TP N°6
Gestion des évènements

Dessiner un disque :

On va utilisé glDrawArrays avec comme argument GL_TRIANGLE_FAN au lieu de GL_TRIANGLES. GL_TRIANGLE_FAN permet de dessiner des triangles connectés l'un à l'autre ayant tous un vertex en commun qui est au centre.
On définit la liste des vertex et on les met dans un tableau grâce à une boucle. Le vertex central sera le 1er à être inséré dans le tableau :

```
double pi=3.14, x, y;
float rayon = 1.5;
int  numPoints = 100 ;
int indice = 1;
STRVertex g_vertex_buffer_data[numPoints+1];
g_vertex_buffer_data[0] = {vec3(0.0f, 0.0f, 0.0f),vec3(0.0f, 0.0f, 1.0f)};
for(int i=0; i<numPoints ; i++)
  {
     x= rayon * cos(i * 2 * pi / numPoints);
     y= rayon * sin(i * 2 * pi / numPoints);
     g_vertex_buffer_data[indice] = {vec3(x, y, 0.0f), vec3(0.0f, 0.0f, 1.0f)};
     indice++;
  }
```

Est ce que le disque obtenu est parfait ? Corriger la forme et utiliser l'instruction :
`glPolygonMode (GL_FRONT_AND_BACK, GL_LINE);`
afin de voir chaque triangle.

Gestion des évènements

Glut permet de gérer les évènements liés au clavier et à la souris :

- glutKeyboardFunc(KeyPressed) : fait référence à la fonction « KeyPressed » qui sera appelée lorsqu'un évènement clavier est détecté.
Le prototype de « KeyPressed » : `void KeyPressed (unsigned char touche, int x, int y)`

- glutKeyboardUpFunc(KeyPressed) : s'utilise comme glutKeyboardFunc(), sauf qu'elle est déclenchée par l'évènement de relâcher la touche clavier.

- glutSpecialFunc(SpecialKey) et glutSpecialUpFunc(SpecialKey) : gèrent les touches spéciales, telles que les boutons de fonction et les boutons de flèches de direction : GLUT_KEY_DOWN, GLUT_KEY_LEFT, GLUT_KEY_RIGHT, GLUT_KEY_UP.
Le prototype de «SpecialKey»: `void SpecialKey (int k, int x, int y)`

- glutMouseFunc(Mouse) : elle fait référence à la fonction qui sera appelée lorsqu'un bouton souris est appuyé. Celui ci peut prendre les valeurs : GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON ou GLUT_RIGHT_BUTTON. *etat* est l'état du bouton : GLUT_DOWN ou GLUT_UP.
Le prototype: `void Mouse (int bouton, int etat, int x, int y)`

- void glutMotionFunc(MouseMotion) : s'exécutent automatiquement lorsque la souris se déplace avec un bouton appuyé.

- void glutPassiveMotionFunc(MouseMotion) : s'exécutent lorsque la souris se déplace sans qu'un bouton ne soit appuyé.
Le prototype de MouseMotion : void MouseMotion (int x, int y)

Exemples :
1-
```
void KeyPressed (unsigned char touche, int x, int y)
{
   switch (touche)
   {
      case 'r' :
         glClearColor(1.0f, 0.0f, 0.0f, 0.5f);
         glClear(GL_COLOR_BUFFER_BIT );
         break;
      case 'b':
         glClearColor(0.0f, 0.0f, 1.0f, 0.5f);
         glClear(GL_COLOR_BUFFER_BIT );
         break;
      case 'v':
         glClearColor(0.0f, 1.0f, 0.0f, 0.5f);
         glClear(GL_COLOR_BUFFER_BIT );
         break;
      case 27: /* Escape */ exit(0);
           break;
      default:
         glClearColor(0.0f, 0.0f, 0.0f, 0.5f);
         glClear(GL_COLOR_BUFFER_BIT );
   }
   glutPostRedisplay();  /// cette instruction va forcer glut à rafraîchir l'affichage.
}
```
2-
```
 void MouseMotion (int x, int y)
{
   diffX = xOld - x;
   diffY = yOld - y;
   glutPostRedisplay();
   xOld = x;
   yOld = y;
}
```
Utiliser la variable diffX pour appliquer une rotation du disque en suivant les mouvements de la souris.

Exercices :

- Construire un cylindre placé le long de l'axe des Z, une extrémité (un disque) sur le plan (OXY) et l'autre de profondeur différente. Utiliser GL_TRIANGLE_STRIP comme argument de glDrawArrays. GL_TRIANGLE_STRIP permet de dessiner des triangles connectés, chacun partage deux vertex avec le suivant : {0, 1, 2} {1, 2, 3} {2, 3, 4} ...
- Utiliser les boutons flèche du clavier (ou les bouton souris) pour faire agrandir et diminuer la taille de l'objet dessiné.