PRÁCTICA 3. MODO GRÁFICO

☑ OBJETIVOS:

El alumno aprenderá a manejar las instrucciones básicas de salida para programar en modo gráfico.

🛠 EQUIPO:

- 💻 Computadora
- 💾 Disco Flexible o algún otro medio para almacenar sus prácticas
- 🗗 Turbo C (TC)

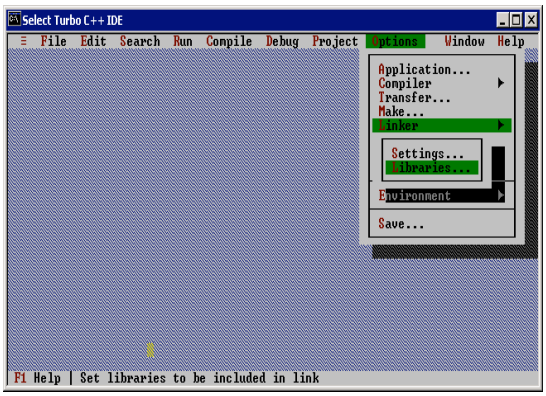
GENERALIDADES

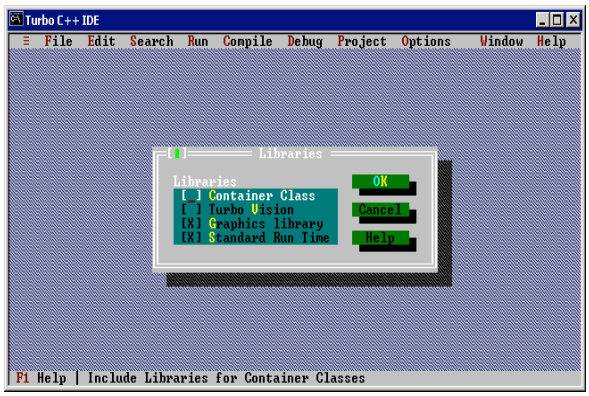
LA LIBRERÍA GRAPHICS.H

La calidad de los monitores es medida por la resolución ó nitidez que presentan. El manejo de pantalla para modo gráfico se utiliza esta resolución, imprimiendo en PIXELES. La resolución normal del monitor mide 640 x 480, 640 columnas por 480 renglones.

Los archivos necesarios en disco para el manejo de modo gráfico son <GRAPHICS.H> , los archivos *.BGI y los archivos *.CHR. Los archivos BGI son los dispositivos detectores del monitor y los CHR son los distintos archivos de FONTS ó tipos de letras. La librería GRAPHICS es una utilería de Turbo C que cuenta con una serie de instrucciones reservada para el manejo de gráficos.

Dentro de las opciones de configuración debe estar marcado con (X) la opción de Gráficos, seleccione la opción dentro del submenú OPTIONS, la de LINKER, y luego opción LIBRARIES.





INICIALIZACIÓN DE GRÁFICOS.

Para la activación del modo gráfico se procede a su inicialización con las siguientes instrucciones:

```
graphDriver = DETECT;
```

`graphDriver` y `graphMode` son variables tipo `int` (valores numéricos enteros) declaradas por el usuario para detectar el tipo de monitor y su resolución. `DETECT` es palabra reservada en mayúsculas que realiza la detección del monitor automáticamente. Para inicializar la pantalla se utiliza la instrucción reservada:

```
initgraph(&graphDriver, &graphMode, driverPath);
```

`graphDriver` y `graphMode` nos dan las características físicas del monitor y el `driverPath` indica donde se localizan los archivos de gráficos, si se colocan comillas `<" ">` indica en el directorio en que estamos localizados. Para colocar el directorio donde se encuentran los BGI y CHR se añade el camino: `"C:\\TURBOC\\BGI"`

Para regresar a modo texto contamos con la instrucción: `restorecrtmode( );` Asimismo contamos con la instrucción de terminación de modo gráfico, necesaria al finalizar nuestro programa: `closegraph( );`

- IMPRESIÓN DE DATOS EN MODO GRÁFICO.

Las instrucciones más utilizadas en modo gráfico para el manejo de texto son las siguientes:

INSTRUCCIÓN	CARACTERÍSTICA	EJEMPLO
`clearviewport()`	Limpiar pantalla en modo gráfico	`clearviewport();`
`setcolor()`	Colorea el texto y líneas	`setcolor(YELLOW);`
`setbkcolor()`	Colorea el fondo de la pantalla	`setbkcolor(BLUE);`
`outtext()`	Imprimir un mensaje en pantalla	`outtext("Ingeniería");`
`moveto(x,y)`	Localizar en coordenadas x, y	`moveto(100,100);`
`outtextxy(x,y,cadena)`	Imprimir texto en coordenadas x, y	`outtextxy(100,100,"Ingeniería");`
`getmaxx()`	Obtiene coordenada máxima en x	`getmaxx();`
`getmaxy()`	Obtiene coordenada máxima en y	`getmaxy();`
`settextstyle(f,d,t)`	Da formato al texto en un tipo de letra (font), dirección y tamaño	`settextstyle(1,0,0);`

FONTS Tipos de letra básicos:	
0 Defaultfont	1 Triplexfont
2 Smallfont	3 Sanseriffont
4 Gothicfont	5 Userfont

EJEMPLO 1

```
#include <conio.h>
#include <stdio.h>
#include <graphics.h>

void main(){
   int gd, gm ;
   gd = DETECT;
   initgraph(&gd,&gm, "c:\\tc\\bgi");
   setbkcolor(BLUE);
   setcolor (WHITE);
   outtextxy(100,100,"Ingeniería");
   getch();
}
```

MANEJO DE FIGURAS EN MODO GRÁFICO.

Las instrucciones básicas para realizar figuras en modo gráfico son:

INSTRUCCIÓN	CARACTERÍSTICA	EJEMPLO
`putpixel(x,y,color)`	Pinta un pixel en las coordenadas X,Y indicadas	`putpixel(100,100,3);`
`line(xizq,ysup,xder,yinf)`	Dibuja una línea del primer punto a un segundo punto	`line(50,50,100,50);`
`lineto(xpos,ypos)`	Dibuja una línea de la posición actual a las coordenadas indicadas.	`lineto(100,50,100,90);`
`linerel(dx,dy)`	Dibuja una línea de la posición actual a una segunda coordenada obtenida de los incrementos indicados.	`linerel(200,100);`
`setlinestyle(tipo,a,b)`	Determina el estilo de línea : 0 línea sólida 1 punteada 2 guión centrado 3 guión quebrado	`setlinestyle(1,0,0);`
`rectangle(x1,y1,x2,y2)`	Dibuja un rectángulo del primer punto a un segundo punto	`rectangle(10,10,80,80);`
`bar(x1,y1,x2,y2)`	Dibuja una barra del primer punto a un segundo punto	`bar(40,40,100,100);`
`bar3d(x1,y1,x2,y2,z,tapa)`	Dibuja una cubo del primer punto a un segundo punto, con una distancia Z hacia atrás y con tapa si se coloca el valor de verdadero ó sin tapa con un valor de falso	`bar3d(30,30,90,90,20,1);`
`circle(xpos,ypos,radio)`	Dibuja un círculo en la posición indicada	`circle(300,200,50);`
`arc(x,y,angi,angf,radio)`	Dibuja un arco del círculo en la posición indicada	`arc(100,100,30,190,50);`
`ellipse(x,y,angi,angf,xr,yr)`	Dibuja una elipse ó parte de una elipse en la posición indicada, con las diagonales indicadas por los radios	`ellipse(80,80,5,90,10,80);`
`fillellipse(x,y,xrad,yrad)`	Dibuja y rellena una elipse en la posición indicada	`fillellipse(80,200,20,40);`
`pieslice(x,y,angi,angf,xrad)`	Dibuja y rellena una parte de círculo en la posición indicada	`pieslice(80,80,5,80,5,10);`
`sector(x,y,angi,angf,xr,yr)`	Dibuja y rellena una parte de una elipse	`sector(80,80,5,180,5,10);`
`setfillstyle(estilo,color)`	Determina el estilo de relleno : 0 -emptyfill 6 -ltbkslashfill 1 -solidfill 7 -hatchfill 2 -linefill 8 -xhatchfill 3 -ltslashfill 9 -interleavefill 4 -slashfill 10-widedotfill 5 -bkslashfill 11-closedotfill	`setfillstyle(9,RED);`

EJEMPLO 2

```
#include <graphics.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
void main(){
   int gd, gm, i,x,y ;
   /* inicialización de gráficos */
```

```
   gd=DETECT;
   initgraph( &gd, &gm, "C:\\TC\\BGI" );
   /*  manejo de texto */
   setbkcolor(BLUE);
   setcolor(WHITE);
   settextstyle(4,0,0);
   outtextxy(100,100, " Ing. en Computación " );
   settextstyle(3,0,0);
   outtextxy(100,150, " Ing. en Computación " );
   settextstyle(2,0,0);
   outtextxy(100,200, " Ing. en Computación " );
   settextstyle(1,0,0);
   outtextxy(100,250, " Ing. en Computación " );
   settextstyle(0,0,0);
   outtextxy(100,300, " Ing. en Computación " );
   getch();
   /*  manejo de figuras */
   clearviewport();
   for(i=1;i<500;i++) {
      putpixel(rand()%getmaxx(),rand()%getmaxy(),GREEN);
   }
   getch();
   /*  LINEAS  */
   clearviewport();
   setlinestyle(3,0,0);
   line(100,50,540,50);
   setlinestyle(2,0,0);
   line(100,150,540,150);
   setlinestyle(1,0,0);
   line(100,250,540,250);
   setlinestyle(0,0,0);
   line(100,350,540,350);
   getch();
   /*  CUADROS  */
   clearviewport();
   rectangle(50,100,150,200);
   bar(250,100,350,200);
   bar3d(450,100,550,200,50,1);
   getch();
   /*  CURVAS  */
   clearviewport();
   circle(100,150,75);
   arc(250,150, 45,270,50);
   ellipse(400,150,0,360,50,100);
   ellipse(550,150,90,320,75,40);
   fillellipse(100,350,50,100);
   fillellipse(220,350,40,40);
   pieslice(350,350,120,320,50);
   sector(500,350,20,220,100,25);
   getch();

   /*  RELLENO  */
   setfillstyle(1,RED);
   getch();
}
```

✍ ACTIVIDAD

Realizar una presentación en modo gráfico donde aplique diversidad de figuras, movimiento, colores y sonido con creatividad y originalidad.