

```

***** D I R P 1 . P A S *****
*-----*
* Task      : Displays all files in any directory on the
*            screen, including subdirectories and volume
*            label names. File handling is performed by a
*            direct call to DOS functions 4EH and 4FH.
*            See also DIRP2.PAS.
*-----*
* Author    : Michael Tischer
* Developed on : 07/08/88
* Last update : 01/22/92
*****

program DIRP1;

Uses Crt, Dos; { Add CRT and DOS units }

{-- Type declarations -----}

type DirBufType = record { Data structures of functions 4EH and 4FH }
    Reserved      : array [1..21] of char;
    Attr          : byte;
    Time          : integer;
    Date          : integer;
    Size          : longint;
    Name          : array [1..13] of char;
end;

MonVec    = array[1..12] of string[3]; { Array with month names }

{-- Constants -----}

const FA_ReadOnly    = $01; { File attributes }
      FAHidden      = $02;
      FA_SysFile    = $04;
      FA_VolumeID   = $08;
      FA_Directory  = $10;
      FA_Archive    = $20;
      FA_AnyFile    = $3F;

      FENTS = 14; { Number of visible entries at a time }
      Months : MonVec = ( 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
                          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec' );

{*****}
{ * FindFirst: Finds first directory entry. }
{ * Input   : None }
{ * Output  : TRUE or FALSE, depending on whether entry is found }
{*****}

function FindFirst(SFileName : string; { File to be searched for }
                  Attribute : integer) : boolean; { Srch. attribute }

var Regs : Registers; { Processor registers for interrupt call }

begin
    SFileName := SFileName + #0; { Filename ended with null }
    Regs.ah := $4E; { Function number: Search for first }
    Regs.cx := Attribute; { Attribute to be sought }
    Regs.ds := Seg(SFileName); { Segment addresses of filenames }
    Regs.dx := succ(Ofs(SFileName)); { Offset addresses of filenames }
    MsDos( Regs ); { Call DOS interrupt 21H }
    FindFirst := ( (Regs.flags and 1) = 0 ) { Test carry flag }
end;

{*****}
{ * FindNext: Finds the next directory entry. }
{ * Input   : None }
{ * Output  : TRUE if the entry is found, otherwise FALSE }
{ * Info    : This function should be called after the FindFirst }
{ *          function is successfully called. }
{*****}

function FindNext : boolean;

var Regs : Registers; { Processor registers for interrupt call }

begin
    Regs.ah := $4F; { Function number: Search for next }
    MsDos( Regs ); { Call DOS interrupt 21H }
    FindNext := ( (Regs.flags and 1) = 0 ) { Test carry flag }
end;

{*****}
{ * SetDTA: Sets the DTA address. }
{*****}

```

[illegible]

```

{*****}
{ * Dir: Controls directory reading and output. * }
{ * Input   : SPath       = Search path with file pattern * }
{ *         ATTRIBUTE = Search attribute * }
{ * Output  : None * }
{*****}

procedure Dir( SPath : string; Attr : byte );

var NumOfEntries,           { Total number of entries found }
    NumInScrn      : integer; { Number of entries per screen }
    WtKey          : char;    { Wait for a keypress }
    DirBuf         : DirBufTyp; { Indicates a directory entry }

begin
    SetDTA(Seg(DirBuf), Ofs(DirBuf)); { DirBuf is the new DTA }
    clrscr; { Clear screen }
    ScreenDesign; { Prepare screen for directory output }

    NumInScrn := -1; { No more entries to display }
    NumOfEntries := 0; { No more entries found }
    if FindFirst( SPath, Attr ) then { Search for first entry }
    repeat
        NumOfEntries := succ(NumOfEntries); { One more entry found }
        NumInScrn := succ(NumInScrn); { One more entry in window }
        if NumInScrn = FENTS then { Is the window full? }
        begin
            Window( 14, (20-FENTS) shr 1 + 5+ FENTS,
                    66, (20-FENTS) shr 1 + 6+ FENTS );
            GotoXY(1, 1); { Move cursor to bottom line of window }
            TextBackground( LightGray ); { White background }
            TextColor( Black ); { Black text }
            write('           Please press a key ');
            WtKey := ReadKey; { Read a key }
            GotoXY(1, 1); { Cursor in upper-left corner of window }
            TextBackground( Black ); { Black background }
            TextColor( LightGray ); { White text }
            write(' ');
            Window(15, (20-FENTS) shr 1+4, 65, (20-FENTS) shr 1 +3+FENTS);
            GotoXY(1, FENTS); { Return cursor to old position }
            NumInScrn := 0; { Start counting at 0 }
        end;
        PrintData( DirBuf ); { Display entry data }
    until not(FindNext); { Are there any more entries? }

    Window(14, (20-FENTS) shr 1 +5+FENTS, 65, (20-FENTS) shr 1 +6+FENTS);
    GotoXY(1, 1); { Cursor in upper-left corner of window }
    TextBackground( LightGray ); { White background }
    TextColor( Black ); { Black text }
    write(' ');

    GotoXY(2, 1);
    case NumOfEntries of
        0 : write('No files found');
        1 : write('One file found');
        else write(NumOfEntries, ' files found')
    end;

    Window(1, 1, 80, 25); { Make entire screen a window }
end;

{*****}
{**                               MAIN PROGRAM                               **}
{*****}

begin
    case ParamCount of
        0 : Dir( '*.*', FA_AnyFile ); { All files in current directory }
        1 : Dir( ParamStr(1), FA_AnyFile ); { Display specific directory }
        else writeln('Invalid number of parameters');
    end;
end.

```