

Ficha Prática n.º 3

1. Template para as aulas práticas

- 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_aulas.zip” e coloque o conteúdo na pasta do novo projeto. Deverá ficar somente com um ficheiro “*template_aulas.c*”. Este ficheiro irá servir de template de programas em todos os projectos OpenGL/GLUT que vierem a ser realizados nas aulas práticas.
- 1.4. Analise o referido programa e repare nas semelhanças e diferenças face aos projetos das aulas anteriores. Leia cuidadosamente os comentários.
- 1.5. Execute o programa e observe atentamente o seu funcionamento.

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

2. Projeto Relógio

- 2.1. Crie uma cópia do projeto resultante da questão anterior.
- 2.2. Renomeie o ficheiro “*template_aulas.c*” para “*relogio.c*”
- 2.3. Crie uma função genérica para criar um polígono:

```
void poligono(GLint n, GLfloat x0, GLfloat y0, GLfloat r)
```

que desenhe um polígono regular de n lados, com o centro geométrico no ponto de coordenadas $x0, y0$ e distância aos vértices r .
- 2.4. Use a função da alínea anterior para desenhar um círculo na janela principal da aplicação. Este irá servir de modelo ao mostrador do relógio analógico que se pretende construir.
- 2.5. Crie a função `void mostrador()`

Use as funções do OpenGL para modelar, com base no desenho de pequenos segmentos de recta, as 60 marcas principais do mostrador do relógio, correspondentes aos minutos. Use segmentos um pouco maiores para as marcas correspondentes às horas.

2.6. Crie a função `void ponteiros()`

Use as funções do OpenGL para modelar, com base no desenho de segmentos de recta de diferentes espessuras, os três ponteiros do relógio: segundos, minutos e horas. Use as equações paramétricas da circunferência para localizar um dos vértices dos ponteiros (o outro vértice é sempre coincidente com o centro do mostrador).

2.7. Use a função de temporização do GLUT para construir o mecanismo interno do relógio. A função *timer* só altera as variáveis globais, o *callback* de desenho é que se encarrega de desenhar o relógio segundo os novos valores.

2.8. Adicione duas teclas para permitir aumentar e diminuir, respetivamente, a velocidade do timer (*delay*).

Observações

1. As equações paramétricas da circunferência são as seguintes:

$$\begin{cases} x = r * \cos(t) + x_c \\ y = r * \sin(t) + y_c \end{cases}$$

em que:

(x_c, y_c) são as coordenadas do centro da circunferência;

r é o raio da circunferência;

$0 \leq t < 2\pi$ *radianos*.

2. A correspondência entre diferentes unidades de medida de ângulos é a seguinte:

$$180^\circ = \pi \text{ radianos}$$

3. Ao proceder à animação tenha em atenção dois aspectos importantes:

- a. A marca das 12 horas está localizada na posição do mostrador correspondente a um ângulo de 90° (e não 0°);
- b. Os ponteiros do relógio movem-se em sentido retrógrado (e não em sentido direto).

4. Funções

```
void glLineWidth(GLfloat width);
```

Especifica a grossura das linhas desenhadas.

Esta instrução tem que ser colocada fora do bloco *glBegin/glEnd*.

O valor inicial é 1, sendo afetado pela opção:

glEnable/glDisable(GL_LINE_SMOOTH)

```
void glPointSize(GLfloat size);
```

Especifica a diâmetro dos pontos.

Esta instrução tem que ser colocada fora do bloco *glBegin/glEnd*.

O valor inicial é 1, sendo afetado pela opção:

glEnable/glDisable(GL_POINT_SMOOTH)