```
%
%
%
% ctrl-D to clear any hanging input
%
% dBASE IV Postscript Download File (Postscri.dld)
% Copyright 1989 Ashton-Tate
% version 1.12
% --------- User-adjustable parameters -------
% default page size & orientation
/paper 1 def     % (1=letter, 2=legal)
/orient 1 def    % (1=portrait, 2=landscape)
/nLPP 66 def     % lines per page (60, 66, or 78 for portrait; 45 for
landscape)
% printer-dependent portrait offsets
/tpxoff 18 def   % x (letter)
/tpyoff 28 def   % y (letter)
/gpxoff 22 def   % x (legal)
/gpyoff 26 def   % y (legal)
% printer-dependent landscape offsets
/tlxoff 10 def   % x (letter)
/tlyoff 44 def   % y (letter)
/glxoff 14 def   % x (legal)
/glyoff 80 def   % y (legal)
% point sizes to use
/PicaPoint 12 def                % Pica
/ElitePoint 10 def                    % Elite
/CompressedPoint 8 def           % compressed
/CurPoint        PicaPoint def   % initial pointsize
% fonts to use
/CurFSet 1 def   % default font set
/FSet {
      /CurFSet exch def
      % Font Set 1 (default)
      /n    /Courier def                    % normal
      /b    /Courier-Bold def           % bold
      /i    /Courier-Oblique def        % italic
      /bi   /Courier-BoldOblique def    % bold+italic
      CurFSet 2 eq
            {
            /n    /Helvetica def
            /b    /Helvetica-Bold def
            /i    /Helvetica-Oblique def
            /bi   /Helvetica-BoldOblique def
            } if
      CurFSet 3 eq
            {
            /n    /Times-Roman def
            /b    /Times-Bold def
            /i    /Times-Italic def
            /bi   /Times-BoldItalic def
            } if
      /FontNorm n def
```

```
      /FontBold b def
      /FontItal i def
      /FontBoldItal bi def
      Norm
   } def
% --------- End User-adjustable parameters -------
% ------ User-callable macros
/Letter {/paper 1 def OrientPaper} def  % Set Letter
/Legal     {/paper 2 def OrientPaper} def    % Set Legal
/Port {/orient 1 def OrientPaper} def   % Set Portrait
/Land {/orient 2 def OrientPaper} def   % Set Landscape
/60LPP     {60 LPP} def
/66LPP     {66 LPP} def
/78LPP     {78 LPP} def
/1Font     {1 FSet} def
/2Font     {2 FSet} def
/3Font     {3 FSet} def
% Synonyms
/LETTER {Letter} def /letter {Letter} def
/LEGAL {Legal} def /legal {Legal} def
/PORT {Port} def /port {Port} def
/LAND {Land} def /land {Land} def
/60lpp {60LPP} def
/66lpp {66LPP} def
/78lpp {78LPP} def
/1FONT {1Font} def /1font {1Font} def
/2FONT {2Font} def /2font {2Font} def
/3FONT {3Font} def /3font {3Font} def
% ------ End
%
% Page Setup
/PSet {
      initmatrix
      CurFSet FSet
      /CPI 10 def
      /LPI 6 def
      /HMI 72 def
      /VMI 72 def
      paper 1 eq
            {orient 1 eq     % letter
                  {/w 8.5 def /h 10.0 def           % letter port
                  nLPP 60 eq {/h 10.0 def} if
                  nLPP 66 eq {/h 10.1 def /LPI 6.6 def} if
                  nLPP 78 eq {/h 13.0 def} if
                  }
                  {/w 11.0 def /h 7.5 def           % letter land
                  /nLPP 45 def
                  } ifelse
            }
            {orient 1 eq     % legal
                  {/w 8.5 def /h 13.1 def           % legal port
                  /nLPP 78 def
                  }
                  {/w 14.0 def /h 6.5 def           % legal land
```

```
                /nLPP 45 def
                } ifelse
            }
            ifelse
      /PageWidth w def
      /PageHeight h def
   } def

% initialize variables
/BaseLine 0 def
/Bon 0 def
/Ion 0 def
%----
%
% Define new BaseLine value
/DefBase {  % args = (new baseline)
   /BaseLine exch def
   } def
%
% Move to a new line
/GoNewLine {     % args = (x,y)
   dup DefBase moveto
   currentpoint exch pop 0 lt {FF} if
   } def
%
% Rotate and/or Translate (if nec.)
/RoTran {
      orient 1 eq
            {paper 1 eq
            {tpxoff tpyoff translate}
            {gpxoff gpyoff translate}
             ifelse
            }
            {-90 rotate
            paper 1 eq
                  {11.0}
                  {14.0}
                  ifelse
            VMI mul neg 0 translate

            paper 1 eq  % Translation for LaserWriter Legal-size limits
                  {tlxoff tlyoff translate}
                  {glxoff glyoff translate}
                  ifelse
            /LPI 6 def}
            ifelse
      } def
%
%     Move to top left corner of new page
/Home {
   RoTran
   clear
   0
   PageHeight VMI mul
```

```
      GoNewLine CRLF
   } def
%
% Set nLPP
/LPP {     % arg = (#lines per page)
   /nLPP exch def
   PSet
   Home
   } def
% Chg Orientation or Paper
%
/OrientPaper {
      PSet
      Home
      } def
%
%    Initialization
/Init {
   1.415 setmiterlimit
   Home
   Norm
   } def
%
% FormFeed
/FF {
   showpage
   Home
   } def
%
% Carriage Return w/o LF
/CR {
   currentpoint
   exch    pop
   0
   exch moveto
   } def
%
% Carriage Return w/ Line Feed
/CRLF {
   CR currentpoint
   VMI LPI div
   sub round
   GoNewLine
   } def
%
% Backspace
/BS {
   currentpoint
   exch                % bring x-pos to top
   ( ) stringwidth     % get current font x,y size
   pop                 % get rid of y-size
   sub                 % backup 1 char's worth
   exch moveto
   } def
```

```
%
% Establish a different font
/NewFont {
   CurFont findfont
   CurPoint scalefont
   setfont
   } def
%
% Start compressed (15 cpi) print
/Cmp+ {/CurPoint CompressedPoint def NewFont} def
%
% End compressed print
/Cmp- {/CurPoint PicaPoint def NewFont} def
%
% Start Pica pitch
/10Cpi {/CurPoint PicaPoint def NewFont} def
%
% Start Elite pitch
/12Cpi {/CurPoint ElitePoint def NewFont} def
%
% Change to a different font (same size): aFont ChgFont
/ChgFont {
   /CurFont exch def
   CurFont findfont
   CurPoint scalefont
   setfont
   } def
%
% Select un-attributed text
/Norm {
   FontNorm ChgFont
   /Bon 0 def
   /Ion 0 def
   } def
%
% Select bolded text
/Bold {
   Ion 0 eq
      {FontBold ChgFont}
      {FontBoldItal ChgFont}
      ifelse
   /Bon 1 def
   } def
%
% Select un-Bolded text
/BNorm {
   Ion 0 eq
      {FontNorm ChgFont}
      {FontItal ChgFont}
      ifelse
   /Bon 0 def
   } def
%
% Select italicized text
```

```
/Ital {
   Bon 0 eq
      {FontItal ChgFont}
      {FontBoldItal ChgFont}
      ifelse
   /Ion 1 def
   } def
%
% Select un-Italicized text
/INorm {
   Bon 0 eq
      {FontNorm ChgFont}
      {FontBold ChgFont}
      ifelse
   /Ion 0 def
   } def
%
% Select bold+italicized text
/BoIt {
   FontBoldItal ChgFont
   /Bon 1 def
   /Ion 1 def
   } def
%
% 1. calc a positive line movement of 1/2 the current point size
% 2. move to the current baseline position
/Script {
   VMI CurPoint idiv 2 idiv
   currentpoint pop BaseLine moveto
   } def
%
% Start superscript
/Sup+ {0 Script rmoveto} def
%
% End superscript
/Sup- {Script pop} def
%
% Start subscript
/Sub+ {0 Script neg rmoveto} def
%
% End subscript
/Sub- {Script pop} def
%
% Start underlined text
/Und+ {
   currentpoint pop BaseLine % estab start of underline position
   2 sub               % Y-pos = 2 points down from baseline
   } def
%
% End underlined text
/Und- {
   currentpoint  % save current location
   Und+ moveto   % move to ending underline position
   4 2 roll      % bring starting underline position to top of stack
```

```
   lineto        % draw the line
   stroke
   moveto        % return to original location
   } def
%
% Overstrike 1st char w/ 2nd char
/OV {       % string OV
   /str 2 1 roll def
   currentpoint
   str 0 1 getinterval show
   moveto
   str 1 1 getinterval show
   } def
%
PSet
%----- end of Postscri.dld ----
```