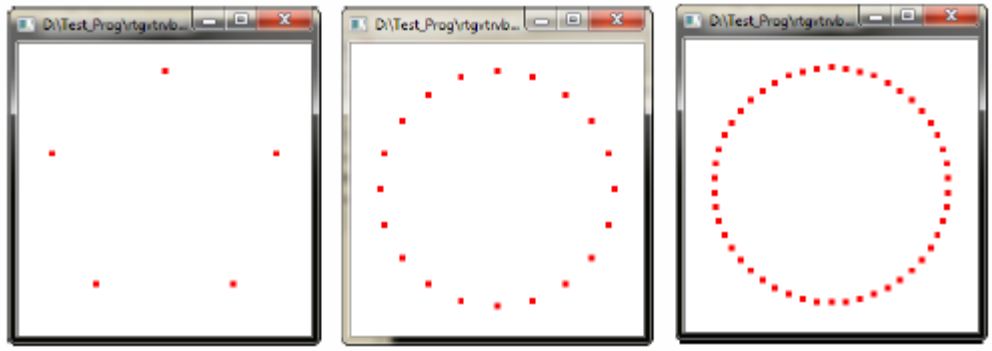
Nos gustaría generar los cinco vértices de un pentágono sobre una circunferencia circunscrita. Para esto podemos usar las siguientes ecuaciones paramétricas para describir la circunferencia (donde 0 <= Φ <=2π)

x= radio sen Φ
y= radio cos Φ



Posteriormente se irá incrementando el número de coordenadas para obtener una mejor representación gráfica de la circunferencia.

Un programa en C++ que resuelve el problema planteado, se describe a continuación.

```
 #include <windows.h>
 #include <stdlib.h>
 #include <math.h>
 #include <GL/glut.h>
 #define GL_PI 3.1416f
 void init(void);
 void display(void);
 void reshape(int,int);
 int main(int argc, char** argv)
 {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
   glutInitWindowSize(250,250); // comentar esta línea y la siguiente,
observar que pasa
   glutInitWindowPosition(100,100); //
   glutCreateWindow(argv[0]);
   init();
   glutDisplayFunc(display);
   glutReshapeFunc(reshape);
   glutMainLoop();
   return 0;
 }
 void init(void)
 {
   glClearColor(0.0,0.0,0.0,0.0); //parámetros: rojo, amarillo y azul, el
cuarto es el parámetro alpha
   glShadeModel(GL_SMOOTH);
 }
```

```
 void display(void)
 {
   GLfloat ang, radio = 8.0f, x, y;
   glClear(GL_COLOR_BUFFER_BIT);
   glBegin(GL_POINTS);
   for (ang = 0.0f; ang < 2 * GL_PI; ang += 2*GL_PI/5)
   {
      x = radio * cos(ang);
      y = radio * sin(ang);
      glVertex2f(x,y);
   }
  glEnd();
  glFlush();
}
 void reshape(int w, int h)
 {
   glViewport(0,0,(GLsizei)w, (GLsizei)h);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   glOrtho(-10.0,10.0,-10.0,10,-10.0,10.0);
   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
 }
```

Ejercicio 01:

Partiendo del programa ejemplo realice las siguientes modificaciones:

1. En lugar de los puntos, ahora trabaje con líneas (GL_LINES), con poli-líneas (GL_LINE_STRIP), con poli-línea cerrada (GL_LINE_LOOP) y ejecuta el programa.
2. Usa diferentes grosores y colores para los puntos de la figura y ejecute el programa.
3. Diseña diferentes figuras: circunferencia, pentágono, hexágono, etc .

Ejercicio 02:

Se puede dibujar discos, discos huecos y discos parciales, para esto OpenGL emplea los objetos cuadráticos, las cuales emplean modeladores cuadráticos para una mejor visualización; estas funciones la provee la librería glu:

```
void gluDisk(GLUquadricObj *pt, GLdouble rinterior, GLdouble rexterior, GLint nlados,
GLint nvueltas)
```

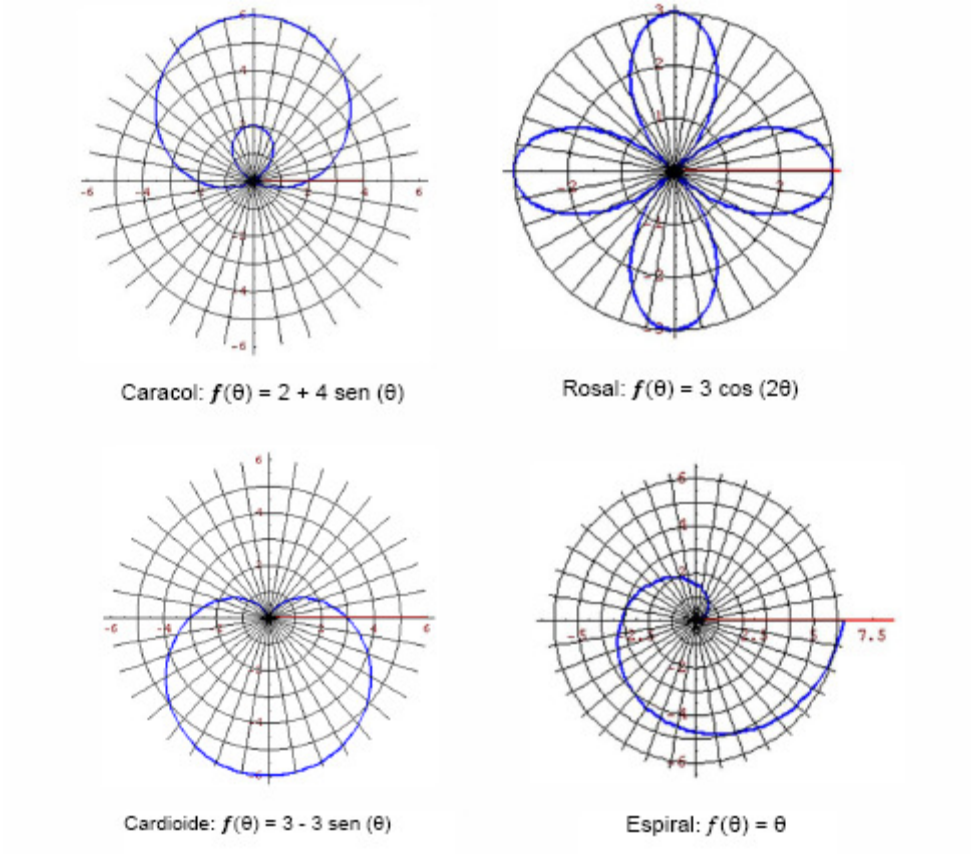
Dibuja un disco centrado sobre el plano XY, si `rinterior=0` entonces dibuja un disco sólido; el borde se representa con `nlados` (número de lados del disco), mientras que `nvueltas` proporciona el número de anillos que se genera radialmente. Finamente `*pu` es una variable de tipo Quadric.

A continuación se muestra como crear y dar valores a una variable cuadrática.

```
  GLUquadricObj *quadobj;
  quadobj = gluNewQuadric();
  gluQuadricDrawStyle( quadobj,GLU_LINE);
```

```
gluDisk(quadobj,0,5,36,3);
```

Utilizar el objeto cuadrático para pintar la “tela de araña” del fondo.
Partiendo del programa ejemplo generar las siguientes curvas:



Enlace http://es.wikipedia.org/wiki/Coordenadas_polares

Debéis utilizar un color para presentar el objeto cuadrático y otro para curva. También podéis utilizar distintos grosores.

El programa por defecto presentará una circunferencia de radio 8, posteriormente el programa deberá recoger los eventos de teclado y actuar de la siguiente manera:

- Si se teclea la ‘a’ se presentará la circunferencia.
- Si se teclea la ‘s’ se presentará el caracol.
- Si se teclea la ‘d’ se presentará el rosal.
- Si se teclea la ‘f’ se presentará la cardioide.
- Si se teclea la ‘g’ se presentará la espiral.

Para gestionar los eventos de teclado se recomienda leer el fichero 08_EJEMPLO_EVENTO_TECLADO.pdf que se encuentra en UBU-Virtual en la carpeta DOCUMENTOS INICIALES DE OPENGL

Cada alumno deberá entregar un fichero con el código fuente del ejercicio 02.

El nombre del fichero debe ser: PEC1_apellido1_apellido2_nombre.cpp

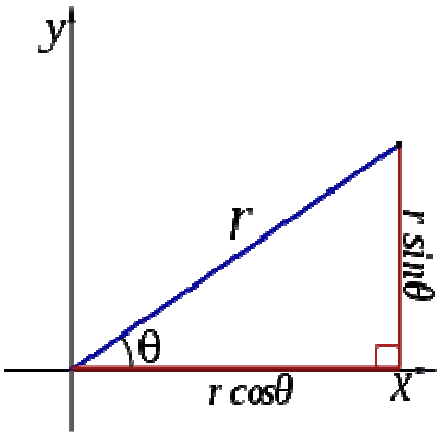
Cada alumno deberá entregar un vídeo (máximo 10 minutos) comentando el código del programa y mostrando su ejecución. Para grabar el video se recomienda utilizar la herramienta https://screencast-o-matic.com/

En el plano de ejes xy con centro de coordenadas en el punto (0,0) se puede definir un sistema de coordenadas polares de un punto **M** del plano, definidas por la distancia **r** al centro de coordenadas, y el ángulo θ del vector de posición sobre el eje **x**.

Conversión de coordenadas polares a rectangulares o cartesianas

Definido un punto en coordenadas polares por su ángulo θ sobre el eje x, y su distancia **r** al centro de coordenadas, se tiene:

$$x = r\cos\theta$$
$$y = r\,sen\theta$$



En el código del ejemplo, dada la fórmula de la circunferencia en coordenadas polares *f(Θ) = r* sabemos obtener sus coordenadas cartesianas.

Aplicando la conversión anterior, debéis saber calcular las coordenadas cartesianas de las fórmulas en coordenadas polares de las figuras propuestas en el enunciado.