

✓ Geometria Computacional?

A geometria computacional é a área que estuda como representar, analisar e manipular formas geométricas usando algoritmos. No GIS, isso inclui:

1. Representar **pontos, linhas e áreas**;
2. Medir **distâncias, comprimentos e áreas**;
3. Criar **buffers**;
4. Verificar intersecções;
5. Realizar operações espaciais.

O Shapely é um módulo Python **muito importante** para isso.

```
!pip install shapely
```

```
Requirement already satisfied: shapely in /usr/local/lib/python3.12/dist-packages (2.1.2)
Requirement already satisfied: numpy>=1.21 in /usr/local/lib/python3.12/dist-packages (from shapely) (2.0.2)
```

```
import pandas as pd
```

```
from shapely.geometry import Point, LineString, Polygon
```

✓ POINT (Ponto)

Representa uma coordenada no espaço.

```
ponto = Point(32.58, -25.96) # lon, lat
```

```
type(ponto)
```

```
shapely.geometry.point.Point
```

✓ LINESTRING (Linha)

Representa uma linha conectando vários pontos.

```
linha = LineString([
    (32.58, -25.96),
    (32.60, -25.97),
    (30.62, 29.98),
    (33.43, 23.45)
])
```

```
linha
```



```
linha.length
```

```
0.044721359549994844
```

✓ POLYGON (Polígono)

Representa áreas fechadas, como bairros, parcelas, distritos.

```
poligono = Polygon([
    (32.58, -25.96),
    (32.60, -25.96),
    (32.60, -25.98),
    (32.58, -25.98)
])
```

```
poligono
```



```
poligono.area
```

```
0.000400000000000054
```

```
poligono.bounds
```

```
(32.58, -25.98, 32.6, -25.96)
```

```
poligono.centroid
```



✓ Medidas Geométricas no Shapely

O **Shapely** permite calcular várias propriedades geométricas essenciais no GIS.

Estas medidas são calculadas em unidades do sistema de coordenadas da geometria (no caso de lat/lon, são graus, não metros).

✓ Área de um Polígono

```
# importacao da biblioteca Shapely para poligono
from shapely.geometry import Polygon
```

```
pol = Polygon([
    (0,0),
    (4,0),
    (4,3),
    (0,3)
])
```

```
12.0
```

```
pol.area
```

```
12.0
```

✓ Comprimento de uma Linha (LineString)

```
linha = LineString([(0,0), (3,4)])
```

```
linha.length
```

```
5.0
```

▼ Centroides

```
pol.centroid
```



```
pol.centroid.x
```

```
2.0
```

```
pol.centroid.y
```

```
1.5
```

▼ Operações Espaciais

O Shapely oferece várias operações essenciais para análises espaciais.

Buffer (Zona ao redor de uma geometria)

Muito usado para análises de proximidade.

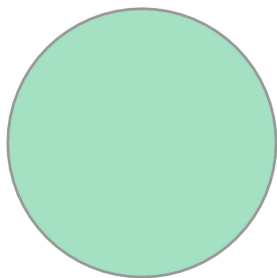
```
# cria o ponto  
p = Point(0,0)
```

```
buffer1 = p.buffer(10)  
buffer2 = p.buffer(100)
```

```
buffer1
```



```
buffer2
```



```
buffer1.area
```

```
313.6548490545941
```

```
buffer2.area
```

```
31365.484905459387
```

```
!pip install matplotlib
```

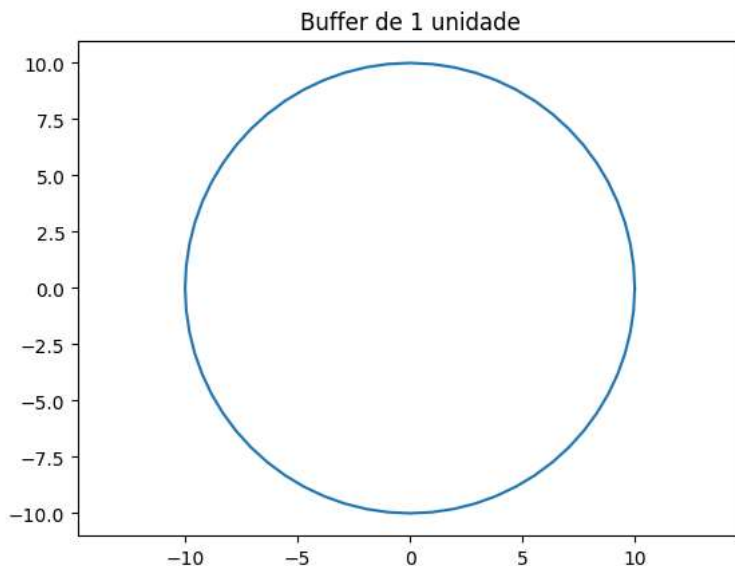
```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4.9)
Requirement already satisfied: numpy>=1.23 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (25.0)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.2.5)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17)
```

```
import matplotlib.pyplot as plt
```

```
buffer1.exterior.xy
```

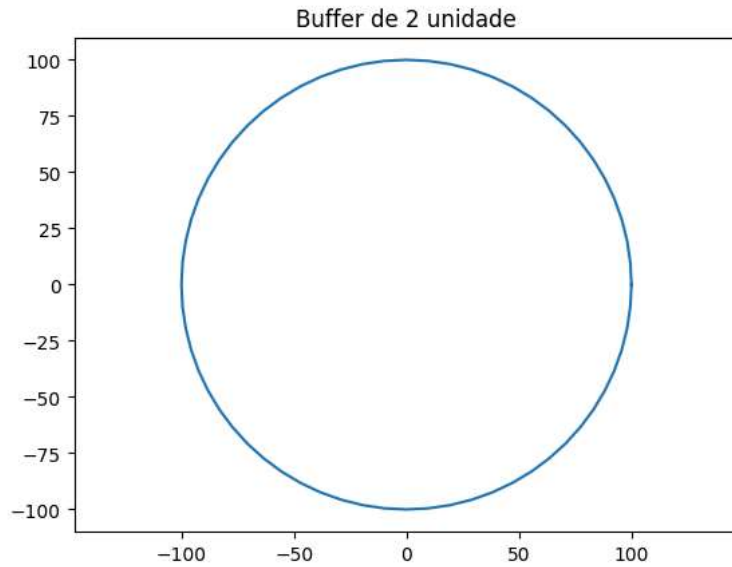
```
(array('d', [10.0, 9.95184726672197, 9.807852804032304, 9.569403357322088, 9.238795325112868, 8.819212643483551,
8.314696123025453, 7.73010453362737, 7.0710678118654755, 6.343932841636455, 5.555702330196023, 4.713967368259978,
3.8268343236508984, 2.902846772544623, 1.9509032201612833, 0.9801714032956077, 0.0, -0.9801714032956065, -1.950903220161282,
-2.9028467725446214, -3.826834323650897, -4.713967368259977, -5.55570233019602, -6.3439328416364535, -7.071067811865475,
-7.73010453362737, -8.314696123025453, -8.81921264348355, -9.238795325112868, -9.569403357322088, -9.807852804032304,
-9.951847266721968, -10.0, -9.95184726672197, -9.807852804032304, -9.56940335732209, -9.238795325112868, -8.819212643483551,
-8.314696123025454, -7.730104533627371, -7.071067811865477, -6.343932841636459, -5.555702330196022, -4.713967368259978,
-3.8268343236509033, -2.9028467725446245, -1.9509032201612866, -0.9801714032956045, 0.0, 0.9801714032956009, 1.950903220161283,
2.9028467725446205, 3.8268343236509, 4.713967368259976, 5.555702330196018, 6.343932841636456, 7.071067811865474,
7.730104533627367, 8.314696123025453, 8.819212643483548, 9.238795325112864, 9.569403357322088, 9.807852804032303,
9.95184726672197, 10.0]),
array('d', [0.0, -0.980171403295606, -1.9509032201612824, -2.902846772544623, -3.826834323650898, -4.7139673682599765,
-5.555702330196022, -6.343932841636455, -7.071067811865475, -7.73010453362737, -8.314696123025453, -8.81921264348355,
-9.238795325112868, -9.56940335732209, -9.807852804032304, -9.951847266721968, -10.0, -9.95184726672197, -9.807852804032304,
-9.56940335732209, -9.238795325112868, -8.819212643483551, -8.314696123025454, -7.730104533627371, -7.0710678118654755,
-6.343932841636455, -5.555702330196022, -4.713967368259978, -3.826834323650899, -2.902846772544624, -1.9509032201612861,
-0.9801714032956083, 0.0, 0.9801714032956059, 1.9509032201612837, 2.902846772544621, 3.8268343236508966, 4.7139673682599765,
5.55570233019602, 6.343932841636453, 7.071067811865475, 7.730104533627367, 8.314696123025453, 8.81921264348355,
9.238795325112864, 9.569403357322088, 9.807852804032303, 9.95184726672197, 10.0, 9.95184726672197, 9.807852804032304,
9.56940335732209, 9.238795325112866, 8.819212643483551, 8.314696123025454, 7.730104533627369, 7.071067811865477,
6.343932841636459, 5.555702330196022, 4.713967368259979, 3.826834323650904, 2.902846772544625, 1.9509032201612873,
0.980171403295605, 0.0]))
```

```
x, y = buffer1.exterior.xy
plt.plot(x, y)
plt.title("Buffer de 1 unidade")
plt.axis("equal")
plt.show()
```



```
x, y = buffer2.exterior.xy
plt.plot(x, y)
```

```
plt.title("Buffer de 2 unidade")
plt.axis("equal")
plt.show()
```



Intersection (Interseção) **bold text** Área em comum entre duas geometrias.

```
p1 = Point(0,0).buffer(2)
p2 = Point(1,0).buffer(2)
```

p1



p2



```
inter = p1.intersection(p2)
```

inter

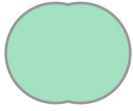


Union (União)

Combina duas geometrias em uma só.

```
u = p1.union(p2)
```

u



```
dif = p1.difference(p2)
```

```
dif
```



```
p2.difference(p1)
```



Union (União)

Combina duas geometrias em uma só.

Fica como exercicio saber fazer a uniao de dois pontos

TPC – 3: Construção de Geometrias e Análises Espaciais

1. Obtenção de Coordenadas em Outros SIGs

Utilize ferramentas externas como QGIS, Google Earth, Google Maps ou similares para obter coordenadas reais da sua zona de interesse.

1.1 Criação de Pontos

Registe 10 (ou mais) coordenadas, preferencialmente da sua própria área geográfica (bairro, localidade, distrito).

Guarde-as em formato tabular (CSV, Excel)

1.2 Criação de Linhas

Com base nos pontos criados em 1.1, construa 4 linhas, representando ruas ou caminhos adjacentes.

As linhas devem ser diferenciadas por **cores** distintas para facilitar a visualização.

1.3 Criação de Polígonos

Construa dois polígonos (A e B), dividindo os 10 pontos em duas áreas distintas.

Calcular para cada polígono:

- Área
- Perímetro
- Centróide

2. Cobertura de Antenas de Telecomunicações

Selecione qualquer zona à sua escolha e:

2.1 Criação dos Pontos das Antenas

- Defina 3 ou 4 pontos que representem antenas de telefonia.

2.2 Cálculo da Área de Abrangência

- Crie um buffer de 5 km ao redor de cada antena para representar a área de cobertura.

2.3 Interseção das Coberturas

- Repita o processo para um buffer de 7 km.

Verifique se existe interseção entre as áreas de abrangência das antenas quando o raio é de 7 km

DICAS

1. Utilize ferramentas SIG para garantir precisão das coordenadas Ao recolher os pontos no **QGIS, Google Earth ou Google Maps**, certifique-se de copiar as coordenadas no formato latitude/longitude (WGS84). Isso evita distorções futuras e garante compatibilidade com qualquer biblioteca Python ou software GIS.
2. Agrupe os pontos de forma lógica para facilitar a criação dos polígonos
3. Para as linhas, escolha ruas que realmente cruzam ou conectam os seus pontos
4. Para as antenas de telefonia, distribua-as estrategicamente Posicione as antenas de forma que não fiquem todas muito próximas nem demasiado afastadas. Assim, a análise da área de abrangência (buffers) fica mais interessante e as intersecções entre áreas fazem mais sentido na prática.
5. Avalie cuidadosamente o raio dos buffers (5 km e 7 km) Ao comparar as áreas de influência, reflita sobre o impacto do aumento do raio: áreas maiores tendem a sobrepor-se mais, enquanto áreas menores criam zonas isoladas. Observe como o nível de intersecção muda à medida que aumenta o alcance das antenas.