

C listing: NOKEYC.C

```

/*****
/*
/*              N O K E Y C
/*
/*-----*/
/*   Task           : Demonstrates clearing the keyboard buffer.
/*                   : This is useful for protecting the user from
/*                   : accidental keystrokes during an important
/*                   : command (e.g., deleting files).
/*-----*/
/*   Author          : Michael Tischer
/*   Developed on     : 01/01/92
/*   Last update      : 04/07/95
*****/

```

```

#include <stdio.h>
#include <dos.h>
#include <bios.h>

```

```

/*== Macros =====*/

```

```

#ifndef MK_FP
/* If MK_FP hasn't been defined, do so */
#define MK_FP(seg,ofs) \
    ((void far *) (((unsigned long)(seg) << 16) | (unsigned)(ofs)))
#endif

```

```

#ifdef __TURBOC__
/* Definitions for TURBO C */

```

```

#define GetKbKey()      ( bioskey( 0 ) )
#define GetKbReady()    ( bioskey( 1 ) != 0 )
#define GetBiosTime(x)  ( x = biostime( 0, NULL ) )
#define CLI()           ( disable() )
#define STI()           ( enable() )

```

```

#else                                     /* Definitions for Microsoft C Compiler */

#define GetKbKey()                       ( _bios_keybrd( _KEYBRD_READ ) )
#define GetKbReady()                     ( _bios_keybrd( _KEYBRD_READY ) != 0 )
#define GetBiosTime(x)                   ( _bios_timeofday( _TIME_GETCLOCK, &x) )
#define CLI()                            ( _disable() )
#define STI()                            ( _enable() )

#endif

/*== Screen routines for Microsoft C =====*/

#ifdef __TURBOC__                         /* Microsoft C? */

/*****
/* Gotoxy      : Places the cursor.
/* Input       : Cursor coordinates.
/* Output      : None
*****/

void gotoxy( int x, int y )
{
    union REGS regs;                     /* Registers for interrupt call */

    regs.h.ah = 0x02;                    /* Function number for interrupt call */
    regs.h.bh = 0;                       /* Color */
    regs.h.dh = y - 1;
    regs.h.dl = x - 1;
    int86( 0x10, &regs, &regs );        /* Interrupt call */
}

```

```

/*****
/* Clrscr      : Clears the screen.
/* Input       : None
/* Output      : None
*****/

void clrscr( void )
{
    union REGS regs;                /* Registers for interrupt call */

    regs.h.ah = 0x07;               /* Function number for interrupt call */
    regs.h.al = 0x00;
    regs.h.ch = 0;
    regs.h.cl = 0;
    regs.h.dh = 24;
    regs.h.dl = 79;
    int86( 0x10, &regs, &regs );   /* Interrupt call */
    gotoxy( 1, 1 );                 /* Set cursor */
}

#endif

/*****
/* Delay:      Halt program execution for a specific time.
/* Input       : PAUSE = Length of time interval in ticks
/* Output      : None
/* Info        : One tick = 1/18.2 seconds
*****/

void delay( unsigned int pause )
{
    long curtime,                   /* Current time.. */

```

```

        targtime;                                /* to target time */

if ( pause )                                     /* Pause not 0? */
{
    GetBiosTime( targtime );                     /* No */
    targtime += (long) pause;                    /* Count to target time */

    do                                           /* Delay loop - get current time */
        GetBiosTime( curtime );
    while ( curtime <= targtime );              /* Time elapsed? */
}                                               /* Yes --> End function */
}

/*****
/* ClearKbBuffer : Clears the contents of the keyboard buffer.      */
/* Input      : None                                                  */
/* Output     : None                                                  */
*****/

void ClearKbBuffer( void )
{
    CLI();                                       /* CLI: Disable hardware interrupts */
    *(int far *) MK_FP(0x40,0x1a) =           /* No more characters in buffer */
    *(int far *) MK_FP(0x40,0x1C);
    STI( );                                    /* STI: Enable hardware interrupts */
}

/*****
/*
/*          M A I N   P R O G R A M
/*
*****/

void main( void )

```

```

{
    int            i,                                /* Loop counter */
                  ccount;        /* Number of character in keyboard buffer */
    unsigned char ch;                                /* Get keys */

    clrscr();
    printf( "NOKEYC - (c) 1992 by Michael Tischer\n\n" );
    printf( "Keyboard buffer purged when counter reaches 0.\n\n" );

    for ( i = 10; i; --i )                            /* Give user time to */
    {                                                    /* press some keys */
        printf( "%5d", i );
        delay( 13 );                                /* Pause for .75 seconds */
    }

    /* ClearKbBuffer(); */                            /* Clear the buffer */

    /*-- Display characters still in keyboard buffer -----*/

    ccount = 0;                                        /* No more characters */
    printf( "\n\nCharacters in keyboard buffer :\n" );

    while GetKbReady()                                /* Any more characters in keyboard buffer? */
    {                                                    /* Yes --> Read and display */
        ch = GetKbKey();
        printf( " %3d ", (int) ch );                /* Display code only first */
        if ( (int) ch > 32 )                            /* Code > 32? */
            printf ( "(%c)", ch );                    /* Yes --> Display character as well */
        printf("\n");
        ++ccount;                                        /* More than one character found */
    }
}

```

```
if ( ccount == 0 )  
    printf( "(None)\n" );  
printf( "\n" );  
}
```

```
/* Out of characters? */  
/* Done */
```