FUNCIONES

Para empezar vamos a explicar brevemente lo que hace cada función que hemos creado:

- **gotoxy(int x, int y):** Pasamos como parámetro las coordenadas en las que queremos colocar el cursor y establecemos mediante los registros dh y dl correspondientes a filas y columnas los valores deseados.

```
//Función para ir a una coordenada (x,y)
void gotoxy(int x, int y){
        union REGS inregs, outregs;
        inregs.h.dh=y;                  //Establecemos la posicion y correspondiente a las filas
        inregs.h.dl=x;                  //Establecemos la posición x correspondiente a las columnas
        inregs.h.bh=0;
        inregs.h.ah=2;
        int86(0x10,&inregs,&outregs);
}
```

- **setcursortype(int tipo):** Para esta función estableceremos un tipo de cursor para cada modo, lo que hacemos aquí es establecer un valor a los registros ch y cl de forma que parezca que el cursor es mas fino o mas grueso, los modos corresponden a 1-invisible, 2-fino, 3-grueso.

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

- **setvideomode(BYTE modo):** Establecemos el modo de video cambiando el valor del registro al; 3-Modo texto 4-Modo gráfico.

- **getvideomode():** Esta función devuelve el modo de video actual, si estamos en modo texto también devuelve el número de columnas

```c
//Función que establece el modo de video 3-Texto 4-Grafico
void setvideomode(BYTE modo){
        union REGS inregs, outregs;
        inregs.h.ah=0x00;
        inregs.h.al=modo;
        int86(0x10, &inregs, &outregs);
}

//Función que devuelve el modo de vídeo actual
BYTE getvideomode(){
        BYTE modo;
        union REGS inregs, outregs;
        inregs.h.ah=0x0F;
        int86(0x10, &inregs, &outregs);

        modo=outregs.h.al;

        if(modo == 3)
                return outregs.h.ah;
        else
                return modo;
}
```

- **pixel(int x, int y, BYTE c):** Pinta un pixel de un determinado color c en las posición (x,y), la función *pintaCubo()* hará uso de esta para pintar por pantalla un cuadrado.

```c
//Funcion que pinta un pixel en las coordenadas x,y del color indicado
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);
}

//Función para pintar un cubo
void pintaCubo(int esq_sup, int tamanio){
        int i;

        setvideomode(MODOVIDEO);

        for(i=esq_sup; i<tamanio; i++){
                pixel(i, esq_sup, 1);
                pixel(i, tamanio, 1);
        }

        for(i=esq_sup; i <= tamanio; i++){
                pixel(esq_sup, i, 2);
                pixel(tamanio, i, 2);
        }

        pausa();
        setvideomode(MODOTEXTO);
}
```

- **textcolor(int color) y textbackground(int color):** Estas funciones cambiarán el contenido de las variables globales de color de forma que todo lo que mostremos por pantalla tenga el color de texto y de fondo especificados en las funciones.

```
//Funcion para cambiar el color del texto

void textcolor(int color){      //Cambiaremos el color de texto global
        colortexto=color;
}

void textbackground(int color){ //Cambiaremos el color de fondo global
        colorfondo=color;
}
```

- **clrscr():** Función que imprime saltos de línea para limpiar la pantalla.

- **cputchar(char caracter):** Escribe por pantalla el carácter dado haciendo uso del color establecido en ese momento, por ejemplo si previamente hemos cambiado el color con la función *textcolor()* el carácter se mostrará de ese color.

```
//Funcion que escribe un caracter en pantalla con el color indicado actualmente
void cputchar(char caracter){
        union REGS inregs, outregs;
        inregs.h.ah=0x09;
        inregs.h.al=caracter;
        inregs.h.bl=colorfondo << 4 | colortexto;
        inregs.h.bh=0x00;
        inregs.x.cx=1;
        int86(0x10, &inregs, &outregs);
}
```

- **getche():** Función que lee un carácter desde el teclado y luego lo muestra con las características de textcolor y textbackground que tengamos especificadas, para eso llamaremos a *cputchar()* una vez tengamos el carácter a mostrar.

```
//Funcion que obtiene un caracter por teclado y lo muestra en pantalla
void getche(){
        union REGS inregs, outregs;
        int caracter;

        printf("Escribe un caracter");
        inregs.h.ah=1;
        int86(0x21, &inregs, &outregs);

        caracter=outregs.h.al;
        printf("\nHas pulsado:");
        cputchar(caracter);
}
```

PROGRAMA PRINCIPAL

Para comprobar la funcionalidad de todas las funciones creadas he creado un programa principal que realizará las siguientes tareas:

1-Primero mostraremos por pantalla los diferentes tipos de cursor posibles, indicando nombre y llamando a su correspondiente función. Haremos uso de la función pausa() que espera una pulsación de tecla para pasar por las distintas funcionalidades.

2-Después establecemos un color de texto y de fondo y hacemos la prueba mostrando un carácter por pantalla ‘c’. Pediremos con la función getche() un nuevo caracter y podremos comprobar que se imprime con las características especificadas anteriormente y no con las características por defecto.

3-Haremos una limpieza de pantalla con antes de dibujar el cuadrado y colocaremos el cursor en la posición (0,0) quedando así una pantalla “limpia”.

4-Por último cambiaremos a modo gráfico, dibujaremos por pantalla un cuadrado y luego cambiaremos de nuevo a modo texto.

Con este programa quedaría demostrado el correcto funcionamiento de todas las funciones implementadas.

```
int main(){

        printf("\nCursor invisible");
        setcursortype(0);
        pausa();

        printf("\nCursor normal");
        setcursortype(1);
        pausa();

        printf("\nCursor grueso");
        setcursortype(2);
        pausa();
        printf("\n");

        textcolor(1);
        textbackground(2);
        cputchar('c');
        printf("\n");

        getche();

        printf("\nRealizando limpieza de pantalla, pulse una tecla");
        pausa();
        clrscr();
        gotoxy(0,0);

        printf("Cambiando a modo video, pulsa una tecla\n");
        pausa();
        pintaCubo(10, 100);

        return 0;
}
```

EJEMPLO DE EJECUCIÓN.

Primero se muestran los diferentes tipos de cursor, luego se muestra un carácter por pantalla con los nuevos colores de background y de texto dados, se lee un caracter desde teclado y se muestra de la misma manera y por último se realiza una limpieza de pantalla y se cambia a modo video para dibujar

```
Turbo Link  Version 5.1 Copyright (c) 1992 Borland International

        Available memory 4149144
C:\P1>EJERCI~1.EXE

Cursor invisible
Cursor normal
Cursor grueso
c
Escribe un caracter5
Has pulsado:5
Realizando limpieza de pantalla, pulse una tecla
```

a continuación se muestra un cuadrado usando la función pixel