CARGA DE ARCHIVOS BMP CON OpenGL.

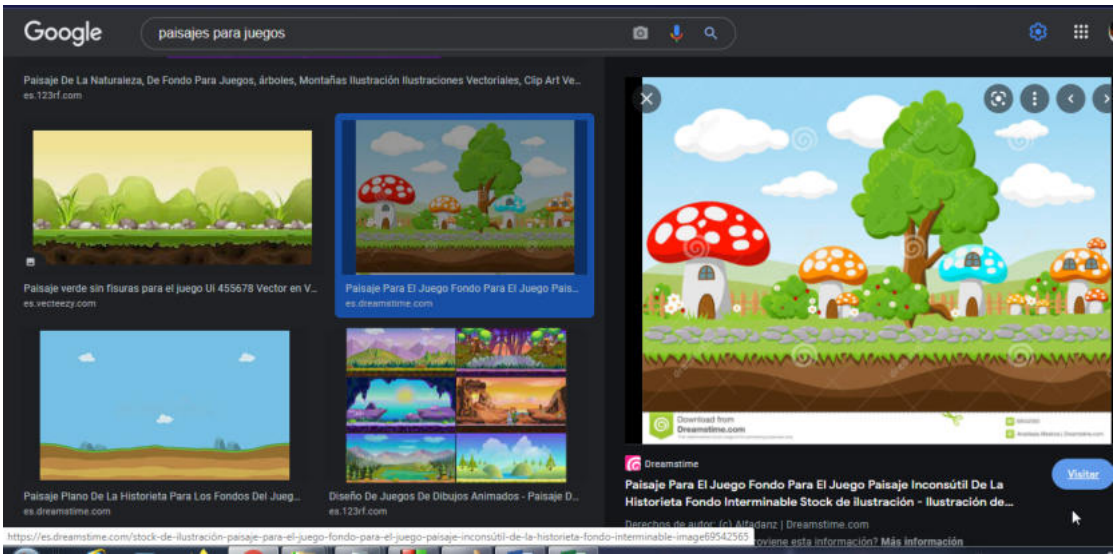
INTRODUCCIÓN.

Una textura, de acuerdo a la manera en que se trabaja en OpenGL, es un arreglo al que se le asigna o carga una imagen (un mapa de bits) y a partir de ese momento el arreglo representará la imagen la cual se podrá utilizar para mapearla sobre un polígono, esto quiere decir que para poder visualizar la imagen es necesario vaciar el contenido de este arreglo en un polígono.

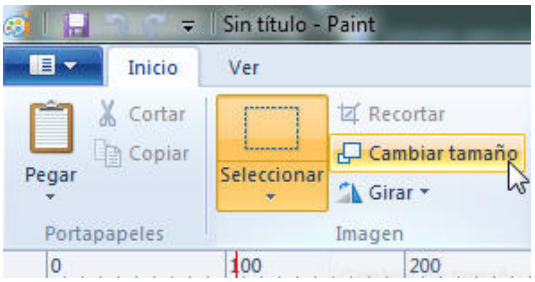
En OpenGl las dimensiones de una textura deben ser siempre potencia de 2 (n^2 tanto para el ancho como el alto de la imagen, otro valor causará que la imagen no sea cargada adecuadamente, distorsionando el color, o visualizando franjas sobre la imagen o en tonos de gris.
Como ejemplo de valores posible en potencia de dos para las resoluciones en píxeles podemos usar (los valores abajo relacionados son el resultado después de elevar valores al cuadrado):

1	49	169	361	625	961	1369
4	64	196	400	676	1024	1444
9	81	225	441	729	1089	1521
16	100	256	484	784	1156	1600
25	121	289	529	841	1225	1681
36	144	324	576	900	1296	1764

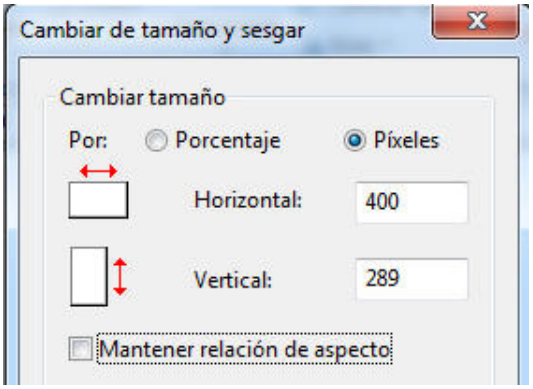
Si quisiera manejar una imagen que tuviera las mismas dimensiones de ancho como de alto, bastaría con repetir cualquiera de los valores, por ejemplo 100x100, 441x441, 676x676, 1600x1600, etc., si requieres imágenes rectangulares por ejemplo una imagen con la base mayor a la altura podemos usar cualquier combinación siempre y cuando tanto la base como la altura sean valore en potencia de dos (N^2). Para el desarrollo de un ejemplo se usará una imagen a la cual será necesario cambiar sus dimensiones para que cumpla lo arriba establecido, por lo que se tomarán las dimensiones 900x729.
Los pasos necesarios para mapear texturas sobre polígonos son los siguientes.
Buscamos una imagen.



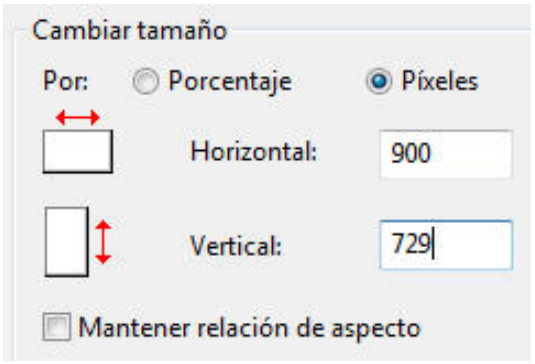
Copiamos o descargamos la imagen, podemos usar la aplicación de dibujo ***Paint*** de Microsoft, ejecutamos la aplicación y creamos un nuevo proyecto el cual modificaremos de tamaño para que reciba la imagen copiada anteriormente.



Indicamos el tamaño en pixeles y desmarcamos la casilla mantener relación de aspecto para poder indicar los valores de forma independiente.



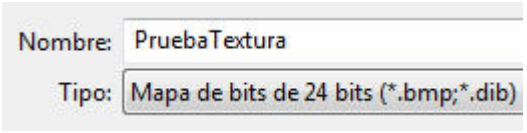
Para nuestro ejemplo tecleamos valores en potencia de 2 como ejemplo se indicará 900x729.



A continuación cargamos o copiamos la imagen y la ajustamos al tamaño del lienzo de trabajo. Al momento de ajustar, ya sea estirando o reduciendo es importante tomar en cuenta que debe ajustarse al ancho y alto del lienzo, ya que en caso de no hacerlo adecuadamente al momento de guardar los cambios se reajusta el tamaño de la imagen causando que las proporciones no correspondan a valores en potencias de dos (n^2).

Se ajusta la imagen

Se graba el archivo de imagen en formato BMP de 24 bits.



Con el propósito de entender el manejo de las rutas para ubicar el archivo dentro del programa en Python no se debe usar la contrabarra como indicador de separador de carpetas, en su lugar usaremos la barra normal, las rutas sugeridas en los ejemplos corresponden a las rutas en mi equipo, por lo que ustedes deben ser cuidadosos al momento del uso de las rutas para ubicar sus imágenes, por ejemplo mi ruta de trabajo es (/home/javier/Documentos/Programas/Python/Texturas/PruebaTexturas.jpg).

Error común. Debe especificar dentro de su programa la ruta para cargar la imagen mediante el uso de la barra normal (/), no la contrabarra (\ )

Forma correcta: `"C:/ZZZ_Sprites/PruebaTextura.bmp"`

Forma incorrecta: `"C:\ZZZ_Sprites\PruebaTextura.bmp"`

FUNCIÓN PARA CARGAR EL ARCHIVO DE IMAGEN BMP PROGRAMA

C05_ImagenEnPoligono01.py

```
                                        # Función para cargar textura
def cargar_textura(ruta):
    imagen = Image.open(ruta).transpose(Image.FLIP_TOP_BOTTOM)
    img_data = imagen.convert("RGBA").tobytes()
    width, height = imagen.size

    tex_id = glGenTextures(1)
    glBindTexture(GL_TEXTURE_2D, tex_id)

    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0,
                 GL_RGBA, GL_UNSIGNED_BYTE, img_data)

    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR)
    return tex_id
```

CARGAR LA IMAGEN AL POLÍGONO PARA REPRESENTAR LA IMAGEN.

C05_ImagenEnPoligono01.py

Una vez cargada la imagen en el arreglo debemos asignarla al polígono donde se representará.

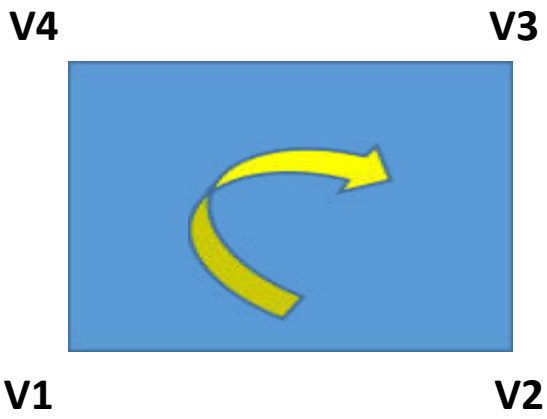
```
        # Seleccionar textura actual
        glBindTexture(GL_TEXTURE_2D, imagen_de_textura)

        # Dibujar el polígono con la textura de la imagen
        glBegin(GL_QUADS)
        glTexCoord2f(0, 0); glVertex2f( 0.1, -0.3)
        glTexCoord2f(1, 0); glVertex2f( 0.7, -0.3)
        glTexCoord2f(1, 1); glVertex2f( 0.7,  0.3)
        glTexCoord2f(0, 1); glVertex2f( 0.1,  0.3)
        glEnd()
```

ENTENDIENDO CÓMO SE ASIGNA LA IMAGEN DENTRO DEL POLÍGONO.

C05_ImagenEnPoligono01.py

Analizando la creación del siguiente cuadrado o polígono, nos podemos dar cuenta que se representa de la siguiente forma.

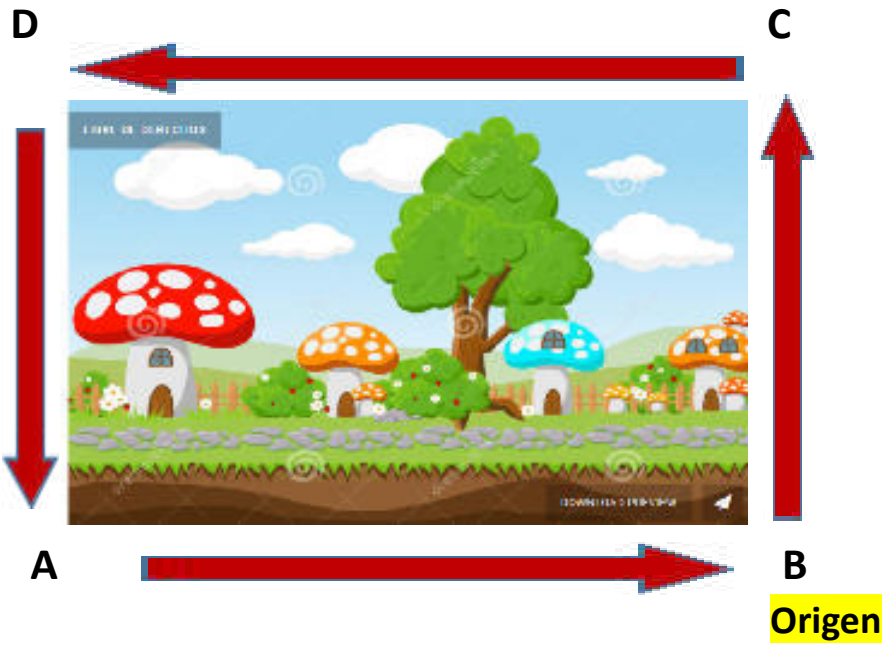
```
glBegin(GL_QUADS)
glVertex2f( 0.1, -0.3)
glVertex2f( 0.7, -0.3)
glVertex2f( 0.7,  0.3)
glVertex2f( 0.1,  0.3)
glEnd()
```



Para cargar la imagen consideraremos el siguiente orden para ser asignado posteriormente al polígono.

Coordenadas **glTextCoord2f** para la carga de una imagen normal, esto es, como la veríamos de forma normal.

A	**0**	**0**	Esquina inferior izquierda de la imagen original
B	**1**	**0**	Esquina inferior derecha de la imagen original
C	**1**	**1**	Esquina superior derecha de la imagen original
D	**0**	**1**	Esquina superior izquierda de la imagen original



El origen de referencia para cargar la imagen a partir de ahora comenzará siempre en la ***esquina inferior izquierda*** en sentido contrario de las manecillas del reloj.

TOMAR COMO REFERENCIA EL PROGRAMA **C05_ImagenEnPoligono02.py**

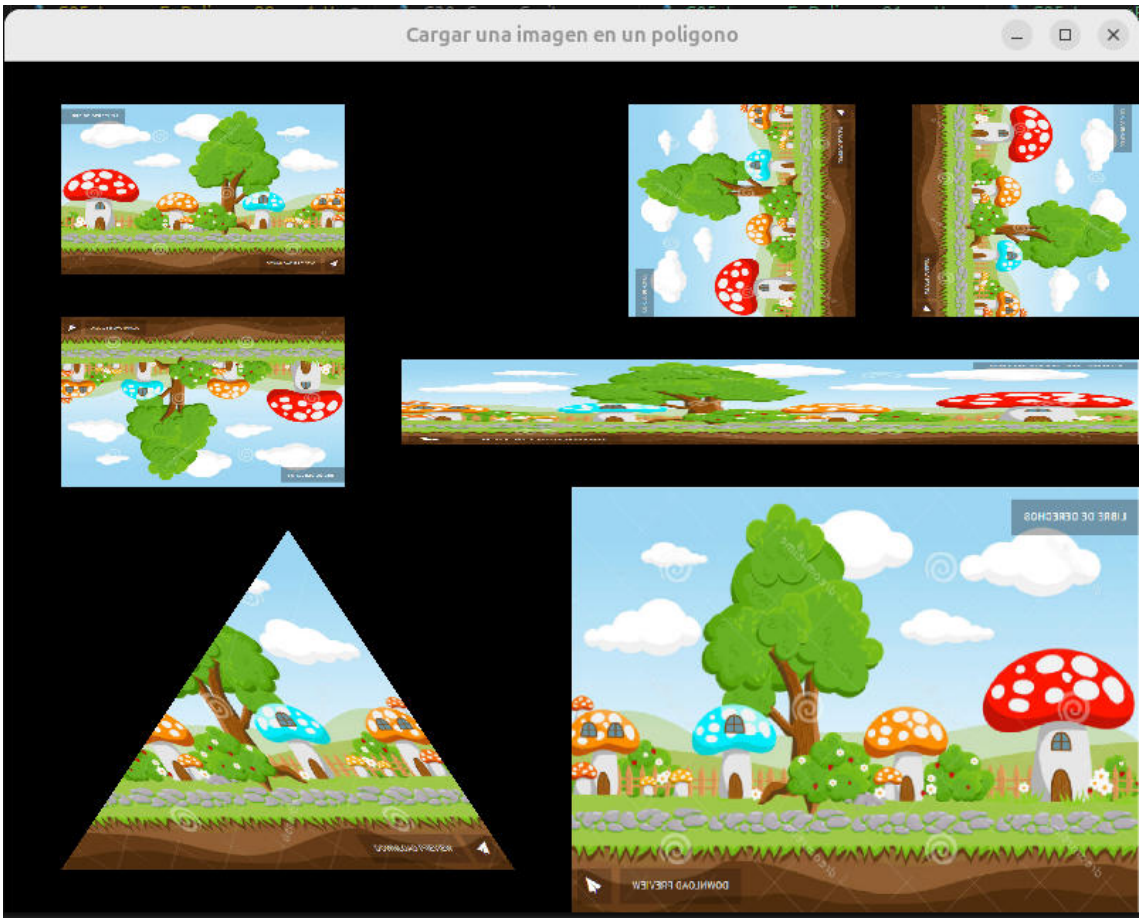
El código completo queda que la siguiente forma

```
import glfw
from OpenGL.GL import *
from PIL import Image
import time


                                    # Función para cargar textura
def cargar_textura(ruta):
    imagen = Image.open(ruta).transpose(Image.FLIP_TOP_BOTTOM)
    img_data = imagen.convert("RGBA").tobytes()
    width, height = imagen.size

    tex_id = glGenTextures(1)
    glBindTexture(GL_TEXTURE_2D, tex_id)

    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0,
                 GL_RGBA, GL_UNSIGNED_BYTE, img_data)

    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR)

    return tex_id


                                      # Programa principal GLFW
def main():

    if not glfw.init():                # Inicializar GLFW
        return

    ventana = glfw.create_window(800, 600, "Cargar una imagen en un
poligono", None, None)
    glfw.make_context_current(ventana)

    ruta =
"/home/javier/Documentos/Programas/Python/Texturas/PNGs/PruebaTextura.
bmp"
    imagen_de_textura = cargar_textura(ruta)

    # Configurar OpenGL
    glEnable(GL_TEXTURE_2D)

                                      # Ciclo de glfw
```

```
    while not glfw.window_should_close(ventana):

        glClear(GL_COLOR_BUFFER_BIT)

        # Seleccionar textura actual
        glBindTexture(GL_TEXTURE_2D, imagen_de_textura)

        # Dibujar el los poligonos con la textura de la imagen
       # Cuadro superior izquierdo
        glBegin(GL_QUADS)
        glTexCoord2f(0, 0); glVertex2f(-0.9, 0.5)
        glTexCoord2f(1, 0); glVertex2f(-0.4, 0.5)
        glTexCoord2f(1, 1); glVertex2f( -0.4, 0.9)
        glTexCoord2f(0, 1); glVertex2f(-0.9, 0.9)
        glEnd()

        # cuadro inferior izquierdo
        glBegin(GL_QUADS)
        glTexCoord2f(1, 1); glVertex2f(-0.9, 0.0)
        glTexCoord2f(0, 1); glVertex2f(-0.4, 0.0)
        glTexCoord2f(0, 0); glVertex2f( -0.4, 0.4)
        glTexCoord2f(1, 0); glVertex2f(-0.9, 0.4)
        glEnd()

        # cuadro uno superior derecho (cerca del centro)
        glBegin(GL_QUADS)
        glTexCoord2f(0, 1); glVertex2f(0.1, 0.4)
        glTexCoord2f(0, 0); glVertex2f(0.5, 0.4)
        glTexCoord2f(1, 0); glVertex2f(0.5, 0.9)
        glTexCoord2f(1, 1); glVertex2f(0.1, 0.9)
        glEnd()

        # cuadro dos superior derecho (en la orilla derecha)
        glBegin(GL_QUADS)
        glTexCoord2f(1, 0); glVertex2f(0.6, 0.4)
        glTexCoord2f(1, 1); glVertex2f(1.0, 0.4)
        glTexCoord2f(0, 1); glVertex2f(1.0, 0.9)
        glTexCoord2f(0, 0); glVertex2f(0.6, 0.9)
        glEnd()

        # cuadrote inferior derecho
        glBegin(GL_QUADS)
        glTexCoord2f(1, 0); glVertex2f(0.0, -1.0)
        glTexCoord2f(0, 0); glVertex2f(1.0, -1.0)
        glTexCoord2f(0, 1); glVertex2f(1.0, 0.0)
        glTexCoord2f(1, 1); glVertex2f(0.0, 0.0)
        glEnd()

        # el triangulo largo largo
        glBegin(GL_QUADS)
```

```
        glTexCoord2f(1, 0); glVertex2f(-0.3, 0.1)
        glTexCoord2f(0, 0); glVertex2f(1.0, 0.1)
        glTexCoord2f(0, 1); glVertex2f(1.0, 0.3)
        glTexCoord2f(1, 1); glVertex2f(-0.3, 0.3)
        glEnd()

        # El triangulo
        glBegin(GL_TRIANGLES)
        glTexCoord2f(0, 0); glVertex2f(-0.9, -0.9)
        glTexCoord2f(1, 0); glVertex2f(-0.1, -0.9)
        glTexCoord2f(1, 1); glVertex2f(-0.5, -0.1)
        glEnd()

        glfw.swap_buffers(ventana)
        glfw.poll_events()

    glfw.terminate()

main()
```

RESULTADO.



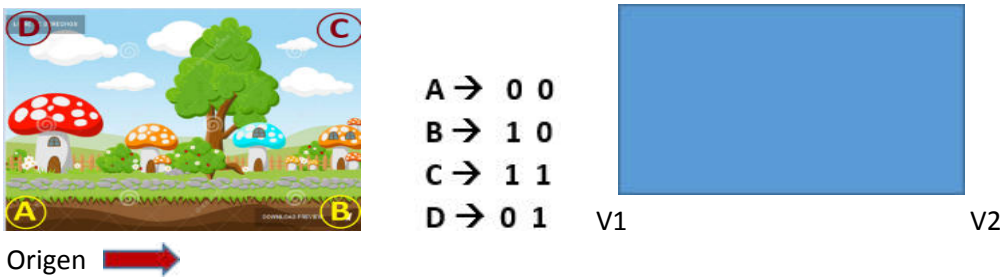
Lo que tenemos que hacer ahora es asignar el orden de carga de la imagen dentro del polígono.

Está línea define la imagen que vamos a trabajar

glBindTexture(GL_TEXTURE_2D, imagen_de_textura)

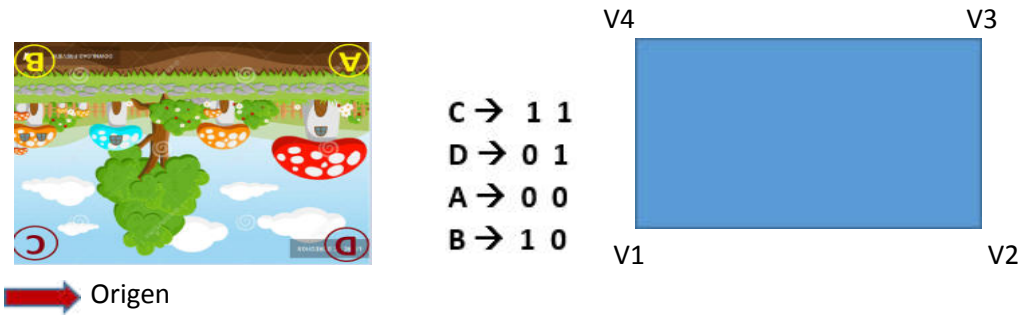
FORMA 1. Carga de la imagen de forma normal dentro de un rectángulo (lado izquierdo pantalla)

```
glBegin(GL_QUADS)
glTexCoord2f(0, 0); glVertex2f(-0.9, 0.5)
glTexCoord2f(1, 0); glVertex2f(-0.4, 0.5)
glTexCoord2f(1, 1); glVertex2f( -0.4, 0.9)
glTexCoord2f(0, 1); glVertex2f(-0.9, 0.9)
glEnd()
```



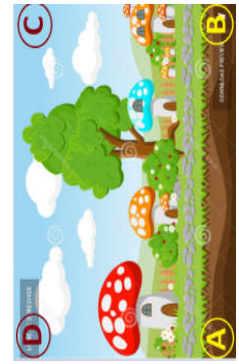
FORMA 2. Cargar la imagen invertida o de cabeza.

```
glBegin(GL_QUADS)
glTexCoord2f(1, 1); glVertex2f(-0.9, 0.0)
glTexCoord2f(0, 1); glVertex2f(-0.4, 0.0)
glTexCoord2f(0, 0); glVertex2f( -0.4, 0.4)
glTexCoord2f(1, 0); glVertex2f(-0.9, 0.4)
glEnd()
```



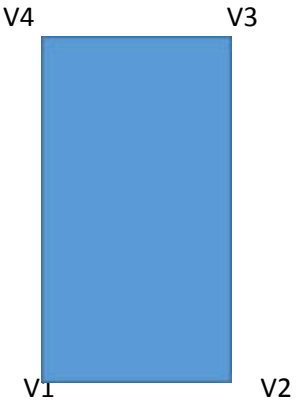
FORMA 3. Cargar la imagen del lado derecho (cerca del centro).

```
glBegin(GL_QUADS)
glTexCoord2f(0, 1); glVertex2f(0.1, 0.4)
glTexCoord2f(0, 0); glVertex2f(0.5, 0.4)
glTexCoord2f(1, 0); glVertex2f(0.5, 0.9)
glTexCoord2f(1, 1); glVertex2f(0.1, 0.9)
glEnd()
```



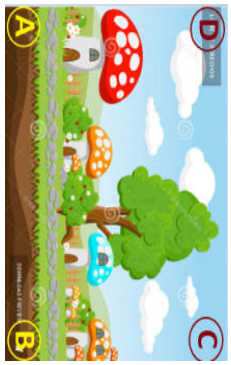
Origen

D → 0 1
A → 0 0
B → 1 0
C → 1 1



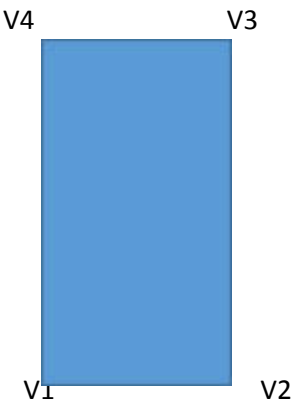
FORMA 4. Cargar la imagen del lado derecho.

```
glBegin(GL_QUADS)
glTexCoord2f(1, 0); glVertex2f(0.6, 0.4)
glTexCoord2f(1, 1); glVertex2f(1.0, 0.4)
glTexCoord2f(0, 1); glVertex2f(1.0, 0.9)
glTexCoord2f(0, 0); glVertex2f(0.6, 0.9)
glEnd()
```

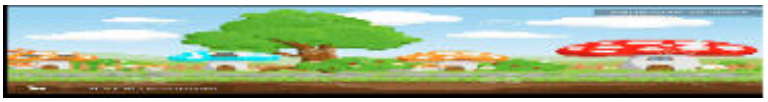


Origen

B → 1 0
C → 1 1
D → 0 1
A → 0 0



FORMA 5. Cargar la imagen normal pero invertida en el rectángulo largo de la pantalla

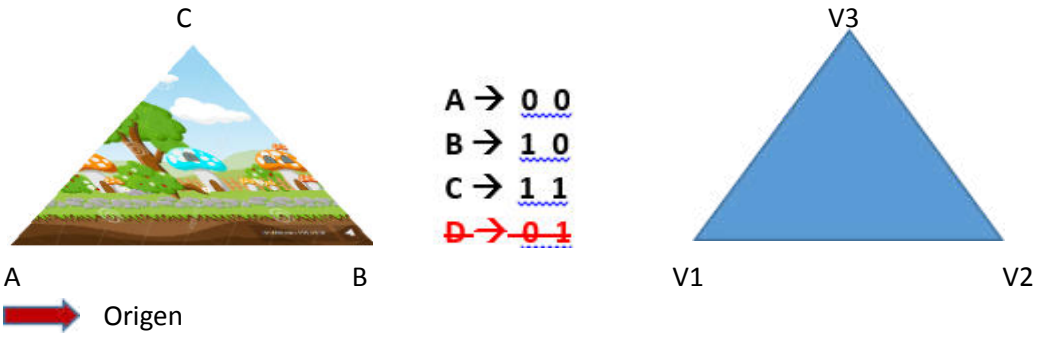


B → 1 0
A → 0 0
D → 0 1
C → 1 1

```
glBegin(GL_QUADS)
glTexCoord2f(1, 0); glVertex2f(-0.3, 0.1)
glTexCoord2f(0, 0); glVertex2f(1.0, 0.1)
glTexCoord2f(0, 1); glVertex2f(1.0, 0.3)
glTexCoord2f(1, 1); glVertex2f(-0.3, 0.3)
glEnd()
```

Una vez entendido cómo se debe cargar la imagen en el polígono, como ejemplo adicional se cargará la imagen en un triángulo.

```
glBegin(GL_TRIANGLES)
glTexCoord2f(0, 0); glVertex2f(-0.9, -0.9)
glTexCoord2f(1, 0); glVertex2f(-0.1, -0.9)
glTexCoord2f(1, 1); glVertex2f(-0.5, -0.1)
glEnd()
```

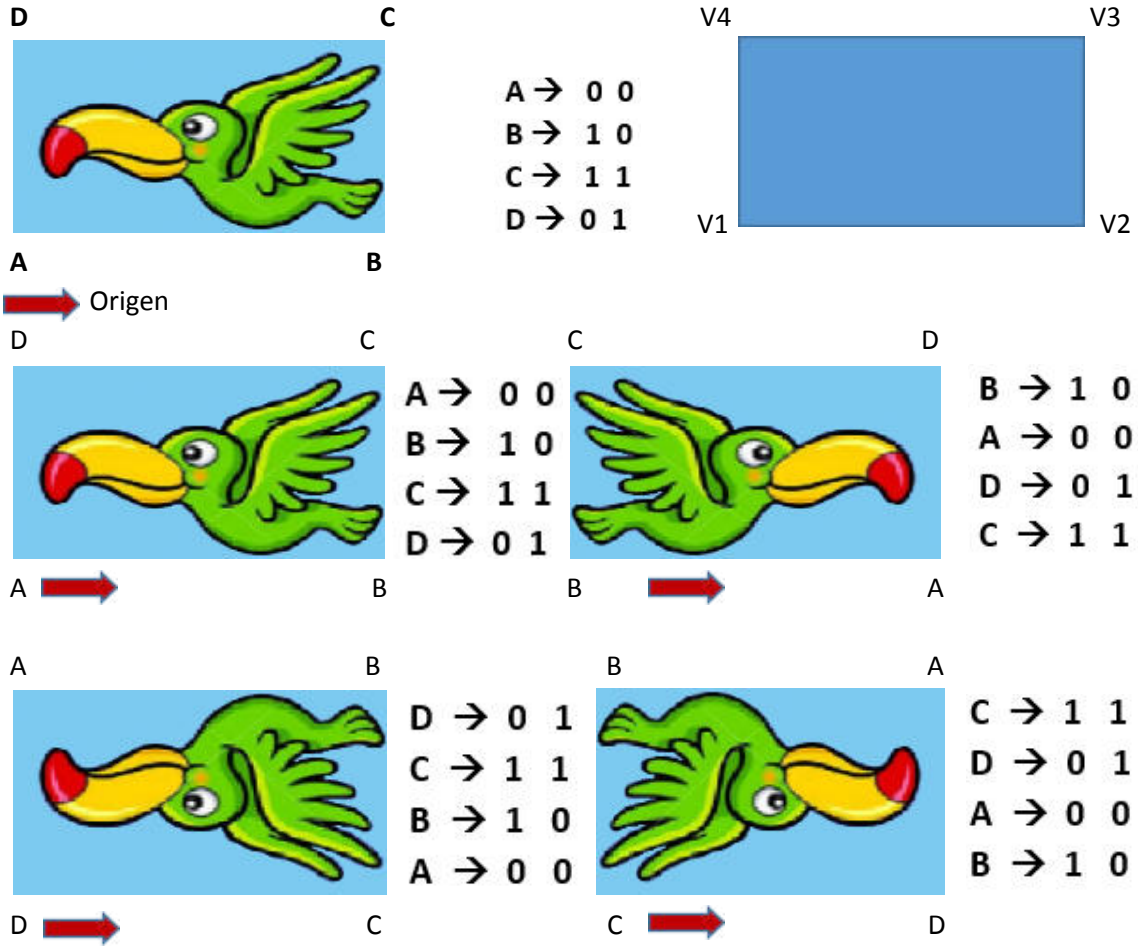


Como podemos observar al cancelar la referencia “D” de la imagen, no permite apreciar los detalles de ese lado.

OTRO EJEMPLO PARA LA CARGA DE LA IMAGEN.

Otro ejemplo para cargar una imagen y las variantes que podemos presentar para cargarla en el polígono y mostrarla en pantalla.

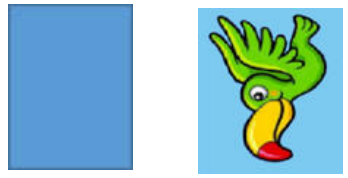
Forma 1. Carga normal de la imagen (Es este caso el dibujo original está viendo hacia la izquierda) y la manera en que está definido el orden de los vértices del polígono para la carga, a partir de aquí lo único que cambiará serán las coordenadas **glTextCoord2f** para cargar la imagen de acuerdo a los vértices del polígono.



En los siguientes ejemplos de carga de la imagen, es conveniente ajustar la forma del polígono a la forma de la imagen, ejemplo, si la base de la imagen original es más ancha que la altura y se va a rotar 90 grados, el resultado final es una imagen donde ahora la altura es mayor que la base, lo cual provocará que la imagen se vea muy ancha, ya que la imagen se ajusta a la forma del polígono.

V4 V3 V4 V3

V1 V2



V1 V2

C B

D → 0 1
A → 0 0
B → 1 0
C → 1 1

D A

C → 1 1
B → 1 0
A → 0 0
D → 0 1

D → A

C → B

B C

A → 0 0
D → 0 1
C → 1 1
B → 1 0

A → D

A D

B → 1 0
C → 1 1
D → 0 1
A → 0 0

B → C

En conclusión cuando se está dibujando el objeto, hay que indicar, para cada vértice de este, qué posición de la textura le corresponde. Esto se hace mediante la siguiente función, donde (s,t) indica una posición sobre el mapa de la imagen.

```
glTexCoord2f( GLfloat s, GLfloat t);
```

Lo que se debe hacer siempre es indicar la coordenada de la textura antes de indicar el vértice del polígono.

CÓDIGO PARA CARGAR LA IMAGEN COMO FONDO DE LA PANTALLA.

El código completo para cargar la textura queda de la siguiente manera.
IMPORTANTE!!! No olvidar modificar la ruta dónde se encuentra el archivo de imagen dentro de su computadora.

```
import glfw
from OpenGL.GL import *
from PIL import Image
import time


                                # Función para cargar textura
def cargar_textura(ruta):
    imagen = Image.open(ruta).transpose(Image.FLIP_TOP_BOTTOM)
    img_data = imagen.convert("RGBA").tobytes()
    width, height = imagen.size

    tex_id = glGenTextures(1)
    glBindTexture(GL_TEXTURE_2D, tex_id)

    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0,
                 GL_RGBA, GL_UNSIGNED_BYTE, img_data)

    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR)

    return tex_id

                                  # Programa principal GLFW
def main():

    if not glfw.init():           # Inicializar GLFW
        return

    ventana = glfw.create_window(800, 600, "Cargar una imagen en un
poligono", None, None)
    glfw.make_context_current(ventana)

    ruta =
"/home/javier/Documentos/Programas/Python/Texturas/PNGs/PruebaTextura.bm
p"
    imagen_de_textura = cargar_textura(ruta)

    # Configurar OpenGL
    glEnable(GL_TEXTURE_2D)

                                  # Ciclo de glfw
```

```
    while not glfw.window_should_close(ventana):

        glClear(GL_COLOR_BUFFER_BIT)

        # Seleccionar textura actual
        glBindTexture(GL_TEXTURE_2D, imagen_de_textura)

        # Dibujar el los poligonos con la textura de la imagen
        glBegin(GL_QUADS)
        glTexCoord2f(0, 0); glVertex2f(-1.0, -1.0)
        glTexCoord2f(1, 0); glVertex2f(1.0, -1.0)
        glTexCoord2f(1, 1); glVertex2f(1.0, 1.0)
        glTexCoord2f(0, 1); glVertex2f(-1.0, 1.0)
        glEnd()

        glfw.swap_buffers(ventana)
        glfw.poll_events()

    glfw.terminate()


main()
```

RESULTADO: