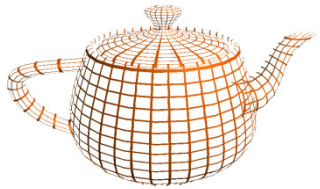
Computação Gráfica

Primitivas OpenGL



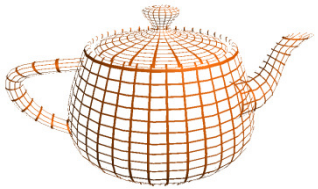
Modelação 3D

- Definição de um ponto em 3D

```
glVertex3f(x,y,z);
glVertex3fv(a); // sendo float a[3];
```

- Para desenhar pontos fazer:

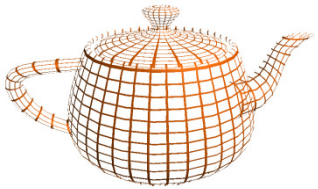
```
glBegin(GL_POINTS);
        glVertex3f(0.0f, 0.0f, 10.0f);
        glVertex3f(1.0f, 1.0f, 1.0f);
glEnd();
```



Modelação 3D

Opções para glBegin:

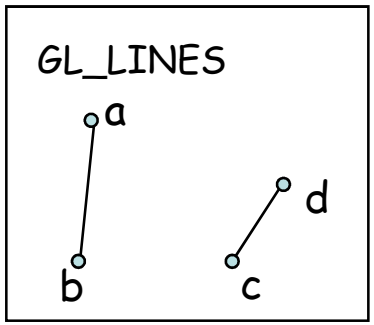
- GL_POINTS
- GL_LINES
- GL_LINE_STRIP
- GL_LINE_LOOP
- GL_TRIANGLES
- GL_TRIANGLE_STRIP
- GL_TRIANGLE_FAN
- GL_QUADS
- GL_QUAD_STRIP
- GL_POLYGON

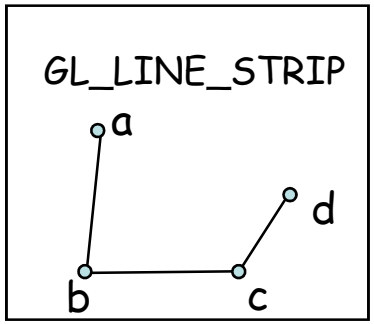


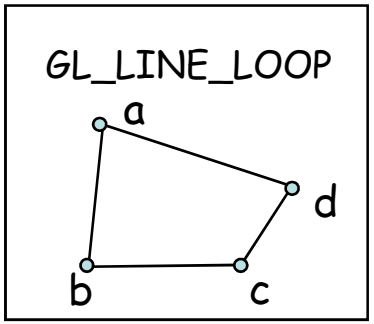
OpenGL - Primitivas

- Linhas

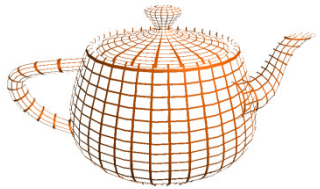
```
glBegin(...)
     glVertex3fv(a);
     glVertex3fv(b);
     glVertex3fv(c);
     glVertex3fv(d);
glEnd();
```







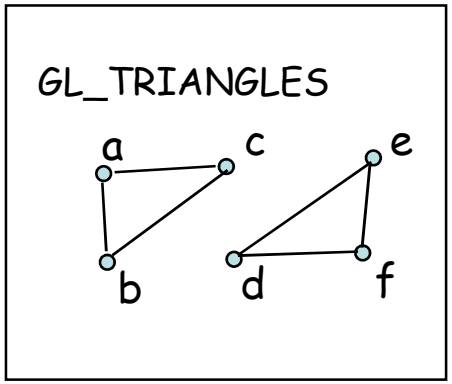
Nota: os pontos a,b,c,d estão a ser representados por vectores de floats

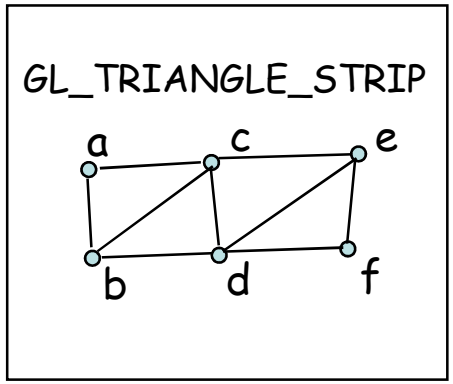


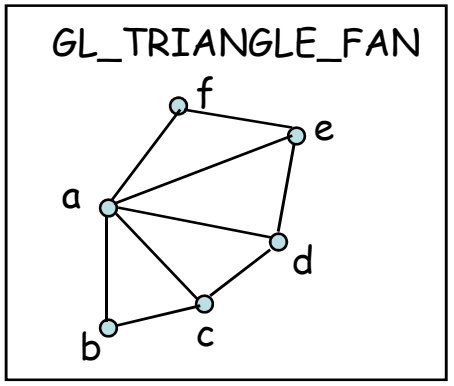
OpenGL - Primitivas

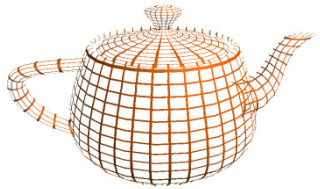
- Triângulos

```
glBegin(...)
     glVertex3fv(a);
     glVertex3fv(b);
     glVertex3fv(c);
     glVertex3fv(d);
     glVertex3fv(e);
     glVertex3fv(f);
glEnd();
```





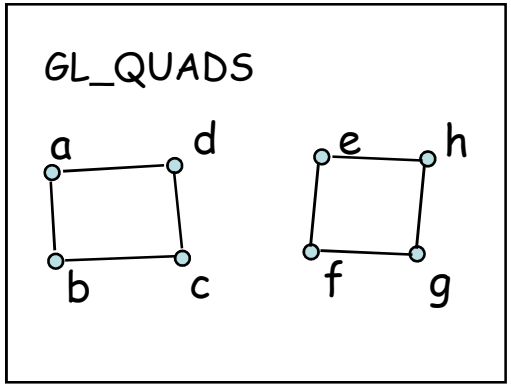


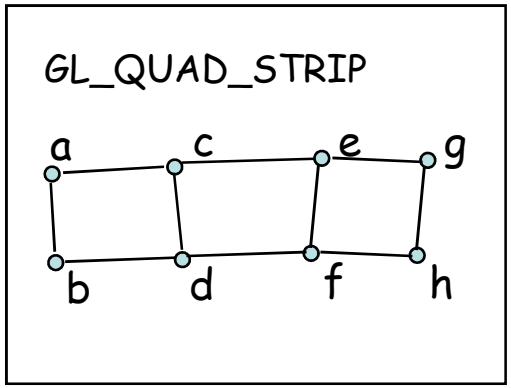


OpenGL - Primitivas

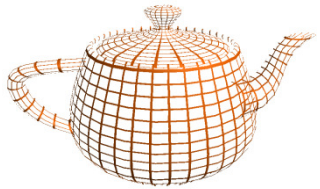
- Quads

```
glBegin(...)
     glVertex3fv(a);
     glVertex3fv(b);
     glVertex3fv(c);
     glVertex3fv(d);
     glVertex3fv(e);
     glVertex3fv(f);
     glVertex3fv(g);
     glVertex3fv(h);
glEnd();
```





Nota: atenção à ordem dos vértices no diagrama



Exercício

- Implementar uma função que desenhe um prisma:
  - A função deve receber como parâmetros a altura, raio e número de lados do prisma
  - Utilizar *triangle strips* e *fans*
- Definir normais apropriadas para os vértices
- Iluminar com uma luz direccional