

```
/**
 * Arquivo: geometria.c
 * Versão : 1.0
 * Data   : 2024-10-14 17:03
 * -----
 * Este programa implementa um cliente simples para demonstrar o uso de tipos de
 * dados criados pelo usuário, o Ponto2D e o Ponto3D.
 *
 * 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
 */

#include "genlib.h"
#include "ponto.h"
#include "simpio.h"

/* Declarações de subprogramas: */

double meu_euclides (double x1, double x2, double y1, double y2);

/* Função Main: */

int main (void)
{
    Ponto2D A;
    A.x = 0.0, A.y = 0.0;

    Ponto2D B;
    B.x = 3.0, B.y = 4.0;

    Ponto3D X;
    X.x = 0.0, X.y = 0.0, X.z = 0.0;

    Ponto3D Y;
    Y.x = 3.0, Y.y = 3.0, Y.z = 3.0;

    printf("A distância entre (%.1f, %.1f) e (%.1f, %.1f) é de %.5f.\n",
           A.x, A.y, B.x, B.y, euclidiana_2d(A, B));

    printf("A distância entre (%.1f, %.1f) e (%.1f, %.1f) é de %.5f.\n",
           A.x, A.y, B.x, B.y, meu_euclides(A.x, B.x, A.y, Y.y));

    printf("A distância entre (%.1f, %.1f, %.1f) e (%.1f, %.1f, %.1f) é de %.5f.\n",
           X.x, X.y, X.z, Y.x, Y.y, Y.z, euclidiana_3d(X, Y));
}

/**
 * FUNÇÃO: meu_euclides
 * Uso: meu_euclides(x1, x2, y1, y2);
 * -----
 * Esta função recebe 4 números double sem qualquer relação explícita entre si,
 * x1, x2, y1 e y2, e interpreta esses números como se fossem as coordenadas de
 * dois pontos no plano cartesiano:
 */
```

```
*
*   Ponto 1: (x1, y1)
*   Ponto 2: (x2, y2)
*
* Partindo dessa interpretação a função retorna a distância Euclidiana entre
* esses dois "pontos". Note que nada na função remete à interpretação dos
* valores como pontos.
*/

double meu_euclides (double x1, double x2, double y1, double y2)
{
    return sqrt(pow((x1 - x2), 2.0) + pow((y1 - y2), 2.0));
}
```