Círculos con el Mouse

Funciones del programa:

- **void init(void)**
- **void pixel(int x,int y)**
- **void polar(int xc,int yc,int r, float a)**
- **void circulo(int x,int y, int r)**
- **void circulo2(int xc,int yc, int radio)**
- **void puntos(int xc,int yc, int x, int y)**
- **void lineSegment(void)**
- **int main(int argc, char** argv)**
- **void onMouse(int button, int state, int x, int y)**
- **void onMotion(int x, int y)**
- **void onPassive(int x,int y)**

Main

El programa comienza la ejecución con la función main(), la cual maneja todo lo referente a la correcta ejecución del programa.

Init (void)

Se encarga de Iniciar la ventana, pero a su vez dando otros atributos a la misma dejando todo listo para poder trabajar con la ventana.

Pixel (int x, int y)

Función que pinta un pixel en las coordenadas x, y que le llegan por parámetro.

- glBegin(GL_POINTS);

  Comienza el trazado en el modo GL_POINTS.

- glVertex2i(x, y);

  Pinta un punto en la coordenada x, y del plano.

- glEnd();

  Termina el trazado de puntos, aquí finaliza la función.

Void polar (int xc,int yc,int r, float a)

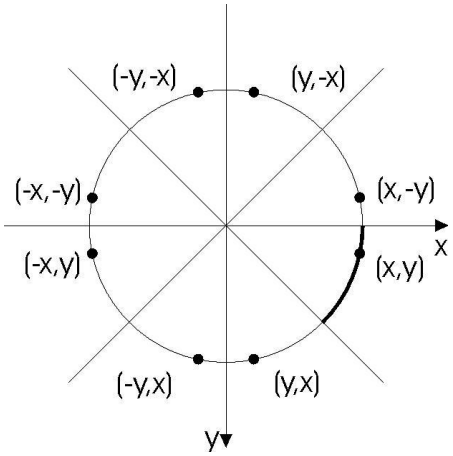
Función encargada de transformar las coordenadas rectangulares a coordenadas polares usando un par de ecuaciones de la forma:

$$X = X_0 + rCos\theta$$

$$Y = Y_0 + rSen\theta$$

Void puntos (int xc,int yc, int x, int y)

En esta función por principios de simetría del círculo, se pintan los pixeles de ¼ parte del circulo y los otros ¾ se calculan cambiando las coordenadas que se han mandado.



Void circulo (int x,int y, int r)

Calcula un círculo de 360 puntos de perímetro con las coordenadas y el radio proporcionado. Esta función se auxilia de otras, como son:

- polar(x,y,r,i)
- pixel(punto.x,punto.y)

Con las cuales se convierten primero a coordenadas polares y luego se trazan estas coordenadas.

Void circulo2 (int xc,int yc, int radio)

Esta función se podría decir que es una adaptación de la línea de Bresenham para dibujar círculos, en la que básicamente se calcula el incremento de θ para que quede un círculo completamente cerrado y donde además se tiene un parámetro de decisión con el cual se sabe que coordenadas deberán aumentar para dar la forma del círculo.

lineSegment (void)

Función encargada del dibujado de las líneas con el mouse, para ello se presenta el siguiente código:

```
glClear(GL_COLOR_BUFFER_BIT);

glColor3f(0.0, 1.0, 0.0); // color de la línea

glClear(GL_COLOR_BUFFER_BIT| GL_DEPTH_BUFFER_BIT);
```

```
glDrawPixels(400,400,GL_RGB,GL_UNSIGNED_BYTE,colorArray);

circulo(xc,yc,r);// dibujado de un circulo de 360 puntos

glFlush();

glutSwapBuffers();
```

Para comenzar esta función asigna el color de la línea y dibuja con **DrawPixels** lo que se tenga almacenado, esto se verá reflejado cuando se dibuje más de 1 círculo.

Void onMouse(int button, int state, int x, int y)

Con esta función se logra ejecutar cuando se produce algún evento en la ventana. Tiene parámetros los cuales son los datos del mouse, **button** es el botón con el que se creó el evento, **state** es el estado que tiene el botón, si está presionado o fue liberado, **x** y **y** son las coordenadas del mouse.

El código utilizado es el siguiente:

```
if ( (button == GLUT_LEFT_BUTTON) & (state == GLUT_DOWN) ) {

            xc=x;

            yc=abs(400-y);

}
```

Donde si el botón fue el izquierdo y está presionado cambiamos las coordenadas iniciales para después dibujar la línea. Las finales se actualizan en con **void onMotion.**

Void onMotion(int x, int y)

Con esta función estamos calculando las coordenadas del mouse, ya que se está ejecutando cada vez que se detecta algún movimiento del mouse sobre la ventana.

El código utilizado es:

```
dx=abs(xc-x);

dy=abs(yc-abs(400-y));

r=sqrt(pow(dx,2)+pow(dy,2));

glutPostRedisplay();
```

Void onPassive(int x,int y)

Función encargada de dibujar el círculo definitivo. Se ejecuta cuando en el ratón se ha soltado el click izquierdo. Con esto se capturan las coordenadas finales y se comienza el dibujado del círculo definitivo.

Esto se hace borrando la pantalla, dibujando lo que se tenía guardado (por si existiera algún circulo ya dibujado), dibujando el circulo definitivo con circulo2(xc,yc,r), y por ultimo guardando una “imagen” ya con el circulo dibujado.