Nombre de la práctica	MATPLOTLIB			No.	3
Asignatura:	SIMULACION	Carrera:	INGENIERÍA EN SISTEMAS COMPUTACIONALES	Duración de la práctica (Hrs)	2 horas

NOMBRE DEL ALUMNO: Francisco David Colin Lira
GRUPO: 3502

I. Competencia(s) específica(s):

Desarrolla programas para generar números pseudoaleatorios utilizando diferentes métodos y aplica pruebas estadísticas para garantizar que sean uniformemente distribuidos e independientes con el fin de utilizarlos en la solución de problemas.

Encuadre de CACEI:

Encuadre con CACEI: Registra el (los) atributo(s) de egreso y los criterios de desempeño que se evaluarán en esta práctica.

Encuadre con CACEI

No. atributo	Atributos de egreso del PE que impactan en la asignatura	No.Criterio	Criterios de desempeño	No. Indicador	Indicadores
1	El estudiante identificará los principios de las ciencias básicas para la resolución de problemas prácticos de ingeniería	CD1	Propone alternativas de solucion	I1	diseño algoritmico
				I2	empleo de formulas y funciones
2	el estudiante diseñará esquemas de trabajo y procesos, usando metodologías congruentes en la resolución de problemas de ingeniería en sistemas computacionales	CD1	identifica metodologías y procesos empleados en la resolución de problemas	I1	identificacion y reconocimiento de distintas metodologias para la resolucion de problemas.
				I2	Manejo de procesos especificos en la solucion de problemas y/o deteccion de necesidades.
		CD2	diseña soluciones a problemas, empleando metodologías apropiadas al area	I1	uso de metodologías para el modelado de la solucion de sistemas y aplicaciones
				I2	diseño algoritmico.

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

Aula y equipo de cómputo personal.

III. Material empleado:

- Equipo de cómputo
- Python
- MatplotLib
- Anaconda
- Jupyter Notebook

IV. Desarrollo de la práctica:

Introduccion a MatplotLib

MatplotLib Es una biblioteca que permite la creacion de figuras y graficos de calidad mediante el suso de Python.

- Permite la creaciom de graficos de manera sencilla.
- Permite la integracion de graficos y figuras en un Jupyter Notebook.

Import

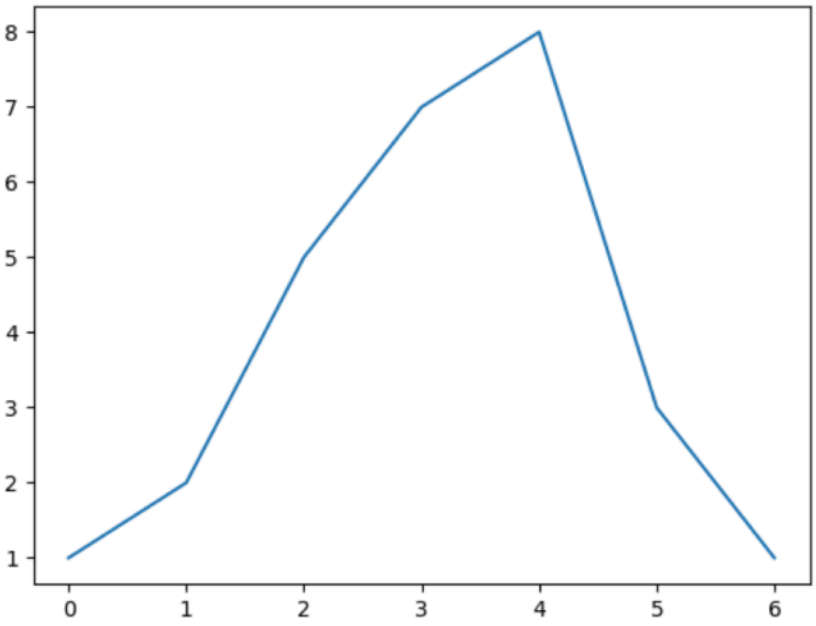
```
import matplotlib
import matplotlib.pyplot as plt
```

```
# Muestra los graficos integrados dentro de Jupyter Notebook
%matplotlib inline
```

Representacion de grafica de Datos

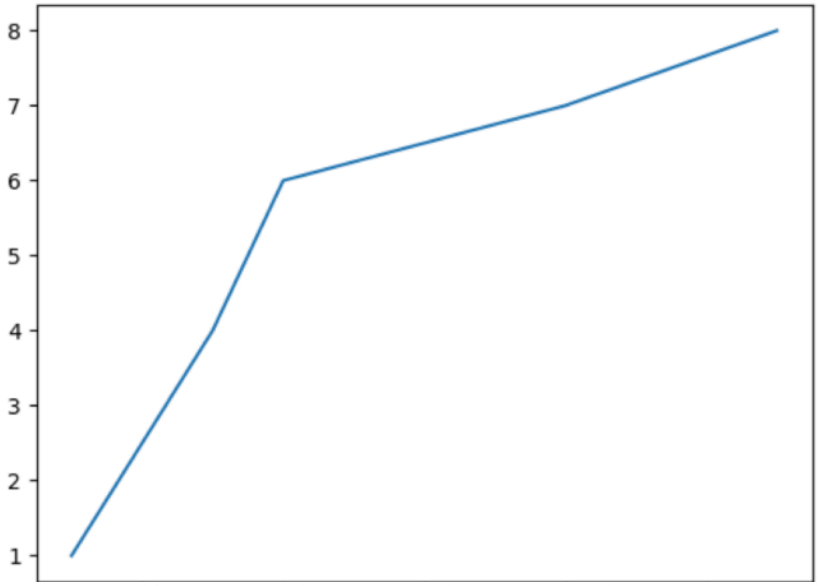
Si a la funcion de trazado se le da una matriz de datos, la usara como coordenas en eje vertical, y utilizara el indice de cada punto de datos en el array como la coordenada horizontal.

```
plt.plot([1, 2, 5, 7, 8, 3, 1])
plt.show()
```



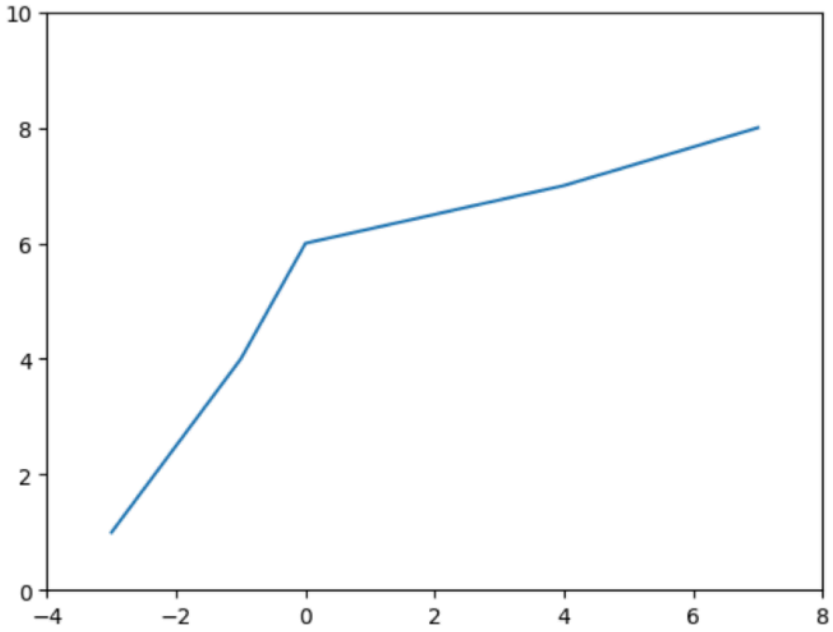
Tambien se pueden proporcionar dos matrices: Una eje horizontal y otra eje vertical

```
plt.plot([-3, -1, 0, 4, 7],[1, 4, 6, 7, 8])
plt.show()
```



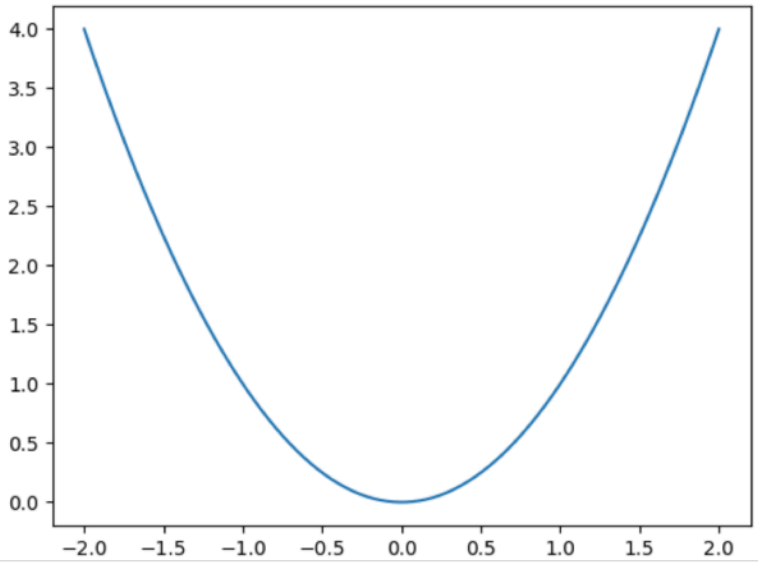
Pueden modificarse las longitudes de los ejes para que la figura no se vea tan ajustada

```
plt.plot([-3, -1, 0, 4, 7], [1, 4, 6, 7, 8])
plt.axis([-4, 8, 0, 10]) # [xmin, xmax, ymin, ymax]
plt.show()
```

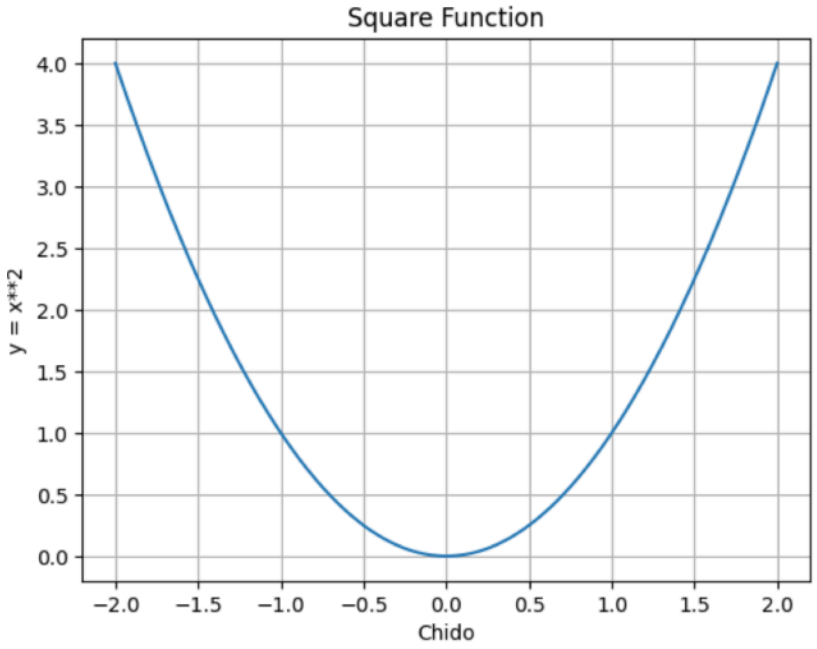


Se sigue el mismo procedimeinto para pintar una funcion matematica.

```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2

plt.plot(x, y)
plt.show()
```



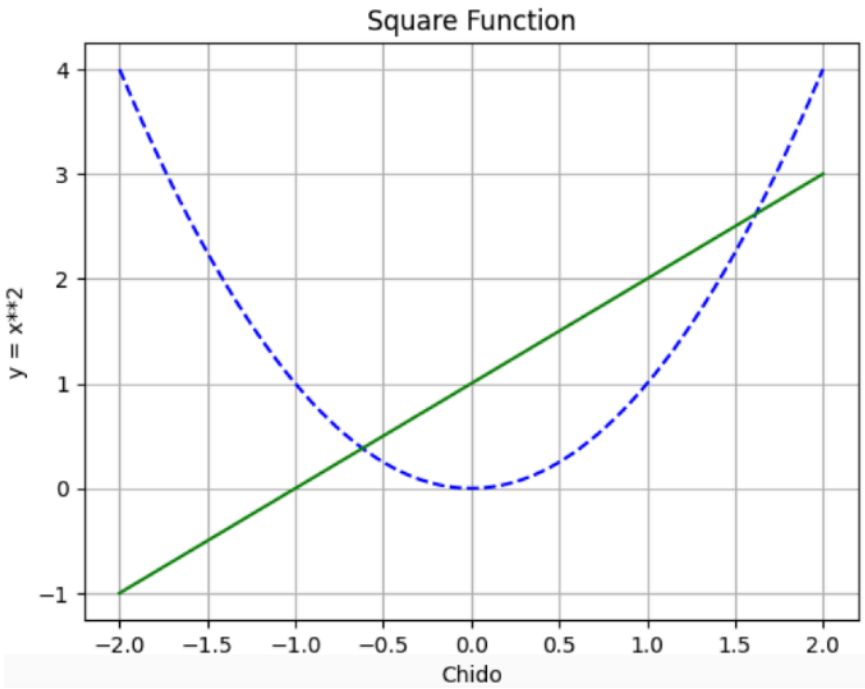

```
plt.plot(x, y)
plt.title("Square Function")
plt.ylabel("y = x**2")
plt.xlabel("Chido")
plt.grid(True)
plt.show()
```



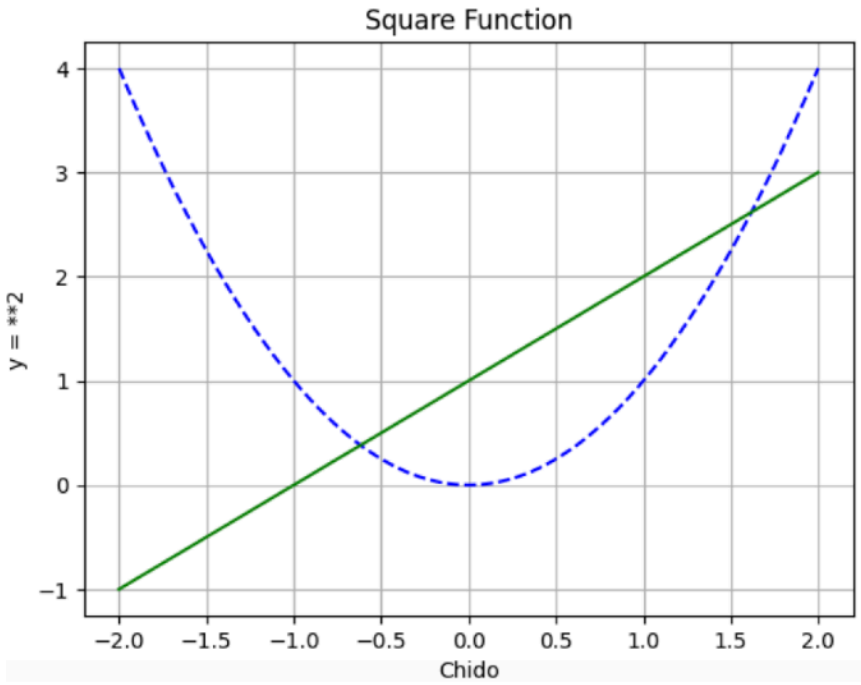
Pueden superponerse graficas y cambiar el estilo de las funciones.

```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x+1

plt.title("Square Function")
plt.ylabel("y = x**2")
plt.xlabel("Chido")
plt.grid(True)

plt.plot(x, y, 'b--', x, y2, 'g')
plt.plot()
```

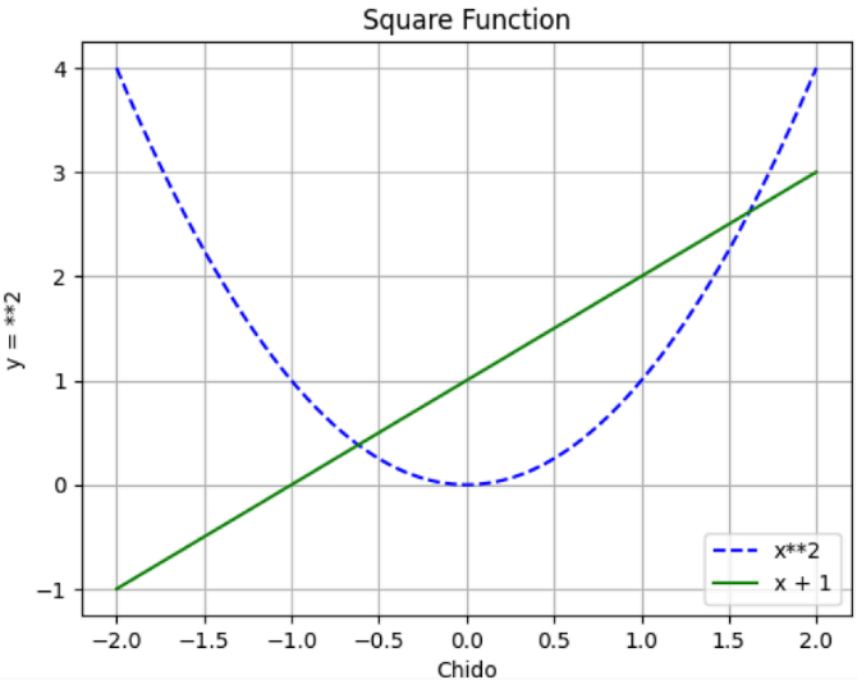
```
[]
```



```
# Separado en distintas lineas las funciones, asi se logra comprender un poco mejor
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

plt.title("Square Function")
plt.xlabel("Chido")
plt.ylabel("y = **2")
plt.grid(True)

plt.plot(x, y, 'b--')
plt.plot(x, y2, 'g')
plt.show()
```

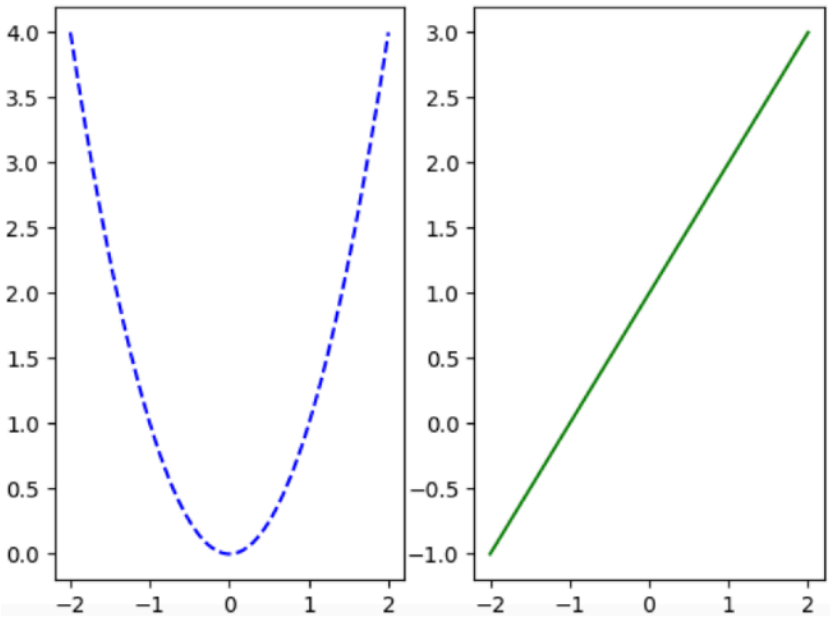


```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

plt.title("Square Function")
plt.xlabel("Chido")
plt.ylabel("y = **2")
plt.grid(True)

plt.plot(x, y, 'b--', label = "x**2") # El label indica por asi decirlo a cual pertenece
plt.plot(x, y2, 'g', label = "x + 1")
plt.legend(loc = "best") # La situa en la mejor localización
plt.show()
```



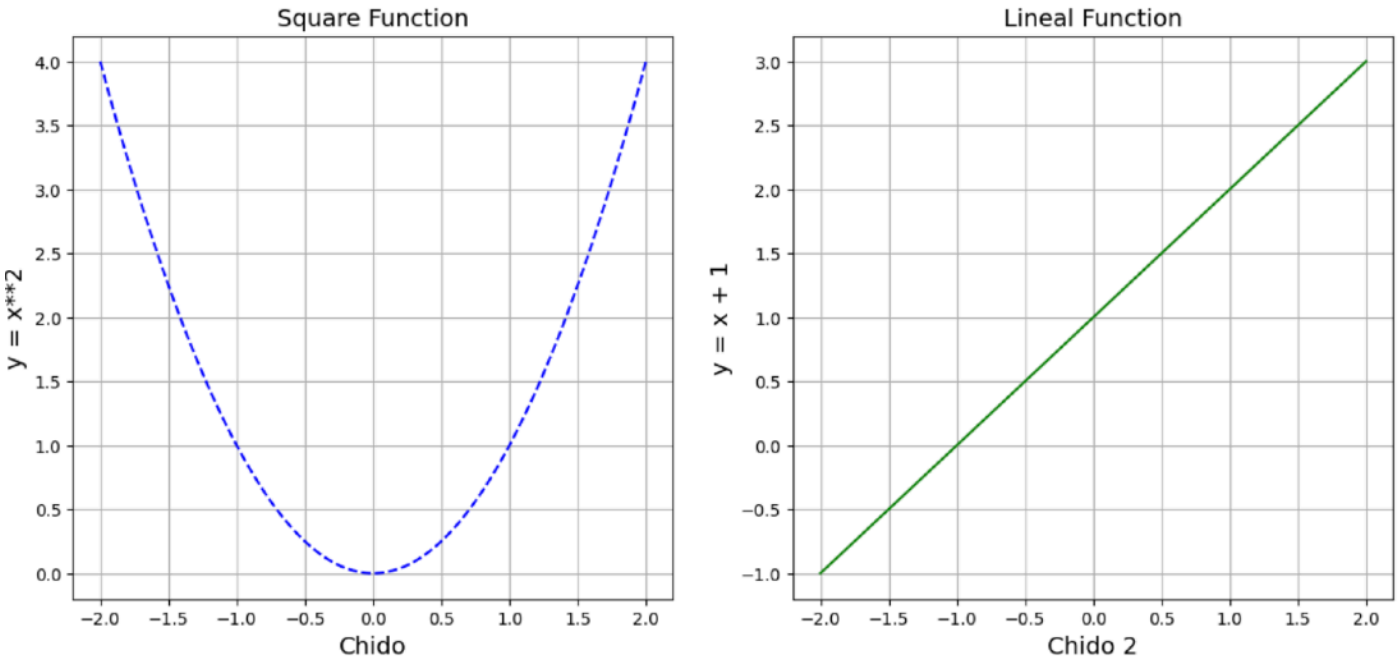
Tambien pueden crearse dos graficas que no se superpongan. Estas graficas de organizan en un grid y se denominan subplots.

```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 =  x + 1

plt.subplot(1, 2, 1) # 1 Row, 2 Columns, 1st Subplot
plt.plot(x, y, 'b--')

plt.subplot(1, 2, 2) # 1 Row, 2 Columns, 2nd Subplot
plt.plot(x, y2, 'g')

plt.show()
```



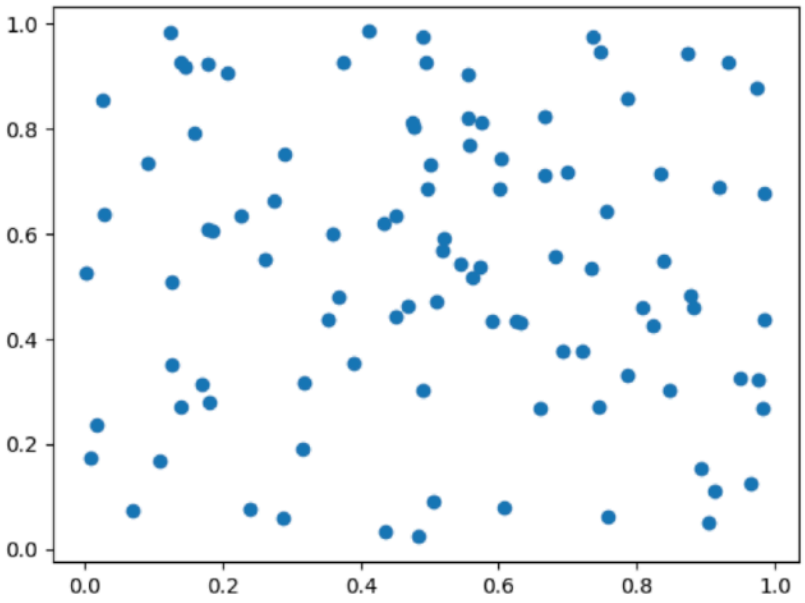
Para que las graficas no queden tan ajustadas, se puede hacer la figura las grande.

```
plt.figure(figsize = (14, 6))


plt.subplot(1, 2, 1) # 1 Row, 2 Columns, 1st Subplot
plt.plot(x, y, 'b--')
plt.title("Square Function", fontsize= 14)
plt.xlabel("Chido", fontsize= 14)
plt.ylabel("y = x**2", fontsize= 14)
plt.grid(True)


plt.subplot(1, 2, 2) # 1 Row, 2 Columns, 2nd Subplot
plt.plot(x, y2, 'g')
plt.title("Lineal Function", fontsize= 14)
plt.xlabel("Chido 2", fontsize= 14)
plt.ylabel("y = x + 1", fontsize= 14)
plt.grid(True)

plt.show()
```

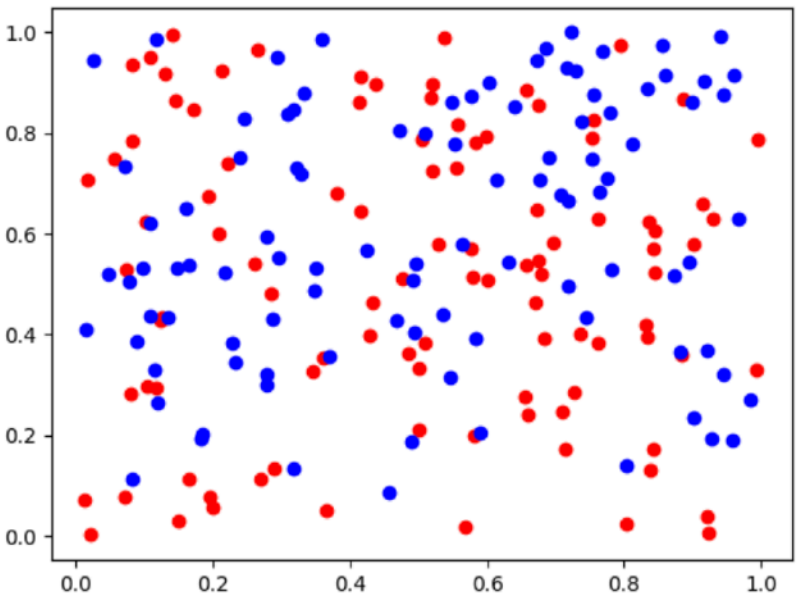


Scatter Plots

```
from numpy.random import rand
x, y = rand(2, 100)
plt.scatter(x, y)
plt.show()
```

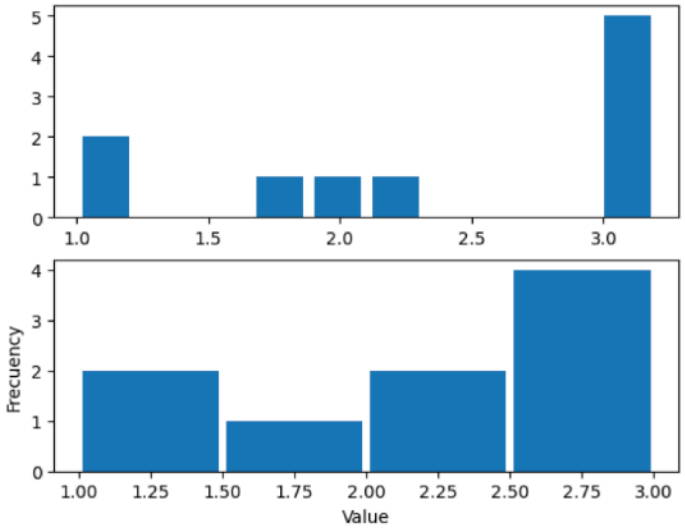



```
from numpy.random import rand
x, y = rand(2, 100)
x2, y2 = rand(2, 100)

plt.scatter(x, y, c = 'red')
plt.scatter(x2, y2, c = 'blue')
plt.show()
```



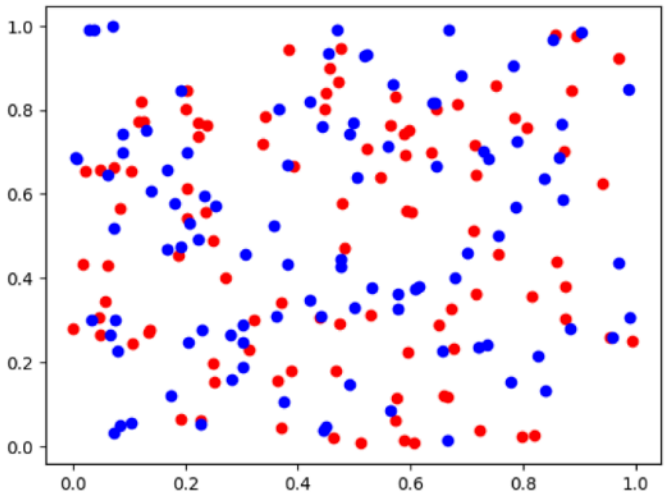
Histogramas

```
data = [1, 1.1, 1.8, 2, 2.1, 3.2, 3, 3, 3, 3]
plt.subplot(211)
plt.hist(data, bins = 10, rwidth = 0.8)

plt.subplot(212)
plt.hist(data, bins = [1, 1.5, 2, 2.5, 3], rwidth = 0.95)
plt.xlabel("Value")
plt.ylabel("Frecuency")

plt.show()
```



Guardar figuras

```
from numpy.random import rand
x, y = rand(2, 100)
x2, y2 = rand(2, 100)

plt.scatter(x, y, c = 'red')
plt.scatter(x2, y2, c = 'blue')

plt.savefig("3502_Mi_Grafica_Chida.png", transparent = True)
```



V. Conclusiones:

En esta practica aprendimos a graficar números pseudo aleatorios, ya que estos se pueden generar fácilmente con herramientas como bibliotecas como las que usamos de numpy y después con matplotlib decirle que tipo de grafica es la que queremos y indicarle mas cosas que este grafico puede llevar para que resulte mas fácil su comprensión y asi mismo sea mas llamativo, la verdad me gusto que se puede poner mas de un grafico ya que eso se puede usar en distintas cosas como el monitoreo de componentes y que muestre los datos y cosas asi.