Guía Práctica: Modo Gráfico.
Desarrollo de Aplicaciones y Manipulación de Datos.

C++

Prof: Mileti

Para realizar gráficos en C++ necesitamos poner el sistema en modo gráfico. Para ello debemos incluir a nuestro programa la biblioteca de gráficos GRAPHICS.H

#include <graphics.h>

Para inicializar el modo gráfico se utiliza la función *initgraph()* y para detenerlo la función *closegraph()*.

A la función *initgraph* se le deben pasar 3 argumentos que permitirán detectar de forma automática la placa de video y fijar el modo de video más alto posible:

```
int driver=DETECT, modo;
initgraph(&driver,&modo,"C:\\TC\\BGI");
```

El primer parámetro (&driver) es asignado a DETECT, por la tanto la función *detectgraph()* es llamada, y un dispositivo y modo gráfico (segundo parámetro) apropiados serán seleccionados.
El tercer parámetro especifica el directorio donde los dispositivos gráficos están localizados.

Si la función *initgraph()* falla, puede interceptarse un código de error e informar al usuario de dicho problema:

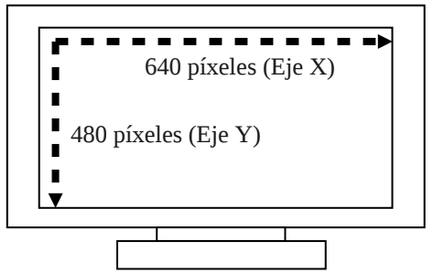
```
if (graphresult()!= grOk) {
  cout << "Error al iniciar el modo gráfico, el programa finalizará";
  getch();
   exit (1);
}
```

De ahora en adelante es posible utilizar cualquier función gráfica en nuestro programa.

Para finalizar el modo gráfico ejecutamos la siguiente instrucción:

closegraph();

Generalmente al iniciar el modo gráfico se dispone de una resolución de 640x480. Es decir disponemos de 640 pixeles en el eje x y 480 pixeles en el eje y.



Para comprobar esto, podemos hacer uso de la funciones *getmaxx()* y *getmaxy()* que devuelven el máximo píxel del eje x e y respectivamente.

cout<<"Resolución: " << getmaxx() << "x" << getmaxy();

En caso de contar con una resolución de 640x480, la línea de código anterior mostraría en pantalla lo siguiente:

Resolución: 639x479

Esto se debe a que el último píxel de eje x es 639, y el último píxel del eje y es 479, ya que ambos ejes comienzan en el píxel 0.

Establecer colores
Para seleccionar el color de fondo en modo grafico debemos utilizar la función *setbkcolor()* y para establecer el color de dibujo *setcolor()*.

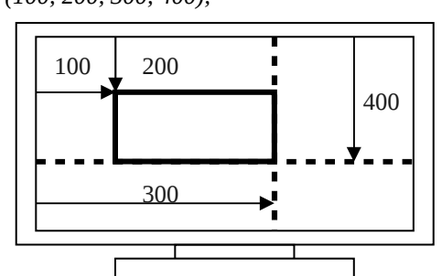
Para limpiar la pantalla en modo gráfico se utiliza la función c*leardevice()* que dejará la pantalla con el color establecido en *setbkcolor()*.

Funciones más utilizadas:

- **rectangle (int X1, int Y1, int X2, int Y2);**

Dibuja un rectángulo sin relleno en la pantalla marcado por los puntos (X1, Y1) y (X2, Y2), donde X1 es el extremo izquierdo del rectángulo, Y1 es el extremo superior, X2 es el extremo derecho e Y2 el extremo inferior del rectángulo.

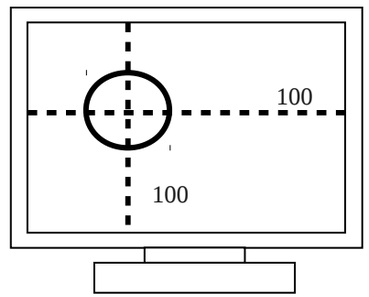
rectangle (100, 200, 300, 400);



- **circle (int X, int Y, int radio);**

Dibuja un circulo (cuyo tamaño de radio está dado por la variable radio) en la pantalla con centro en los puntos (X, Y).

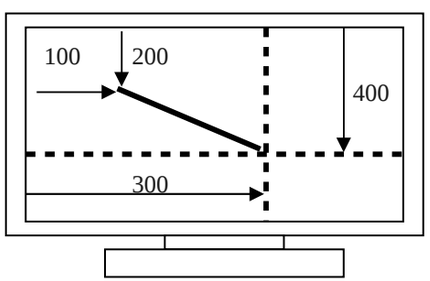
circle (100, 100, 10);



- **line (int X1, int Y1, int X2, int Y2);**

Dibuja una línea recta iniciando en los puntos X1, Y1; concluyendo en X2, Y2.

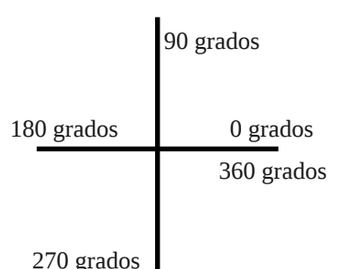
line (100, 200, 300, 400);



- **ellipse (int X, int Y, int stangle, int endangle, int xradius, int yradius);**

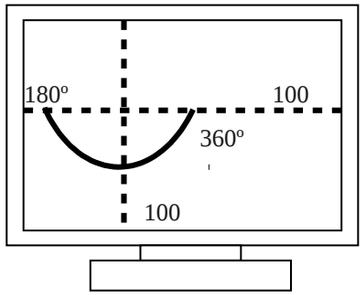
Dibuja un elipse con centro en los puntos X, Y.

stangle y endangle indican el ángulo inicial y final del arco elíptico a dibujar. Para ello se toma como referencia la siguiente figura:

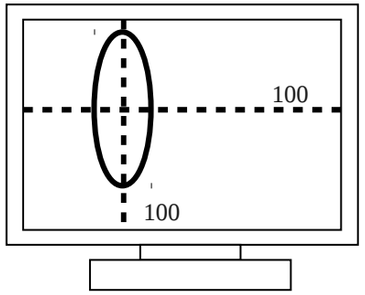


Mediante xradius e yraduis se le puede establecer la forma de óvalo al elipse. Veamos un ejemplo de arco elíptico:

ellipse(100,100,180,360,50,50);



ellipse(100,100,0,360,20,60);



Figuras con relleno:
En las funciones anteriores no es posible establecer un relleno a las figuras creadas. Las funciones que sí aceptan un estilo de relleno son: *bar*, *bar3d*, *fillelipse y pieslice*, entre otras.

Para indicar el estilo de relleno de estas figuras se utiliza la función *setfillstyle* de la siguiente manera:

setfillstyle(int pattern, int color);

El valor de *pattern* indica el tipo de relleno, pudiendo tomar uno de los siguientes valores:

EMPTY_FILL	SOLID_FILL	LINE_FILL
LTSLASH_FILL	SLASH_FILL	BKSLASH_FILL
LTBKSLASH_FILL	HATCH_FILL	XHATCH_FILL
INTERLEAVE_FILL	WIDE_DOT_FILL	CLOSE_DOT_FILL
USER_FILL		

Color representa el color de relleno (del 0 al 15 o constante de color).

Ejemplos:

```
//Dibuja una barra bidimensional con relleno sólido de color verde
setfillstyle(SOLID_FILL, GREEN);
bar(100,10,150,90);

//Dibuja una barra tridimensional con líneas de relleno de color cyan
setfillstyle(LINE_FILL, CYAN);
bar3d(200,20,250,90,15,1);

//Dibuja un elipse y lo rellena con líneas oblicuas de color rojo.
setfillstyle(SLASH_FILL, 4);
fillellipse(50,200,40,70);

//Dibuja un sector con relleno cuadriculado de color blanco
setfillstyle(HATCH_FILL,WHITE);
pieslice(150,200,0,45,100);
```

Crear animaciones utilizando ciclo for
Mediante el ciclo for es posible controlar el cambio de algunos parámetros de las funciones gráficas, generando de esta manera algún tipo de animación. A continuación un ejemplo:

```
#include<graphics.h>
#include<conio.h>
#include<iostream.h>
#include<stdlib.h>
#include<dos.h>
void main(){
int driver=DETECT, modo, i;
clrscr();
initgraph(&driver,&modo,"C:\\TC\\BGI");
if (graphresult()!=grOk){
   cout<<"Error al iniciar modo gráfico.";
   getch();
   exit(1);
}
setfillstyle(SOLID_FILL,BLUE);
for(i=300;i>1;i--){
  setcolor(RED);
  line(getmaxx()/2,getmaxy()/2,0, getmaxy());
  line(getmaxx()/2,getmaxy()/2,getmaxx(),getmaxy());
  setcolor(BLACK);
  fillellipse(getmaxx()/2,getmaxy()/2,i,i);
  delay(50);
  //cleardevice();
 }
closegraph();
}
```

Establecer tamaño y estilo del texto
Para indicar el formato del texto que aparecerá en pantalla durante el modo gráfico se utiliza la función *settextstyle:*

settextstyle(int font, int direction, int charsize);

El valor de *font* es una constante que indica la fuente a utilizar. Las constantes disponibles son:
DEFAULT_FONT TRIPLEX_FONT SMALL_FONT
SANS_SERIF_FONT GOTHIC_FONT
Direction es una constante que indica la orientación del texto, puede ser HORIZ_DIR (el texto se muestra horizontalmente de izquierda a derecha) o VERT_DIR (el texto se muestra con sentido vertical de abajo hacia arriba).
Charsize se corresponde con un número entero que indica el tamaño de la fuente. Los valores van del cero en adelante, aumentando el tamaño de la letra. A continuación se establece el formato para dibujar un texto.

settextstyle(SANS_SERIF_FONT, HORIZ_DIR, 4);

Una vez indicado el formato del texto podemos presentar en pantalla una leyenda a través de la función *outtextxy()*:

*outtextxy(int x, int y, char *textstring);*

Los valores de *x* e *y* establecen las coordenadas en donde el texto será dibujado.
Textstring se refiere a la cadena de caracteres que se mostrará en pantalla.

outtextxy(100,200,"Esto es modo gráfico!!!");

Colores disponibles (modo texto)

Constante	Valor
BLACK	0
BLUE	1
GREEN	2
CYAN	3
RED	4
MAGENTA	5
BROWN	6
LIGHTGRAY	7
DARKGRAY	8
LIGHTBLUE	9
LIGHTGREEN	10
LIGHTCYAN	11
LIGHTRED	12
LIGHTMAGENTA	13
YELLOW	14
WHITE	15
BLINK	128