

Problemas de geometria

Implemente as funções declaradas no arquivo `geometria.h` (veja a listagem do código desse arquivo abaixo).

```
#ifndef GEOMETRIA
#define GEOMETRIA

struct s_ponto {
    double x, y;
};

typedef struct s_ponto ponto;
typedef struct s_ponto vetor;

struct s_segmento {
    ponto p, q;
};

typedef struct s_segmento segmento;

struct s_triangulo {
    ponto p, q, r;
};

typedef struct s_triangulo triangulo;

// Calcula o produto interno <u,v>
double produto_interno(vetor u, vetor v);

// Calcula o vetor u - v
vetor subtrai(vetor u, vetor v);

/* Calcula o vetor resultante da rotação do vetor v
   de um ângulo de 90 graus (no sentido anti-horário). */
vetor roda90(vetor v);

// Calcula distância
double distancia(ponto p, ponto q);
```

```

/* Retorna 1 se o coseno do ângulo entre os vetores u e v é
   positivo e retorna -1 se for negativo e 0 se for nulo. */
int sinal_do_coseno(vetor u, vetor v);

/* Retorna 1 se p, q e r estão em sentido horário e -1 se for
   anti-horário. Se os pontos forem colineares devolva 0. */
int sentido(ponto p, ponto q, ponto r);

/* Retorna 1 se os interiores dos segmentos se cruzam
   e retorna 0 caso contrário. */
int cruza(segmento s, segmento t);

/* Retorna 1 se o ponto p está no interior do triângulo t
   e retorna 0 caso contrário. */
int dentro(ponto p, triangulo t);

/* Devolve a cordenada do ponto em que s e t se intersecta
   caso eles se intersectem ou qualquer ponto caso contrário. */
ponto cruzamento(segmento s, segmento t);

// Calcula o ponto que é a projeção de p no segmento s.
ponto projeta(ponto p, segmento s);

/* Devolve 1 se o interior dos triângulos a e b se
   intersectam e devolve 0 caso contrário. */
int intersecta(triangulo a, triangulo b);

#endif

```

Observações

Siga as instruções no arquivo README para compilação. Seu código deverá incluir `geometria.h` via

```
#include "geometria.h"
```

no início do programa. É proibido modificar o conteúdo desse arquivo.

Se você já havia feito o código, por favor adapte-o para que ele seja compatível com `geometria.h` a fim de que possa ser corrigido automaticamente pelo script de correção automática disponível no arquivo EXE03.zip.

As funções foram declaradas numa certa ordem. A ordem foi pensada de modo que, na hora de implementar uma função é bem provável que você precise chamar alguma função que foi declarada antes dela (exceto pelas primeiras que são muito simples).

Geometria analítica

Se você não se lembra de geometria analítica, aqui vão algumas dicas.

- As definições que você esqueceu você pode consultar no Google.
- O cosseno do ângulo Θ entre dois vetores pode ser calculado através da lei dos cossenos

$$c^2 = a^2 + b^2 - 2ab \cos(\Theta)$$

onde c é o comprimento lado oposto ao ângulo Θ no triângulo formado pelos dois vetores, e a e b são os comprimentos dos vetores. Mas primeiro expanda e depois simplifique a expressão que você encontrar para o cose-no, pois o que queremos saber do cose-no é só o sinal! A expressão final depois de simplificada fica *bem* simples!

- Nas funções **cruza** e **dentro**, você deve usar apenas algumas chamadas à função **sentido** e uns poucos operadores lógicos.
- Para a função **projeta**, lembre-se de que se se projetamos um vetor u sobre um vetor v , o vetor resultante é um vetor múltiplo de v , digamos αv . Portanto, a dica é: você deve encontrar α que minimiza

$$f(\alpha) = \|u - \alpha v\|.$$

Outra dica é que, para minimizar a raiz quadrada de algo que é sempre não negativo, basta minimizar esse algo!