Ficha Prática n.º 4

Importante:

A ficha deve ser realizada em **grupo (2-3 alunos)**, seguindo as seguintes normas:

- A constituição dos grupos deve ser a mesma da ficha anterior;
- A submissão da ficha prática deve ser através do **Canvas**, em *assignment* próprio, e deverá consistir num ficheiro **ZIP** com uma pasta contendo o **código-fonte**.
- A data-limite para entrega da ficha prática é **21 de novembro de 2021 às 23:55**;
- A apresentação da ficha prática decorrerá nos dias 23 e 25 de novembro, no horário das aulas práticas;
- No dia da apresentação, **TODOS** os elementos do grupo deverão estar presentes. Os elementos ausentes serão classificados com 0 valores;
- A apresentação e discussão poderá ser realizada individualmente.

1. Projeto Cubo

1.1. Crie uma cópia de um projeto das aulas anteriores, onde já tenha o Visual Studio Code configurado corretamente. Deste modo, não é necessário configurar novamente o IDE.

1.2. Abra a pasta do novo projeto no Visual Studio Code e apague todos os ficheiros excepto a pasta *.vscode*

1.3. Faça download do ficheiro "template_ficha04.zip" e coloque o conteúdo na pasta do novo projeto. Deverá ficar somente com um ficheiro *"template_ficha04.c"*.

1.4. Analise o referido programa e observe atentamente o seu funcionamento e as estruturas criadas. Veja particularmente as funções *inicia_modelo*, *init*, *draw, cubo, key*, *mouse*.

1.5. **(2 valores)** Altere a função *cubo()*, que cria um cubo de lado 1 com cores diferentes nas várias faces e centrado na origem. A função existente *cubo()* desenha um polígono de 4 lados. O vector só tem definidos os 4 primeiros vértices.

1.6. **(0.5 valores)** Altere os valores de *modelo.theta[0], modelo.theta[1]* e *modelo.theta[2]* usados na função *draw()* na instrução *glRotatef(...)*.

1.7. **(2 valores)** Insira código na função *timer()* para rodar o cubo em torno dos 3 eixos. Use a função *mouse()* para controlar o eixo a ser rodado (x,y,z) dependendo da tecla do rato premida (use a variável *modelo.eixoRodar*, para guardar o índice para o vector *modelo.theta[]*). Se for premida a tecla correspondente ao eixo que está a rodar, o cubo deverá parar de rodar.

1.8. **(1.5 valores)** Insira código para ligar o buffer de profundidade e não desenhar as faces escondidas do cubo:

- main() – acrescentar GLUT_DEPTH na instrução glutInitDisplayMode
- init() – descomentar a instrução glEnable(GL_DEPTH_TEST);
- draw() – acrescentar GL_DEPTH_BUFFER_BIT na instrução glClear

1.9. **(2 valores)** Inserir as teclas '+' e '-' para aumentar e diminuir o tamanho do cubo usando a instrução *glScalef(...)* na função *draw()*

1.10. **(2 valores)** Use a função *glPolygonMode(...)* para alterar representação da parte de trás dos polígonos para linhas e assim ver se todos as faces estão bem orientadas.

1.11. **(2 valores)** Crie uma função *eixos()* para desenhar a parte positiva dos 3 eixos em 3 cores diferentes centrada na origem.

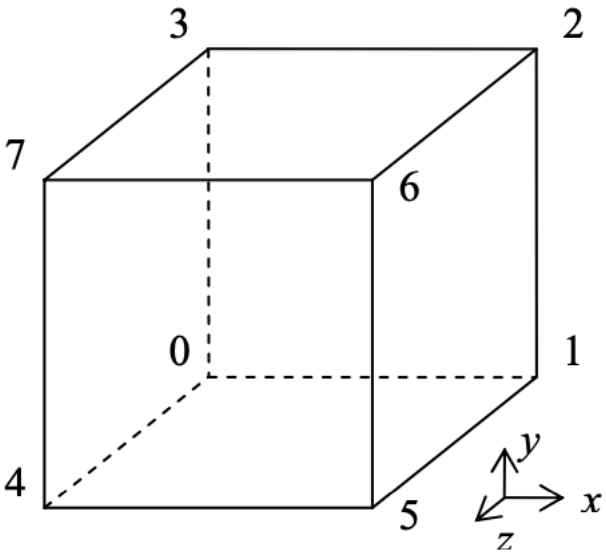
1.12. **(2 valores)** Use a função *eixos()* para desenhar uns eixos pequenos a rodar no canto do ecrã.

1.13. **(2 valores)** Colocar 3 cubos pequenos em cada um dos eixos do grande, use a instrução *glTranslatef(...)*.

1.14. **(2 valores)** Fazer os 3 cubos pequenos aproximarem-se do cubo grande até o tocarem e depois afastarem-se (use a variável *modelo.translacaoCubo* para guardar o afastamento).

1.15. **(2 valores)** Rodar os cubos pequenos em torno do eixo onde estão ao aproximarem-se do grande e evitar que eles rodem quando se afastem (use a variável *modelo.*thetaCubo para guardar a rotação dos cubinhos).

NOTA: Em Mac poderá ser necessário executar o programa no Terminal para visualizar o menu de ajuda.



Funções

glFrontFace(modo)

Instrução para alterar o lado do polígono que é tratado como frente GL_CCW é a opção por omissão.
modo – GL_CCW ou GL_CW

glPolygonMode(face, modo)

Instrução para alterar a representação das faces de um polígono *face* – GL_FRONT, GL_BACK ou GL_FRONT_AND_BACK) *modo* – GL_FILL, GL_LINE ou GL_POINT

glLoadIdentity()

Instrução para carregar a matriz identidade, limpado todas as transformações realizadas.

glPushMatrix()

glPopMatrix()

Instruções para guardar e repor o estado da matriz de transformação usando uma *stack*

glTranslatef(dx,dy,dz)

Instrução para fazer uma translação de *dx* unidades em *x*, *dy* unidades em *y* e *dz* unidades em *z*.

glRotatef(angulo, x, y, z)

Instrução para fazer uma rotação em torno da origem de *angulo* graus em torno do eixo definido pelo vector *x*, *y*, *z*.

glScalef(escala_x, escala_y, escala_y)

Instrução para fazer alterar a escala. Os valores *escala_x*, *escala_y*, *escala_y* são um factor de multiplicação.