

```
/**
 * Arquivo: ponto.h
 * Versão : 1.0
 * Data   : 2024-10-14 16:43
 * -----
 * Esta interface cria dois Tipos de Dados, Ponto2D e Ponto3D, para armazenar
 * coordenadas de pontos no plano e no espaço. Também fornece funções para
 * calcular a distância entre 2 pontos.
 *
 * Prof.: Abrantes Araújo Silva Filho (Computação Raiz)
 *      www.computacaoraiz.com.br
 *      www.youtube.com.br/computacaoraiz
 *      github.com/computacaoraiz
 *      twitter.com/ComputacaoRaiz
 *      www.linkedin.com/company/computacaoraiz
 *      www.abrantes.pro.br
 *      github.com/abrantesasf
 */

/* Inicia boilerplate da interface: */

#ifndef _PONTOS_H
#define _PONTOS_H

/* Includes: */

#include "genlib.h"
#include <math.h>
#include "simpio.h"

/**
 * TIPO DE DADO: Ponto2D
 * -----
 * Este tipo de dado armazena as coordenadas X e Y de um ponto no plano,
 * considerando um sistema de coordenadas cartesiano.
 */

struct st_Ponto2D
{
    double x;    // abscissa
    double y;    // ordenada
};

typedef struct st_Ponto2D Ponto2D;

/**
 * TIPO DE DADO: Ponto3D
 * -----
 * Este tipo de dado armazena as coordenadas X, Y e Z de um ponto no espaço,
 * considerando um sistema de coordenadas cartesiano.
 */

struct st_Ponto3D
{
    double x;    // abscissa
    double y;    // ordenada
    double z;    // profundidade
};

typedef struct st_Ponto3D Ponto3D;
```

```
/**
 * FUNÇÃO: euclidiana_2d
 * Uso: euclidiana_2d(Ponto2D, Ponto2D);
 * -----
 * A função recebe como argumentos dois Ponto2D e retorna a distância
 * Euclidiana entre esses dois pontos, no plano.
 */

double euclidiana_2d (Ponto2D P, Ponto2D Q);

/**
 * FUNÇÃO: euclidiana_2d
 * Uso: euclidiana_2d(Ponto2D, Ponto2D);
 * -----
 * A função recebe como argumentos dois Ponto2D e retorna a distância
 * Euclidiana entre esses dois pontos, no plano.
 */

double euclidiana_3d (Ponto3D P, Ponto3D Q);

/* Finaliza boilerplate: */

#endif
```