Compte rendu TP1

Question 1 :

A quoi servent les classes MainWidget et GeometryEngine?

La classe MainWidget permet de gérer la fenêtre principale de l'application. Elle initialise OpenGL et permet de gérer les évènements (clavier, souris, temps…) ainsi que la partie graphique.
A noter que cette classe est obsolète depuis QT4, on lui préfère la classe QMainWindow.

La classe GeometryEngine permet de générer différentes figures géométriques, en l'occurrence un cube. Elle spécifie ses dimensions, ses coordonnées dans l'espace ainsi que sa texture.

A quoi servent les fichiers fshader.glsl et vshader.glsl?

Les shaders servent à programmer le pipeline de rendu par défaut de la carte graphique. Leur compilation est effectuée lors du lancement de l'application et l'exécution se passe au niveau du GPU. Ici, le fichier fshader.glsl gère les fragments shaders, aussi appelés les pixels shaders.
Ils permettent de traiter le rendu de chaque pixel qui s'affichera à l'écran.

Le fichier vshader.glsl gère les vertex shaders qui interviennent lors du traitement de chaque sommet. Dans les vertex shaders, on peut modifier le calcul des différents attributs de nos sommets.

Question 2 :

Expliquer le fonctionnement des deux méthodes
void GeometryEngine::initCubeGeometry() et
*void GeometryEngine::drawCubeGeometry(QOpenGLShaderProgram *program)*

La méthode initCubeGeometry spécifie les coordonnées du cube en déclarant les vecteurs des différentes faces et les indices correspondants aux textures de chacune des faces, sans oublier d'allouer l'espace mémoire nécessaire dans la carte graphique à l'aide des VBO.

La méthode drawCubeGeometry indique à OpenGL quels VBO utiliser, comment localiser les vecteurs de positions et de textures, puis de dessiner la figure géométrique à l'aide de ces données.

Question 3 :

*En vous inspirant des deux méthodes précédentes, écrivez les méthodes permettant d'initialiser et d'afficher une surface plane (16*16 sommets) composée de triangles. Ensuite créer une fonction pour calculer les 16*16 sommets et faces.*

Tout d'abord, il faut calculer le nombre de sommets nécessaires à OpenGL pour dessiner toutes les faces à l'aide de triangles. Une fois ce nombre déterminé, il faut calculer les coordonnées des différentes faces puis les indices correspondants aux différents triangles qui vont nous permettre de dessiner ces faces.
Problème rencontré : disparition de certaines faces.
Solution trouvée : éloigner suffisamment la caméra et augmenter l'espace mémoire alloué.

Question 4 :

Modifier l'altitude (z) des sommets pour réaliser un relief.
Déplacer la caméra à hauteur fixe au-dessus du terrain.
Utiliser le clavier pour avancer, reculer, et déplacer la caméra de gauche à droite.

Pour réaliser un relief, il faut ajouter un aspect aléatoire aux coordonnées des sommets sur l'axe z.
Afin de déplacer la caméra sur un axe donné, on réalise une translation sur l'axe correspondant.
Note : J'ai également ajouté un événement sur la molette de la souris pour déplacer la caméra en altitude (z) afin de mieux observer le terrain.

Bonus :

Jouer avec la lumière.

Pour créer une lumière de type soleil, j'ajouterai une seule source lumineuse quelque part dans mon monde. Je ferai en sorte qu'elle éclaire suffisamment l'objet qui m'intéresse dans la scène et qu'on puisse voir les ombres et/ou les reflets.
Pour créer des lumières localisées, j'ajouterai plusieurs sources lumineuses en activant les lumières de 0 à 8 selon mes besoins et adapterai leur couleur ainsi que leurs coordonnées.
Pour jouer sur les matériaux et les modèles de lumières, je jouerai sur la réflexion des matériaux et sur l'aspect ambiant, spéculaire ou diffuse.

Texturer le terrain en utilisant des couleurs.

Pour texturer le terrain à partir de couleurs, je stockerai un tableau de couleurs et un tableau d'indices correspondant chacun à une couleur pour chaque sommet du terrain. Ensuite, au moment de dessiner le terrain, la face correspondrait aux différentes couleurs des sommets convergentes en une couleur commune au centre de la face.
Pour adapter la couleur à l'altitude, chaque sommet aurait ainsi une couleur dépendant directement de sa coordonnée en z.