Taller 4-Computación y estadística.

Karen Alexandra Rojas Rincon

Asignación Convertir en coordenadas polares los datos de de y dl

```
import pandas as pd
import numpy as np


def prog(r, n, a1):
  an = a1 + r*(n-1)
  seq = np.arange(start=a1, stop=an, step=r)
  return seq

prog(r=7, n=20, a1=15)
```

```
     array([ 15,  22,  29,  36,  43,  50,  57,  64,  71,  78,  85,  92,  99,
            106, 113, 120, 127, 134, 141])
```

```
np.random.seed(218)

df1 = pd.DataFrame({
    'de': np.sort(np.random.normal(loc = 4, scale = 1, size=96)),
    'dl': np.sort(np.random.normal(loc=4.5, scale=1.2, size=96)),
    'ddd': np.repeat(prog(r=7, n=25, a1=15), 4)
})

df1['localidad'] = np.repeat(['l1','l2']*24, 2)
df1.head()
```

Convirtiendo coordenadas polares de los datos de y dL

```
import numpy as np
```

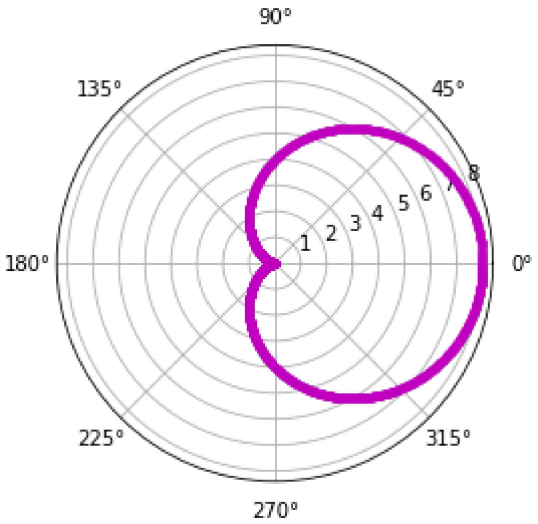
```
def cart2pol(x, y):
    Distancia_origen = np.sqrt(x**2 + y**2)
    Angulo = np.degrees(np.arctan2(y, x))
    return(Distancia_origen, Angulo)

coordpde = cart2pol(df1['ddd'], df1['de'])
#print(coordpde)
df3 = pd.DataFrame({
    'Distancia_origen':coordpde[0],
    'Angulo': coordpde[1],
    })
print(df3)

coordpdl = cart2pol(df1['ddd'], df1['dl'])
#print(coordpdl)
df4 = pd.DataFrame({
    'Distancia_origen':coordpdl[0],
    'Angulo':coordpdl[1],
    })
print(df4)


coordpdle = cart2pol(df1['dl'], df1['de'])
#print(coordpde)
df5 = pd.DataFrame({
    'Distancia_origen':coordpdle[0],
    'Angulo': coordpdle[1],
    })
print(df3)
```

```
    Distancia_origen    Angulo
0          15.033022  3.798342
1          15.116159  7.107577
2          15.125229  7.378006
3          15.171621  8.626162
4          22.130679  6.229563
..               ...       ...
91        169.092860  1.898929
92        176.099226  1.923501
93        176.099370  1.924891
94        176.100721  1.937928
95        176.101439  1.944821

[96 rows x 2 columns]
    Distancia_origen    Angulo
0          15.126545  7.416394
1          15.138604  7.759165
2          15.194439  9.175963
3          15.205817  9.437661
4          22.145045  6.561277
..               ...       ...
91        169.131869  2.262688
```

```
92        176.127238  2.178003
93        176.142714  2.306577
94        176.149097  2.357558
95        176.180378  2.592913

[96 rows x 2 columns]
    Distancia_origen    Angulo
0          15.033022  3.798342
1          15.116159  7.107577
2          15.125229  7.378006
3          15.171621  8.626162
4          22.130679  6.229563
..               ...       ...
91        169.092860  1.898929
92        176.099226  1.923501
93        176.099370  1.924891
94        176.100721  1.937928
95        176.101439  1.944821

[96 rows x 2 columns]
```

Gráfico de cardioide

```
import numpy as np
import matplotlib.pyplot as plt
import math


plt.axes(projection = 'polar')

a=4

rads = np.arange(2,(6 * np.pi), 0.01)

for rad in rads:
    r = a + (a*np.cos(rad))
    plt.polar(rad,r,'m.')

plt.show()
```



✓ 3 s completado a las 22:26