

Periféricos y Dispositivos de Interfaz Humana

Práctica 1



**UNIVERSIDAD
DE GRANADA**

Luis Miguel López López

Prueba de cursores:

Grueso:

```
C:\P1-EJE~1\P1-EJE~1>p1  
Visualizacion del cursor modo grueso:█
```

Vuelta al normal:

```
Visualizacion del cursor modo grueso:  
Volviendo a modo cursor normal._
```

Función de cambio:

```
void setcursortype(int modo){  
    union REGS inregs, outregs;  
    inregs.h.ah = 0x01;  
    switch(modo){  
        case 0: //invisible  
            inregs.h.ch = 010;  
            inregs.h.cl = 000;  
            break;  
        case 1: //normal  
            inregs.h.ch = 010;  
            inregs.h.cl = 010;  
            break;  
        case 2: //grueso  
            inregs.h.ch = 000;  
            inregs.h.cl = 010;  
            break;  
    }  
    int86(0x10, &inregs, &outregs);  
}
```

Se le pasa como valor el tipo de cursor al que se quiere cambiar y esto lo realiza cambiando el registro ch y cl al valor deseado.

Cambios de modo de video:

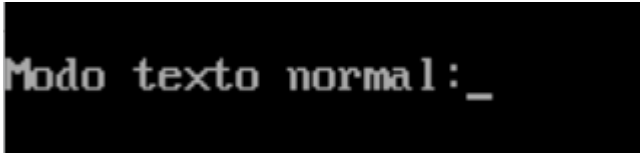


Modo grafico:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: P1



Modo texto grande: _



Modo texto normal: _

```
void setvideomode(int modo){
    union REGS inregs,outregs;
    inregs.h.al = modo;
    inregs.h.ah = 0;
    int86(0x10, &inregs, &outregs);
}
```

El modo está definido al principio del programa como un objeto del tipo BYTE.
Para obtener el modo actual:

```
int getvideomode(){
    union REGS inregs, outregs;
    int modo;
    inregs.h.ah= 15;
    int86(0x10,&inregs,&outregs);
    modo = outregs.h.al;
    return modo;
}
```

Devuelve un entero que almacenamos en una variable y posteriormente lo mostramos por pantalla:

```
modo = getvideomode();
printf("\nModo de video actual: %d",modo);
```

```
todo texto normal:  
todo de video actual: 3_
```

Prueba de colores:

```
Prueba de colores:
```



Dos funciones, una para colocar el color del propio caracter y otra para el fondo, a ambas funciones se le pasa el valor deseado:

```
void textcolor(int ncolor){  
    colortexto = ncolor;  
}  
void textbackground(int ncolor){  
    colorfondo = ncolor;  
}
```

Esto cambia el valor de sus respectivas variables para que en la función de imprimir un caracter se imprima con esos colores:

```
void cputchar(unsigned char c){  
    union REGS inregs, outregs;  
    inregs.h.ah = 0x09;  
    inregs.h.al = c;  
    inregs.h.bl = colorfondo<<4 | colortexto;  
    inregs.h.bh = 0;  
    inregs.x.cx = 1;  
    int86(0x10, &inregs, &outregs);  
}
```

Obtener un caracter e imprimirlo por pantalla:

```
void getch(){  
    union REGS inregs, outregs;  
    int caracter;  
  
    inregs.h.ah = 1;  
    int86(0x21, &inregs, &outregs);  
  
    caracter = outregs.h.al;  
}
```



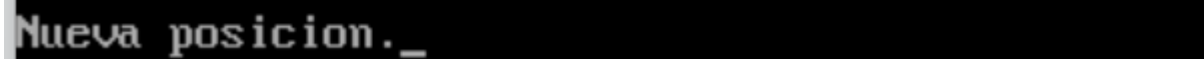
Introduce un caracter:
a

Directamente se imprime por pantalla, por lo que no hace falta devolver ningún valor.

Cambio de posición del cursor:



Colocando el cursor en la posicion 30,10



Nueva posicion._

```
void gotoxy(int x, int y){  
    union REGS inregs, outregs;  
    inregs.h.dl = x;  
    inregs.h.dh = y;  
    inregs.h.bh = 0;  
    inregs.h.ah = 2;  
    int86(0x10, &inregs, &outregs);  
}
```

Se le pasan los parámetros x e y de la terminal donde se quiera colocar el cursor.

Limpiar pantalla:

```
void clrscr(){
    setvideomode(MODOGRAFICO);
    setvideomode(MODOTEXTO);
}
```

Simplemente hace un cambio de modo entre el modo grafico y de vuelta al modo texto para limpiar la pantalla. Se podría modificar para que obtuviese el modo actual y volviese a ese modo.

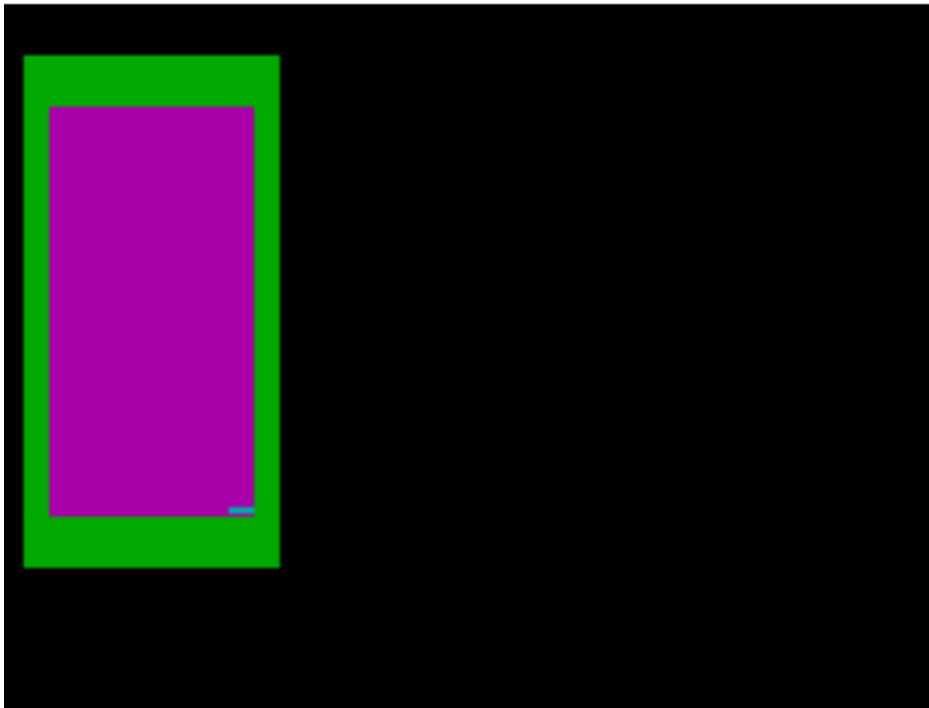
Dibujar un rectángulo:

```
void recuadro(int x1, int y1, int x2,int y2, int c1, BYTE c2){
    int i;
    int j;
    int k;
    int l;
    textbackground(c1);
    for(i=x1; i<=x2;i++){
        gotoxy(i,y1);
        cputchar(' ');
        gotoxy(i,y2);
        cputchar(' ');
    }
    for(i=y1; i<=y2;i++){
        gotoxy(x1,i);
        cputchar(' ');
        gotoxy(x2,i);
        cputchar(' ');
    }
    textbackground(c2);
    for(k=(x1+1);k<x2;k++){
        for(l=(y1+1);l<y2;l++){
            gotoxy(k,l);
            cputchar(' ');
        }
    }
}
```

Emplea el caracter espacio ' ' para dibujar el recuadro. Se le pasan las coordenadas de las dos esquinas y los colores de dentro y fuera. hace dos bucles primero para colorear los

bordes y finalmente el segundo es para rellenarlo. Previamente antes de cada pareja de bucle se establece el color de fondo.

 DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program:



Realizado con los siguiente valores:

```
mi_pausa();  
clrscr();  
recuadro(1,1,10,10,2,5);  
mi_pausa();
```

Dibujo en modo gráfico:

Para ello primero ejecuta la funcion de cambio de modo para colocar el modo gráfico. Usando la función pixel proporcionada en uno de los ejemplos de la práctica se dibuja la palabra PDIH usando unos cuantos bucles for.

```
void dibujo(){  
    int i;  
    int j;  
    int k;  
    int l;  
    setvideomode(MODOGRAFICO);  
    //P  
    for (i=0;i < 30; i++){  
        pixel(10,i+10,1);  
    }  
    for(j=0;j<15;j++){  
        pixel(j+11,10,1);  
        pixel(j+11,24,1);  
    }
```

```
}
for(k=0;k<13;k++){
    pixel(26,k+11,1);
}
//D
for (i=0;i < 30; i++){
    pixel(30,i+10,2);
}
for(j=0;j<15;j++){
    pixel(j+31,10,2);
    pixel(j+31,39,2);
}
pixel(46,11,2);
pixel(47,12,2);
pixel(46,38,2);
pixel(47,37,2);
for(j=0;j<24;j++){
    pixel(48,j+13,2);
}
//I
for (i=0;i < 30; i++){
    pixel(58,i+10,3);
}
for(j=0;j<15;j++){
    pixel(j+51,10,3);
    pixel(j+51,39,3);
}
//H
for (i=0;i < 30; i++){
    pixel(68,i+10,1);
    pixel(83,i+10,1);
}
for(j=0;j<14;j++){
    pixel(j+69,25,1);
}
}
```


Resultado:

 DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: P1



Función pixel:

```
void pixel(int x, int y, BYTE C){
    union REGS inregs, outregs;
    inregs.x.cx = x;
    inregs.x.dx = y;
    inregs.h.al = C;
    inregs.h.ah = 0x0C;
    int86(0x10, &inregs, &outregs);
}
```

Finalmente en el programa principal se espera una pulsación de tecla, se coloca en modo texto y se finaliza la ejecución del programa.

```
dibujo();
mi_pausa();
setvideomode(MODOTEXT0);
return 0;
```

Función mi_pausa también proporcionada por el profesor:

```
void mi_pausa(){
    union REGS inregs, outregs;
    inregs.h.ah = 8;
    int86(0x21, &inregs, &outregs);
}
```