```
/*
DPMISHEL.C
1992 Andrew Schulman. Changed from version in October 1992 MSJ

Shell to run a simple C program in protected mode under DPMI
Must be compiled small model to use C run-time library
Calls v86_main(), switches into protected mode, then calls pmode_main()
*/

#include <stdlib.h>
#include <stdio.h>
#include <dos.h>
#include "dpmishel.h"

void _dos_exit(int retval)
{
  _asm mov ah, 04ch
  _asm mov al, byte ptr retval
  _asm int 21h
}

// Call the DPMI Mode Detection function (INT 2Fh AX=1686h) to see
// if we are *already* running in protected mode under DPMI.
// See 3.1 DDK _Device Driver Adaptation Guide_, pp. 585-586
int dpmi_present(void)
{
  unsigned _ax;
  _asm mov ax, 1686h
  _asm int 2Fh
  _asm mov _ax, ax
  return (! _ax);
}

// Call the DPMI function for Obtaining the Real to Protected Mode
// Switch Entry Point (INT 2Fh AX=1687h), to determine if DPMI is
// available and, if so, switch into protected mode by calling
// the Switch Entry Point.  See DPMI 0.9 spec.
int dpmi_init(void)
{
  void (far *dpmi)();
  unsigned hostdata_seg, hostdata_para, dpmi_flags;
  _asm push si
  _asm push di
  _asm mov ax, 1687h          /* test for DPMI presence */
  _asm int 2Fh
  _asm and ax, ax
  _asm jz gotdpmi             /* if (AX == 0) DPMI is present */
  _asm jmp nodpmi
gotdpmi:
  _asm mov dpmi_flags, bx
  _asm mov hostdata_para, si   /* paras for DPMI host private data */
  _asm mov word ptr dpmi, di
  _asm mov word ptr dpmi+2, es /* DPMI protected-mode switch entry point */
  _asm pop di
  _asm pop si
```

```
    if (_dos_allocmem(hostdata_para, &hostdata_seg) != 0)
        fail("can't allocate memory");

    /* printf("32-bit protected-mode programs %s supported\n",
        (dpmi_flags & 1) ? "are" : "are not"); */

    dpmi_flags &= ~1;   /* this is a 16-bit protected-mode program */

    /* enter protected mode */
    _asm mov ax, hostdata_seg
    _asm mov es, ax
    _asm mov ax, dpmi_flags
    (*dpmi)();

    return 1;
nodpmi:
    return 0;
}

main(int argc, char *argv[])
{
    int ret;

    // call routine for any V86 mode stuff
    if (v86_main(argc, argv) != 0)
        return 1;   // if v86_main() not return 0, don't go to prot mode

    if (! dpmi_present())
        if (! dpmi_init())
            fail("This program requires DPMI");

    /* now in protected mode */
    ret = pmode_main(argc, argv);
    flushall();     // flush all buffers before exiting
    _dos_exit(ret);
}
```