maximize, and close buttons. Below the title bar is a menu bar with the following items: File, Edit, View, Search, Terminal, and Help. The main content area of the terminal displays the following text in a monospaced font:

```
Flushing USB drive buffer  
sync complete  
Unmounted /dev/sdb1.  
unmount complete  
power off complete  
Shell command complete  
█
```

The text indicates a successful sequence of operations: flushing the USB drive buffer, completing the sync, unmounting the device `/dev/sdb1`, completing the unmount, powering off the device, and finally completing the shell command. A cursor (represented by a small square) is visible on the line following the last message.

# Fail Safe



```
Terminal
File Edit View Search Terminal Help
Flushing USB drive buffer
sync complete
Error looking up object for device /dev/
unmount complete
Error looking up object for device /dev/
power off complete
Shell command complete
█
```

The image shows a terminal window titled "Terminal" with a menu bar containing "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal output displays the following sequence of messages: "Flushing USB drive buffer", "sync complete", "Error looking up object for device /dev/", "unmount complete", "Error looking up object for device /dev/", "power off complete", and "Shell command complete". A cursor, represented by a small square, is positioned on the line following the last message.

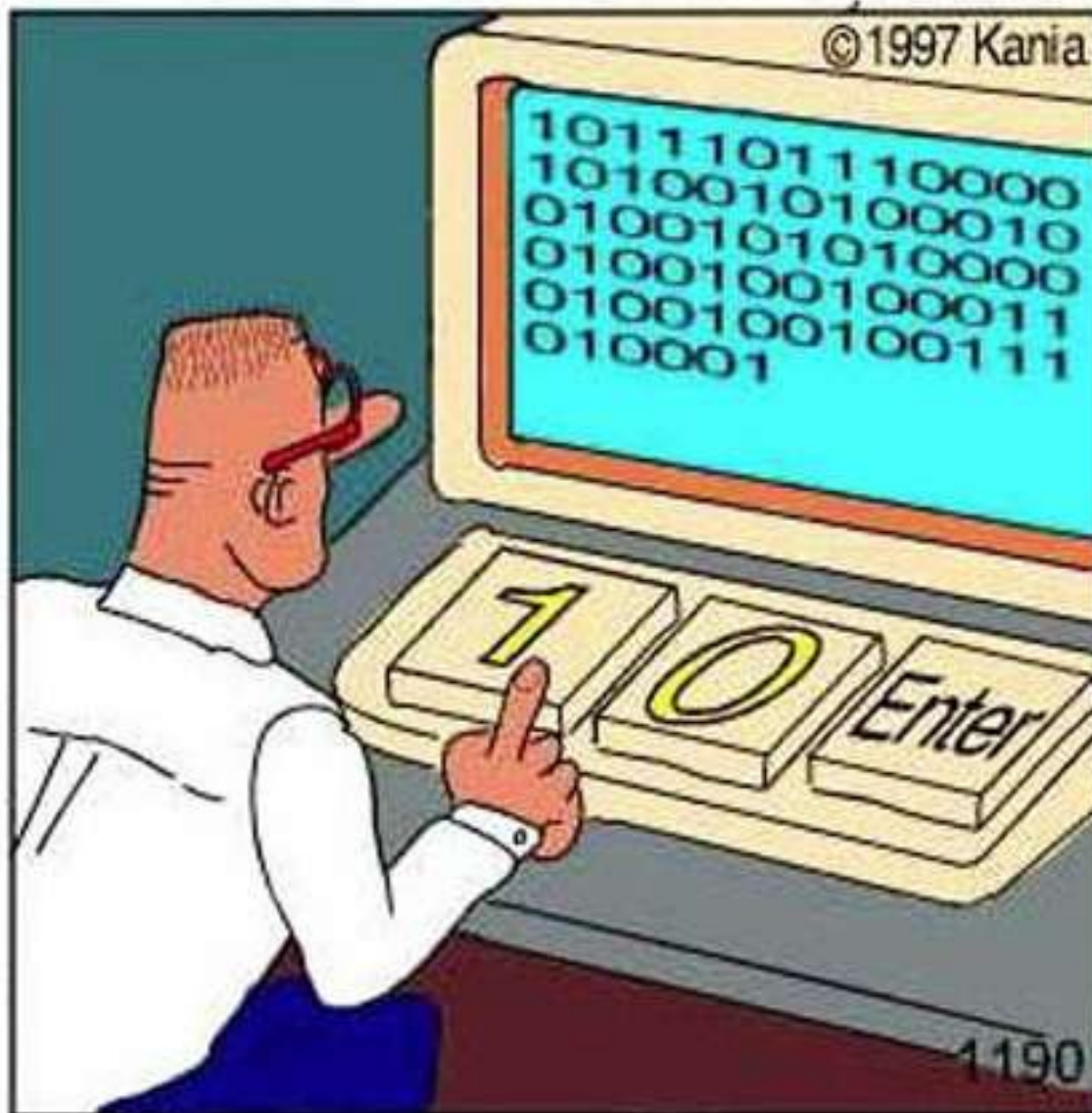
# So, you want to be a programmer...





**If at first you don't succeed,  
you must be a programmer...**





Real programmers code in binary.

I used to want to be a computer programmer when I was younger. We got an Apple II Plus when I was, like, 11 and I wrote programs and BASIC on that, like I think a lot of people did, but I have no idea how to program in the current languages at all.

Chris Parnell



**Any Questions?**