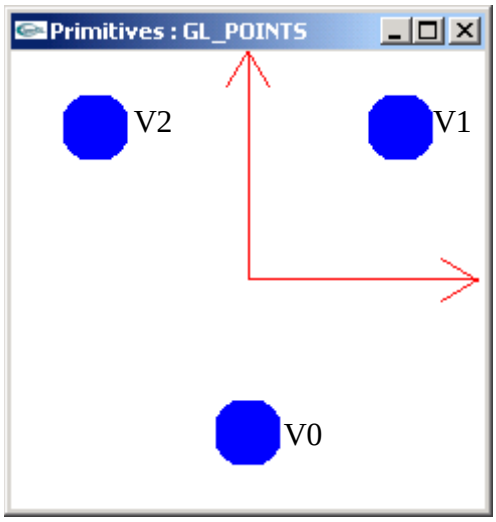
Infographie, exercice série 2

En utilisant le canevas de la série 1, écrire un programme qui affiche les 10 primitives de dessin d'OpenGL.
Dans les dessins ci-dessous, les axes ont une longueur de 30 unités.



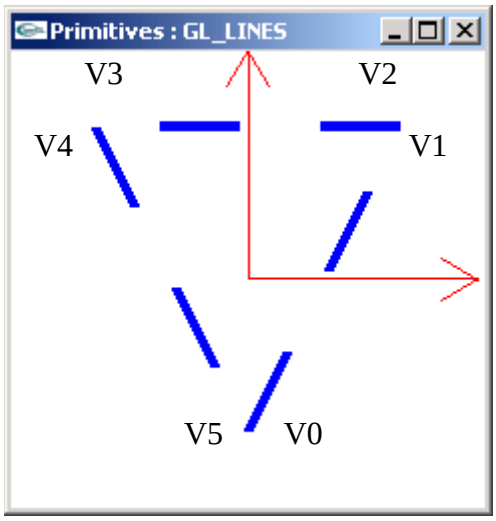
GL_POINTS

La taille et la forme des points sont spécifiées par :

```
GLfloat sizes[2];
GLfloat step;
GLfloat curSize;
// Trouver la taille la plus petite et la plus grande des points
glGetFloatv(GL_POINT_SIZE_RANGE, sizes);
// Trouver le pas minimum entre deux tailles
glGetFloatv(GL_POINT_SIZE_GRANULARITY, &step);
// La taille courante est le milieu des deux tailles extrêmes
curSize = (sizes[1] - sizes[0]) / 2.0f;
// Les points sont dessinés sous forme de cercles
glEnable(GL_POINT_SMOOTH);
```

Puis par :

```
glPointSize(curSize);
```



GL_LINES

L'épaisseur et le style des lignes (ici en traitillé) sont spécifiés par les instructions suivantes :

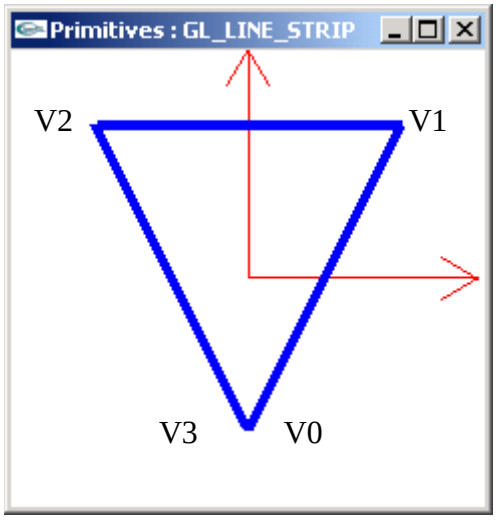
```
GLfloat sizes[2];
GLfloat step;
GLfloat curWidth;
// Trouver la largeur la plus petite et la plus grande des lignes
glGetFloatv(GL_LINE_WIDTH_RANGE, sizes);
// Trouver le pas minimum entre deux largeurs
glGetFloatv(GL_LINE_WIDTH_GRANULARITY, &step);
// L'implémentation de Microsoft accepte sizes entre 0.5 et 10.0
// Le pas le plus petit est de 0.125
// La largeur courante est le milieu des deux largeurs extrêmes
curWidth = (sizes[1] - sizes[0]) / 2.0f;

// Type de lignes (Stipple)
GLint factor = 5;  // 5 pixels par bit
GLushort pattern = 0x00FF;  // Chaque bit à 1 dessine le pixel (16 bits)
glLineStipple(factor, pattern);
```

Puis :

```
glLineWidth(curWidth);     // Largeur de ligne curWidth
glEnable(GL_LINE_STIPPLE); // Activer le type de ligne
```

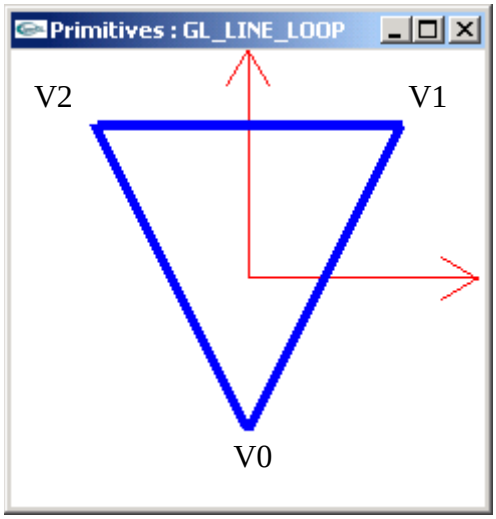
Pour désactiver le type de ligne, on utilise :

```
glDisable(GL_LINE_STIPPLE); // Désactiver le type de ligne
```



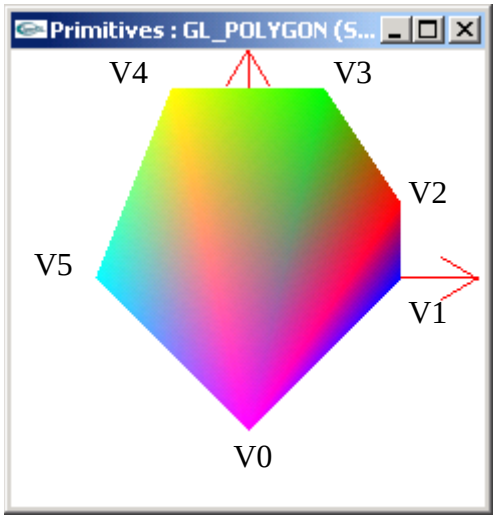
GL_LINE_STRIP

V3 est à la même position que V0



GL_LINE_LOOP

La figure est refermée (pas de V3 !)



GL_POLYGON

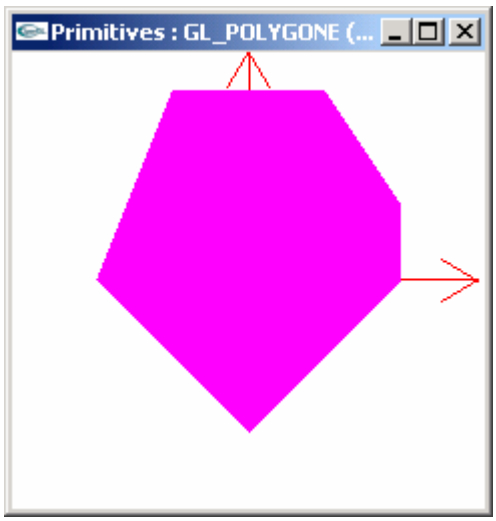
Pour interpoler la couleur entre les différents sommets (vertexes), on utilise :

Chaque sommet a une couleur différente.

```
glShadeModel(GL_SMOOTH);
```

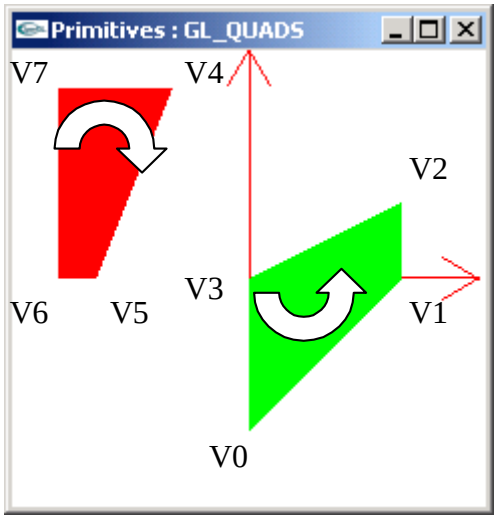
Par opposition à :

```
glShadeModel(GL_FLAT);
```



GL_POLYGON

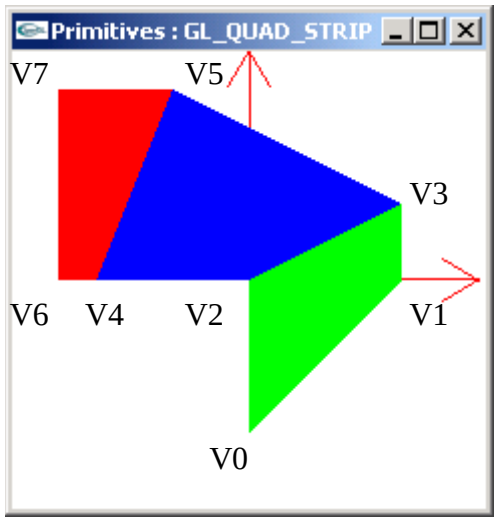
Ici, la couleur est celle du sommet V0.



GL_QUADS

La couleur est celle du dernier sommet de chaque quadrilatère (V3 et V7).

Les sommets du premier quadrilatère tournent dans le sens contraire des aiguilles d'une montre.
Les sommets du deuxième quadrilatère tournent dans le sens des aiguilles d'une montre.
Ceci aura une importance dans la suite !

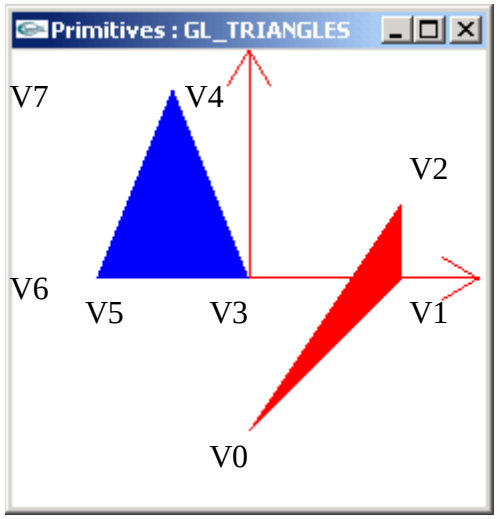


GL_QUAD_STRIP

La couleur de chaque quadrilatère est celle du dernier sommet de chacun d'eux (V3, V5 et V7).

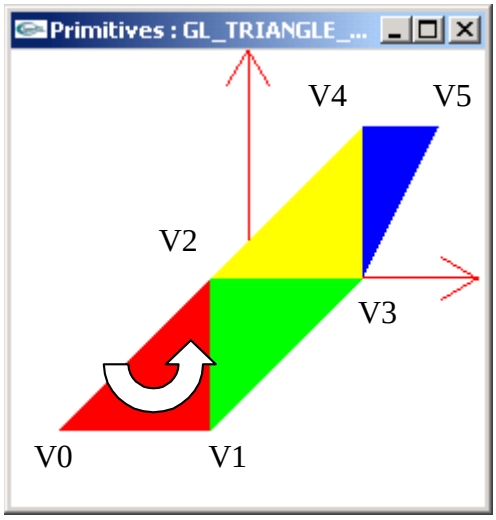
L'ordre de connexion est :

V0, V1, V3, V2 pour le premier,
V2, V3, V5, V4 pour le deuxième,
V4, V5, V7, V6 pour le troisième.



GL_TRIANGLES

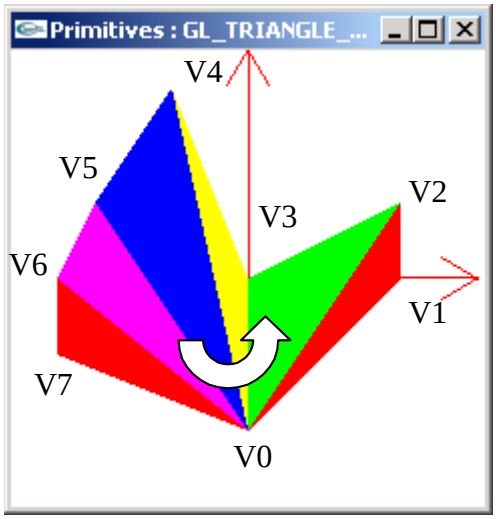
Si le nombre de points n'est pas un multiple de trois, les derniers (ici V6 et V7) sont ignorés.
La couleur rouge est celle de V2.
La couleur bleue est celle de V5.



GL_TRIANGLE_STRIP

Chaque fois qu'on ajoute un sommet, un nouveau triangle est dessiné avec les deux sommets précédents.

L'orientation est déterminée par le premier triangle (V0, V1, V2).



GL_TRIANGLE_FAN

Tous les triangles ont leur sommet en V0.