Matplotlib:

Pyplot

Instalación

Desde la terminal corremos

```
mambda install matplotlib
```

Una vez lo ejecutamos procedemos a importarlo:

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

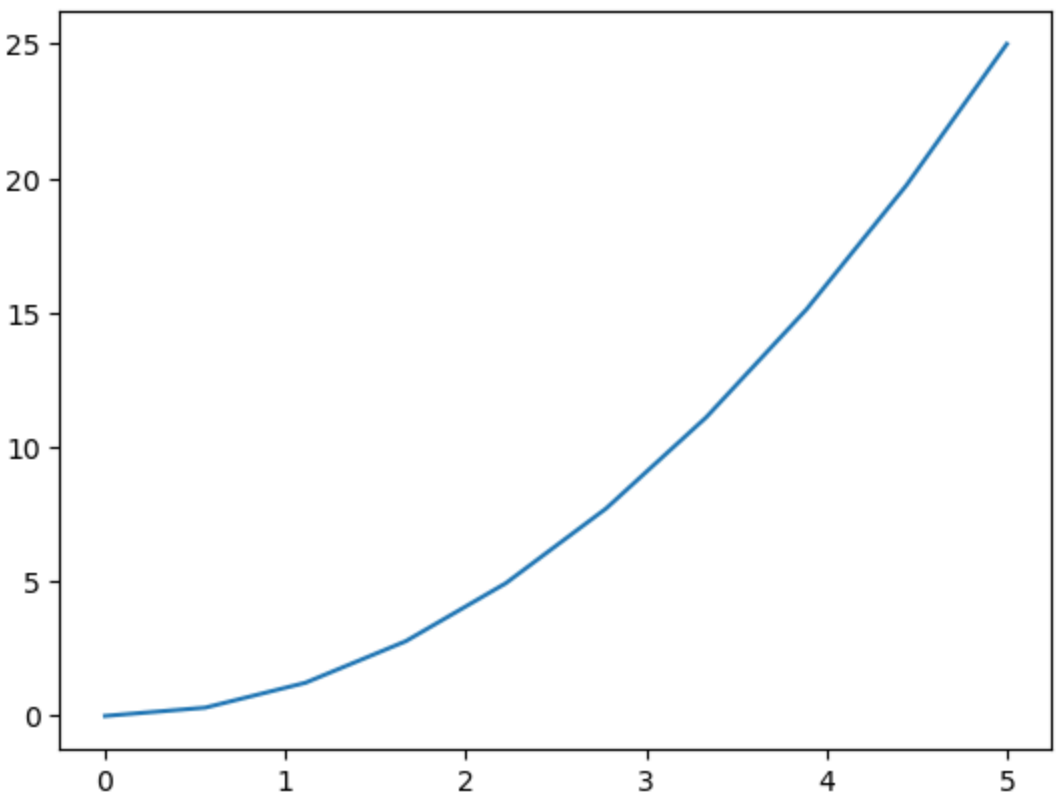
In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
```

Una vez se importa, procedemos a trabajar con las variables.

In [ ]:
```
#Creando una variable
x = np.linspace(0,5,10)
y = x**2

#Visualización de variables
print(f'X = {x}\nTipo = {type(x)}\n\nY = x^2 = {y}\nTipo = {type(y)}\n')
```

```
X = [0.         0.55555556 1.11111111 1.66666667 2.22222222 2.77777778
 3.33333333 3.88888889 4.44444444 5.        ]
Tipo = <class 'numpy.ndarray'>

Y = x^2 = [ 0.          0.30864198  1.2345679   2.77777778  4.9382716   7.71604938
 11.11111111 15.12345679 19.75308642 25.        ]
Tipo = <class 'numpy.ndarray'>
```

In [ ]:
```
#Trabajando con Pyplot

#Generando gráfica
plt.plot(x,y)
#Mostrando gráfica
plt.show()
```



Cambiando los colores de la gráfica

Podemos cambiar los colores de la gráfica mediante el comando:

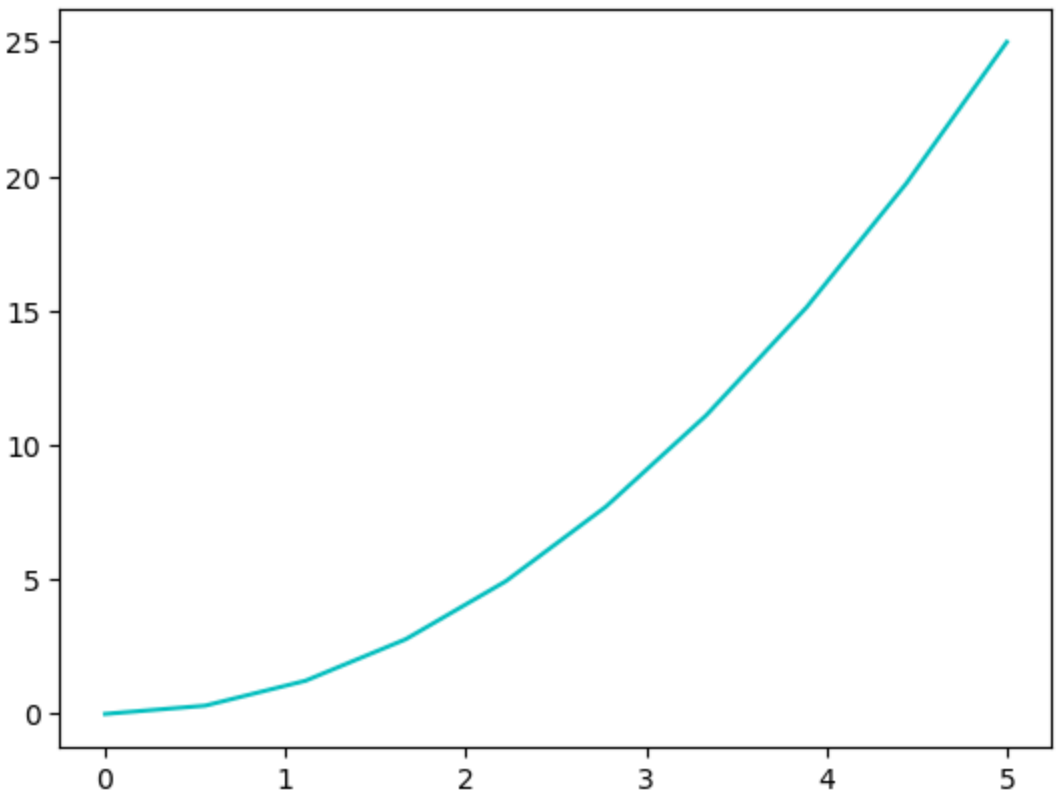
```
plt.plot(x,y,'character')
#Ejemplo
plt.plot(x,y,'r')   #Nos muestra en color rojo
```

character	color
‘b’	blue
‘g’	green
‘m’	magenta

character	color
'r'	red
'c'	cyan
'y'	yellow
'k'	black
'w'	white

In [ ]:
```
plt.plot(x,y,'c')
plt.show
```

Out[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Cambiar más parámetros

Format Strings

character	description
‘.’	point marker
‘,’	pixel marker
'x'	x marker
‘o’	circle marker
‘v’	triangle_down marker
‘^’	triangle_up marker
‘<’	triangle_left marker
‘>’	triangle_right marker

character	description
'1'	tri_down marker
'2'	tri_up marker
'3'	tri_left marker
'4'	tri_right marker
'8'	octagon marker
's'	square marker
'p'	pentagon marker
'P'	plus (filled) marker
'*'	star marker

character	description
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'X'	x (filled) marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Line Styles

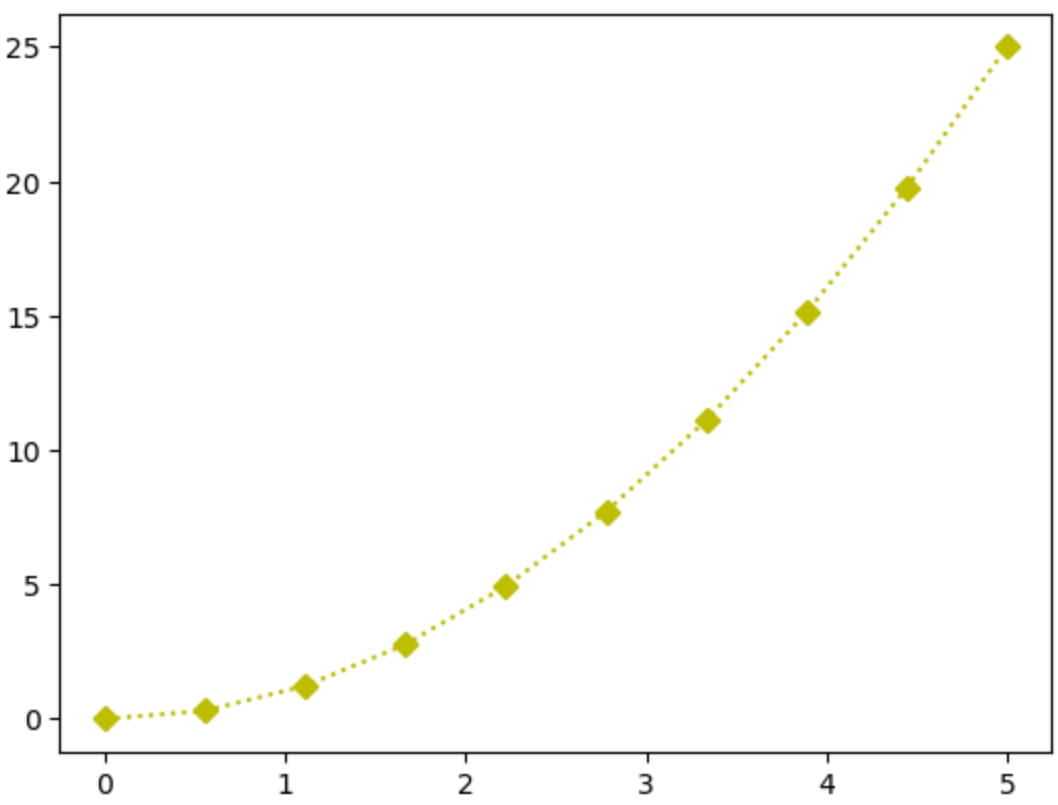
character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style

Aplicando en código

```
plt.plot(x,y, 'yD:') #grafica de color amarillo, con diamantes y puntos consecutivos
plt.show()
```

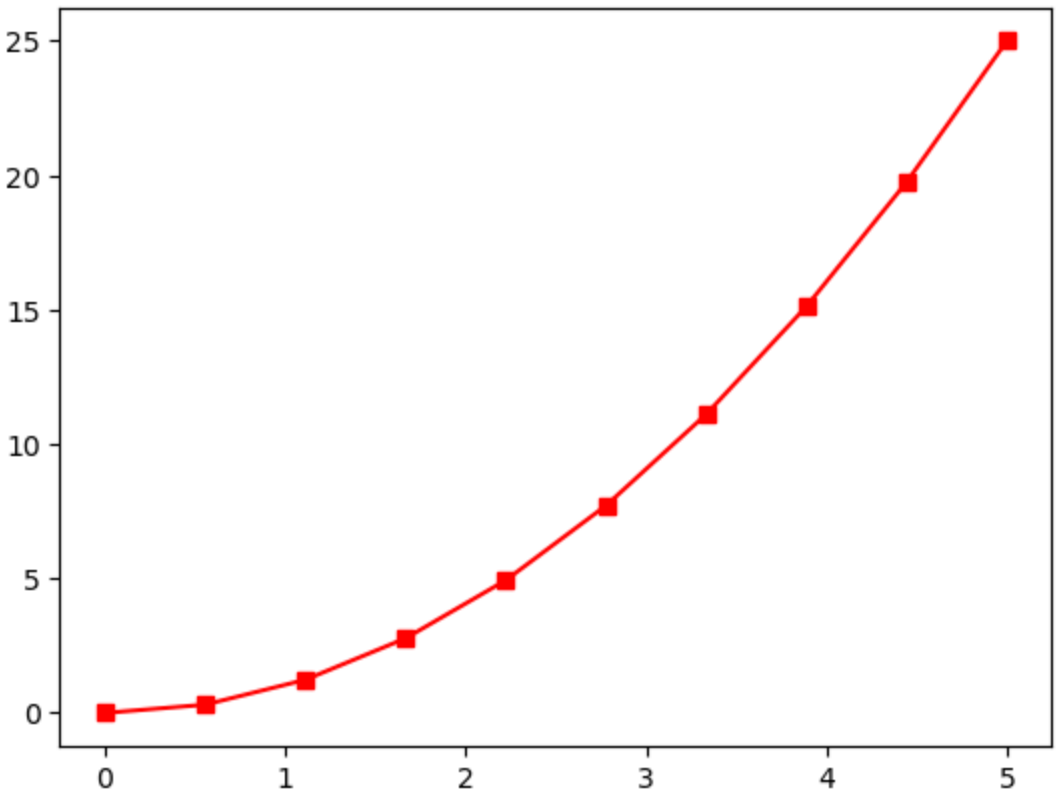
In [ ]:
```
plt.plot(x,y,'yD:')
plt.show
```

Out[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [ ]:
```
#Rojo-Squares-Continuidad
plt.plot(x,y,'rs-')
#Graficando
plt.show
```

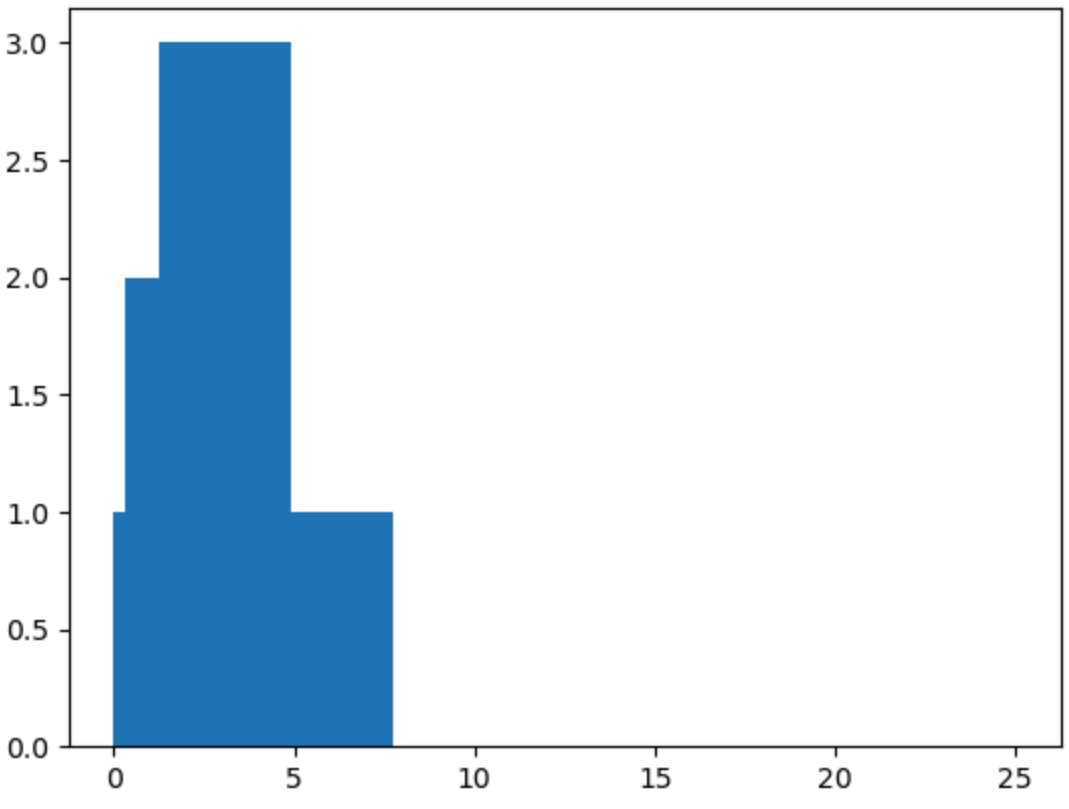
Out[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



NOTA:

Esta sintaxis es muy similar a la usada en Matlab

In [ ]:
```
#Graficando histograma con 2 variables
plt.hist(x,y)
plt.show()
```

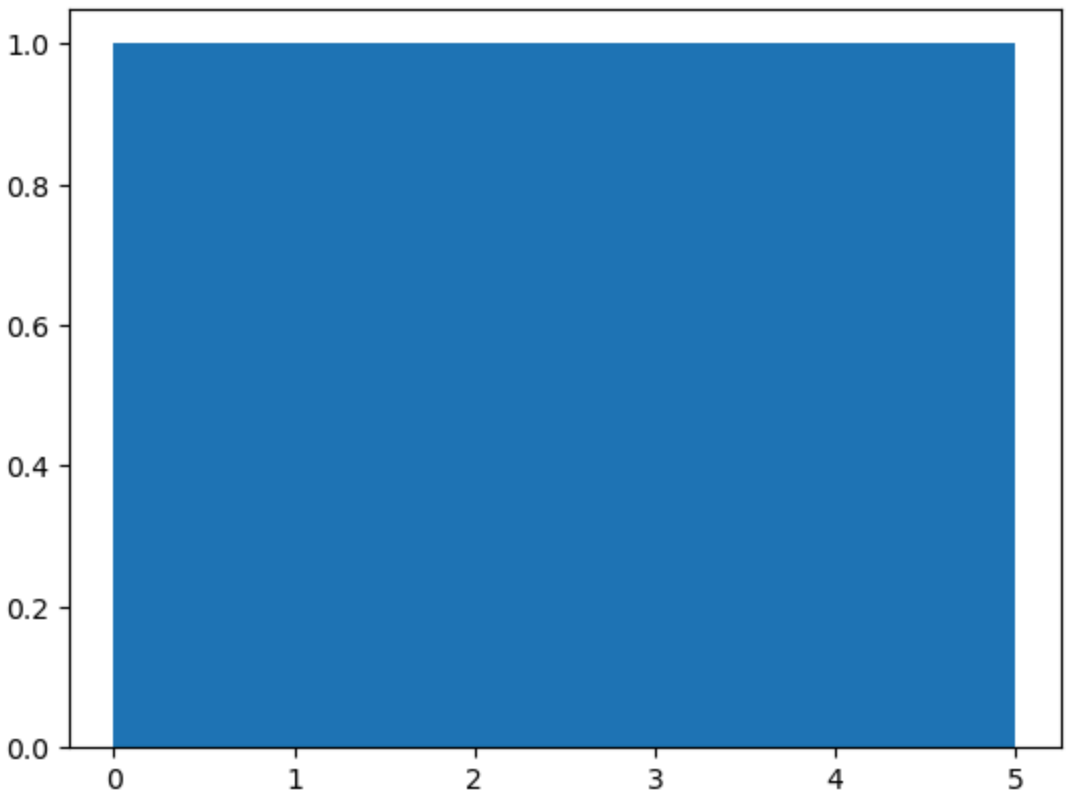


Nota:

Una buena práctica es que cuando estés haciendo histogramas, siempre explora multiples anchos de bins

1. bins = 1
2. bins = 3
3. bins = 5
4. bins = 15

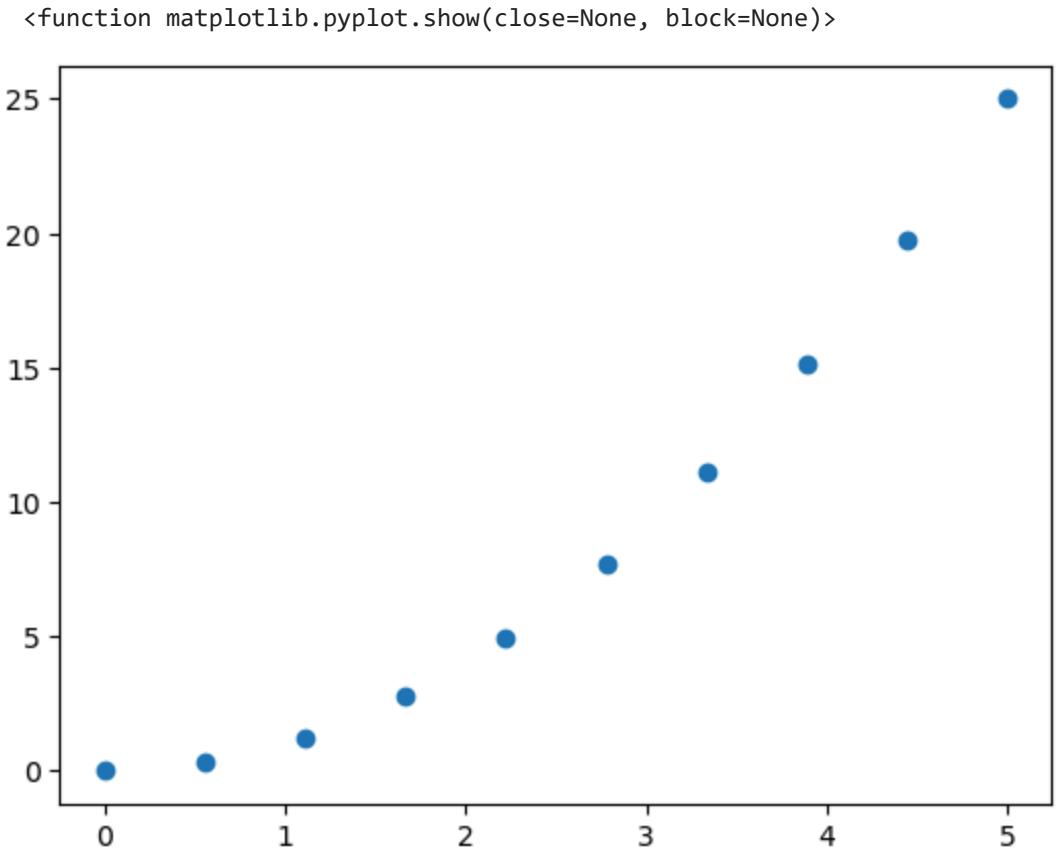
In [ ]:
```
#Graficando histograma con 1 variable
#Con los 10 datos de X
plt.hist(x)
plt.show()
```



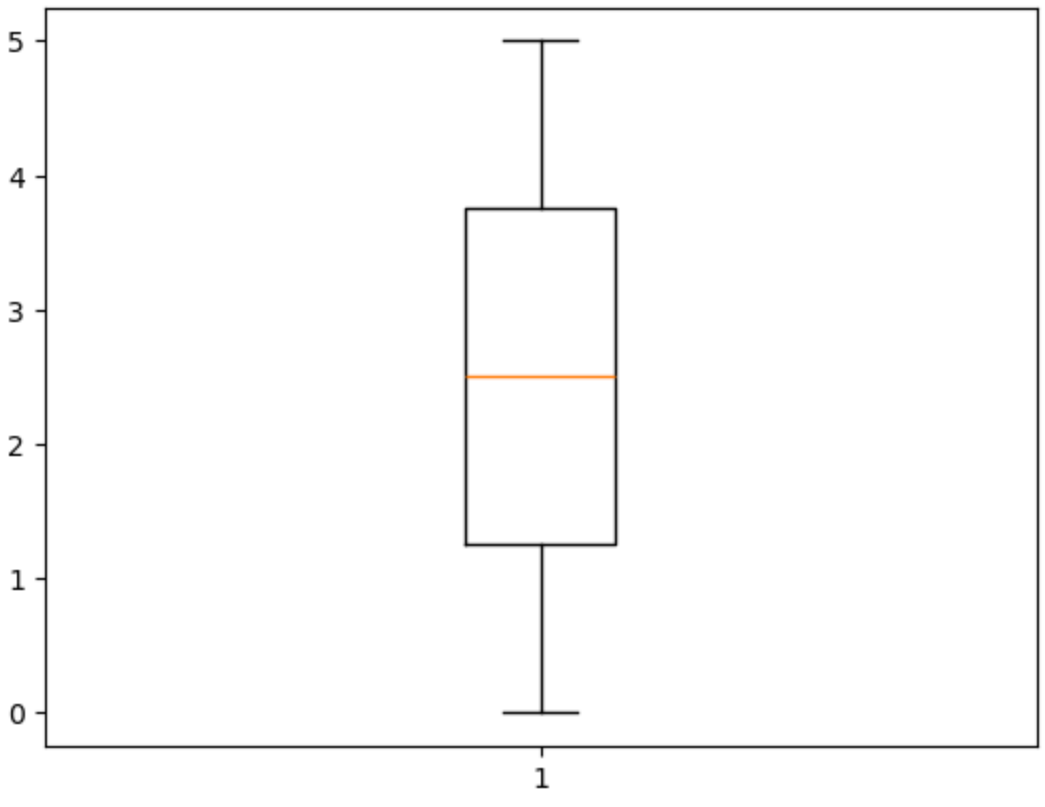
In [ ]:
```
#Graficando gráfica tipo pastel
#Con los 10 datos de X
plt.pie(x)
plt.show()
```

In [ ]:
```
#Grafica de relacion (Scatter)
plt.scatter(x,y)
plt.show
```

Out[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [ ]:
```
#Ver la distribución de los datos
plt.boxplot(x)
plt.show()
```



Referencias:

- Plot types
- Cheat sheet 1
- Cheat sheet 2
- Cheat sheet 3
- Cheat sheet 4
- Cheat sheet 5