

# CI1338 - Geometria Computacional

## INFO7061 - Tópicos em Geometria Computacional

Primeiro Trabalho

22 de março de 2025

### 1 Introdução

O trabalho consiste em fazer uma implementação do seguinte problema: *classificação de polígonos e pontos interiores*.

O problema é definido da seguinte forma:

**Dados:** um conjunto  $\mathcal{S}$  de linhas poligonais fechadas no plano e um conjunto  $\mathcal{P}$  de pontos no plano

**Encontrar:** classificação de cada uma das linhas poligonais fechadas como polígonos “não simples”, “simples e não convexos” ou “simples e convexos”; e uma lista de polígonos simples (convexos ou não) que contêm o ponto  $P$ , para cada  $P \in \mathcal{P}$ .

### 2 Resolução do problema

A resolução do problema, ou seja, a descrição do problema, dos algoritmos e suas corretudes, deve estar em um texto claro em formato de um artigo e em pdf. Deve conter os nomes dos autores (alunos), uma introdução com o problema, os algoritmos e suas explicações. Todas as referências que forem usadas devem estar citadas corretamente no texto.

### 3 Especificação da implementação

A implementação pode ser feita em qualquer linguagem, contanto que seja possível rodar no ambiente computacional do DINF.

A entrada de dados deve ser feita pela entrada padrão (stdin) e a saída de dados pela saída padrão (stdout), ou seja, o seu programa lê do teclado e escreve na tela. O objetivo é que seja executado com redirecionamento de arquivos, como o comando abaixo:

```
$ poligonos < entrada.txt > saida.txt
```

O trabalho deve ser entregue com um **makefile** de forma que ao digitar o comando **make** o executável **poligonos** seja construído no diretório corrente.

Você deve entregar um arquivo compactado (no formato **tar.gz**) com seu nome (ou login) com os seguintes arquivos no diretório raiz:

- texto (em pdf);
- os fontes (podem estar em subdiretórios);
- makefile;
- exemplos usados no texto (podem estar em subdiretórios).

A entrega deve ser feita por e-mail para **andre@inf.ufpr.br**, em um arquivo compactado com todos os arquivos do trabalho, com assunto “geocomp-trabalho 1” (exatamente).

## 4 Classificação e Intersecção de Polígonos

O objetivo é classificar cada polígono dado em “não simples”, “simples e não convexo” ou “simples e convexo”. Além disso, encontrar, para cada ponto  $P \in \mathcal{P}$  todos os polígonos simples (convexos ou não) que contêm  $P$ .

O nome do programa executável deve ser **poligonos**.

A entrada de dados será um texto com a descrição dos polígonos ( $\mathcal{S}$ ) e dos pontos ( $\mathcal{P}$ ). A primeira linha tem dois números inteiros,  $m$  e  $n$ , que são a quantidade de polígonos e de pontos, respectivamente. As linhas seguintes são as descrições dos  $m$  polígonos seguidas das descrições dos  $n$  pontos. A descrição de cada polígono,  $P_i$ , inicia com uma linha para o número inteiro  $n_i$  de vértices do polígono. As  $n_i$  linhas seguintes tem as coordenadas inteiras  $X$  e  $Y$  de cada um dos vértices, separadas por um espaço em branco. As últimas  $n$  linhas tem as coordenadas inteiras  $X$  e  $Y$  de cada um dos pontos de  $\mathcal{P}$ .

Exemplo de entrada:

```
3 3
4
1 4
15 4
15 20
1 20
3
8 12
25 8
25 18
5
2 6
2 1
15 2
20 1
20 6
12 12
25 2
5 5
```

Considere que os polígonos são numerados de 1 a  $m$  (do primeiro ao último) e os pontos numerados de 1 a  $n$ .

Os vértices de cada polígono são dados em ordem anti-horária.

A saída de dados deve ser um texto com a classificação de cada polígono e a lista polígonos que contêm cada ponto.

Para a primeira parte, a saída deve ter  $m$  linhas, cada uma com um número  $i$  (de 1 a  $m$ ) seguido de um espaço e uma das expressões *nao simples*, *simples e nao convexo* ou *simples e convexo* (cada palavra separada das outras por apenas um espaço e sem acentos).

Para a segunda parte, a saída deve ter  $n$  linhas, cada uma com um número  $i$  (de 1 a  $n$ ) seguido de “:”, e a sequência de índices dos polígonos que o contêm (em ordem crescente) separados por espaço (sem espaço no fim). Caso um ponto não esteja contido em nenhum polígono, a sequência deve estar vazia.

Exemplo de saída:

```
1 simples e convexo
2 simples e convexo
3 simples e nao convexo
1:1 2
2:
3:1 3
```