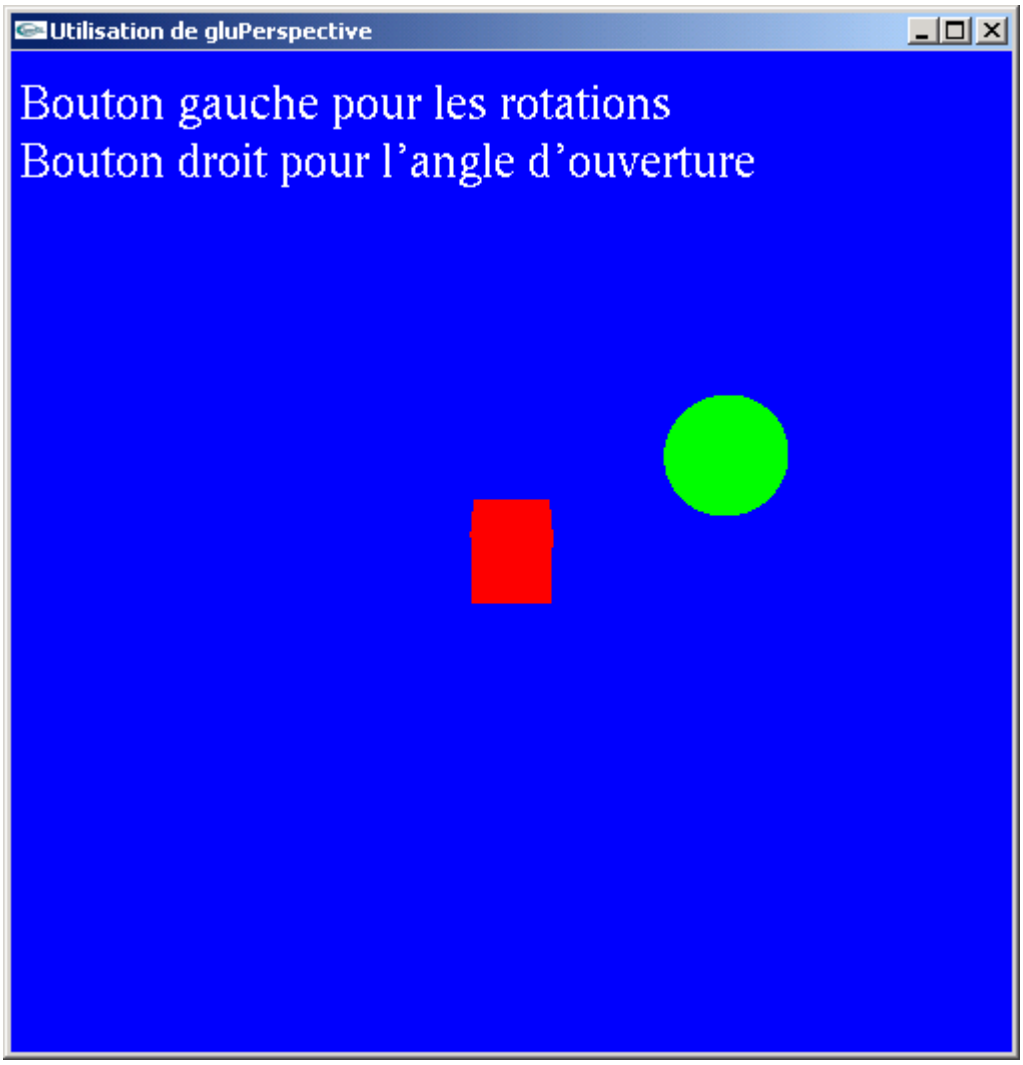
Infographie, exercice série 5

Ecrire un programme qui affiche un cube au centre et une sphère qui tourne (à l'aide du bouton gauche de la souris) autour du cube à une distance de 5.
La caméra est placée à 10 le long de l'axe Z, et à 5 le long de l'axe Y (donné par la variable globale yCamera).



On demande, en fait de compléter les fonctions suivantes (pensez à activer le Depth Buffer) :

- myinit()
- DessinerLaScene()
- PositionnerCamera(GLfloat x, GLfloat y, GLfloat z)
- Display()

Dans le programme distribué, le texte s'affiche par l'appel à la fonction : *AfficherLeTexte()*
Un déplacement de souris avec le bouton droit enfoncé modifie l'angle d'ouverture.

Les fonctions affichant le texte ont la teneur suivante (dans le fichier *Geometrie.cpp*) :

```
void AfficherLeTexte()
{
  // Afficher le texte :
  glColor3f(1.0f, 1.0f, 1.0f);  // Couleur blanche pour le texte
  glMatrixMode(GL_PROJECTION);  // On garde la même projection pour le texte
  glPushMatrix();           // Sauvegarder la matrice de projection actuelle
  glLoadIdentity();         // Charger la matrice identité
  gluPerspective(60.0, RapportAspect, 1.0, 10.0);

  glMatrixMode(GL_MODELVIEW);   // On ne fait pas de rotation pour le texte
  glPushMatrix();       // Sauvegarder la matrice de visualisation actuelle
  glLoadIdentity();     // Charger la matrice identité
  gluLookAt(0, 0, 3, 0, 0, 0, 0, 1, 0);
  drawstr(-1.7, 1.5, 0.0, "Bouton gauche pour les rotations");
  drawstr(-1.7, 1.3, 0.0, "Bouton droit pour l'angle d'ouverture");
  glMatrixMode(GL_PROJECTION);
  glPopMatrix();                // Restaurer la matrice de projection
  glMatrixMode(GL_MODELVIEW);
  glPopMatrix();                // Restaurer la matrice de visualisation
}

// Afficher la chaîne string dans le viewport
void drawstr(GLdouble x, GLdouble y, GLdouble z, char * string)
{
  int len, i;
  glRasterPos3d(x, y, z);  // Positionner le curseur
  len = (int)strlen(string);
  for(i = 0; i < len; i++)
    glutBitmapCharacter(font_style, string[i]);
}
```

La variable *font_style* est déclarée globale de la manière suivante (dans le fichier *Global.cpp*) :

```
GLvoid * font_style = GLUT_BITMAP_TIMES_ROMAN_24;
```

Les fonctions de réponse aux événements souris ont la teneur suivante (dans le fichier *Evenements.cpp*) :

```
void Mouse(int button, int state, int x, int y)
{
  if(button == GLUT_LEFT_BUTTON)
    if(state == GLUT_DOWN)      // Le bouton gauche a été pressé
    {
      ButtonLeftDown = true;
      xInit = x;
      yInit = y;
    }
    else                        // Le bouton gauche a été relâché
    {
      ButtonLeftDown = false;
      RotationOn = false;
    }
  else if(button == GLUT_RIGHT_BUTTON)
    if(state == GLUT_DOWN)      // Le bouton droit a été pressé
    {
      ButtonRightDown = true;
    }
```

```
    else                          // Le bouton droit a été relâché
    {
      ButtonRightDown = false;
    }
}

void Motion(int x, int y) // La souris s'est déplacée
{
  if(ButtonLeftDown) // Le bouton gauche est en bas
  {
    RotationOn = true;
    if((xInit - x) > 0)
    {
      yRot -= Coef;
      if(yRot <= -360) yRot = 0;
    }
    else if((xInit - x) < 0)
    {
      yRot += Coef;
      if(yRot >= 360) yRot = 0;
    }
    xInit = x;
    Display();
  }
  if(ButtonRightDown)  // Le bouton droit est en bas
  {
    int DeltaAngle;
    static int yOld = 0;
    DeltaAngle = yOld - y;
    if(DeltaAngle > 0)
    {
      AngleOuverture++;
      if(AngleOuverture >= 180.0) AngleOuverture = 179.0;
    }
    else
    {
      AngleOuverture--;
      if(AngleOuverture <= 0.0) AngleOuverture = 1.0;
    }
    yOld = y;

    // On modifie la matrice de projection
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(AngleOuverture, RapportAspect, PlanAvant, PlanArriere);

    // La matrice de visualisation devient la matrice courante
    glMatrixMode(GL_MODELVIEW);
    Display();
  }
}
```