TEL101 - Iniciación a la Programación

Clase 10: Matplotlib

Prof. Nicolás Torres
nicolas.torresr@usm.cl
Ingeniería Civil Telemática
Departamento de Electrónica
Universidad Técnica Federico Santa María

Matplotlib

- Es una biblioteca para la generación de gráficos en el lenguaje de programación Python.
- Permite crear gráficos de calidad científica con solo unas pocas líneas de código.

Instalación

- Matplotlib no viene por defecto cuando se descarga Python, por lo tanto, hay que instalarla aparte.
- Se puede descargar desde https://matplotlib.org/.
- Aunque, es recomendable instalar usando `pip`, una herramienta escrita en Python para facilitar la descarga e instalación de paquetes del lenguaje.
- En una celda de Jupyter Notebook, puede ejecutar `!pip install matplotlib`.

Importar

In [1]:
```
import matplotlib
```

Gráficos

- Un gráfico es una representación visual de datos. La visualización de los datos por medio de gráficos ayuda a detectar patrones, tendencias, relaciones y permite explorar los datos.
- Matplotlib posee un paquete llamado `pyplot` que contiene un conjunto de funciones para generar gráficos.

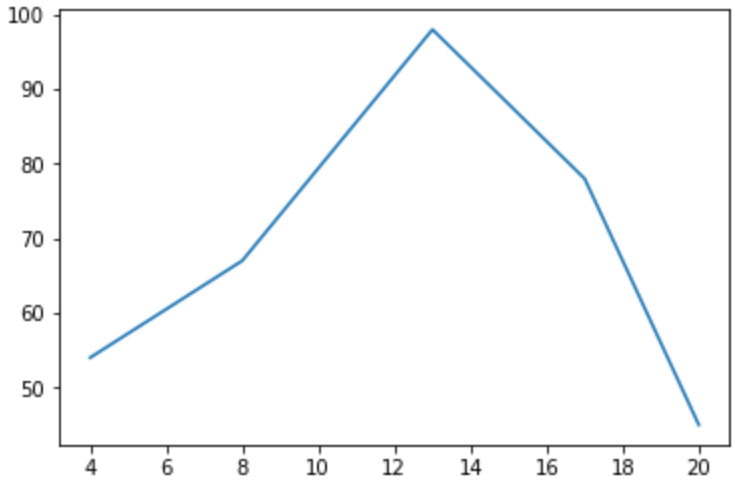
In [2]:
```
import matplotlib.pyplot as plt
```

Gráfico de puntos

- La función `plot(x, y)` se usa para dibujar puntos (marcadores).

- Recibe como argumentos secuencias numéricas con las coordenadas horizontales y verticales de los puntos.

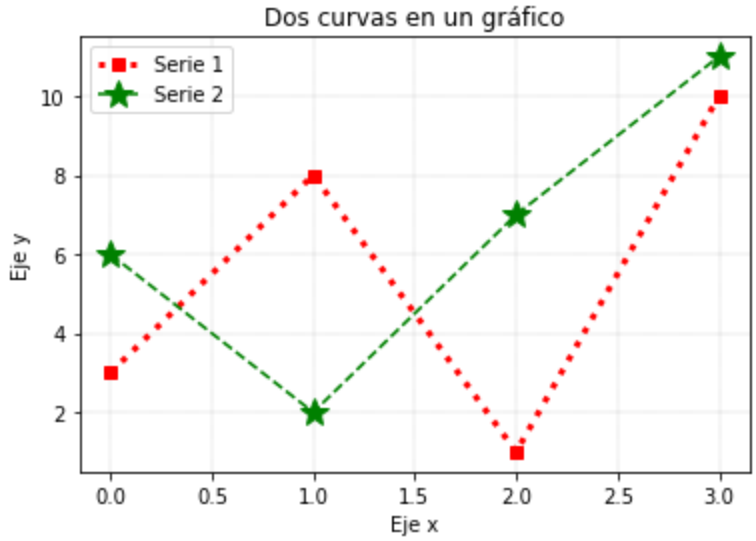
In [9]:
```
x = [4, 8, 13, 17, 20]
y = [54, 67, 98, 78, 45]
plt.plot(x,y)
plt.show()
```



Parámetros opcionales

- El parámetro `x` es opcional. Por defecto, `x` es igual a `range(len(y))`.
- Un tercer argumento opcional es un texto que establece el estilo de los marcadores y la línea que los une en formato `'[marcador][linea][color]'`.
- El tamaño del marcador puede controlarse con el parámetro `ms`, mientras que el color del borde con `mec` y el color de relleno con `mfc`. En tanto, el grosor de la línea puede controlarse con el parámetro `lw`.
- Puede usar las funciones `xlabel()` e `ylabel()` para establecer una etiqueta para cada eje y `title()` para agregar un título al gráfico. Además, puede usar la función `grid()` para incluir líneas de cuadrícula en el gráfico.
- Para conocer todos los parámetros y estilos visite https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html.

In [4]:
```
y1 = [3, 8, 1, 10]
y2 = [6, 2, 7, 11]
plt.plot(y1, 's:r', label='Serie 1', lw=3)
plt.plot(y2, '*--g', label='Serie 2', ms=15)
plt.title("Dos curvas en un gráfico")
plt.xlabel("Eje x")
plt.ylabel("Eje y")
plt.grid(color = 'gray', linestyle = '--', linewidth = 0.2)
plt.legend()
plt.show()
```



Guardar la figura

La función `savefig()` permite guardar los gráficos en un archivo. Se recomienda formato PDF para máxima calidad.

In [5]:
```
x = [4, 8, 13, 17, 20]
y = [54, 67, 98, 78, 45]

plt.plot(x,y,'-oy',ms=10,mec='r',mfc='c')
plt.savefig("grafico.pdf")
plt.show()
```

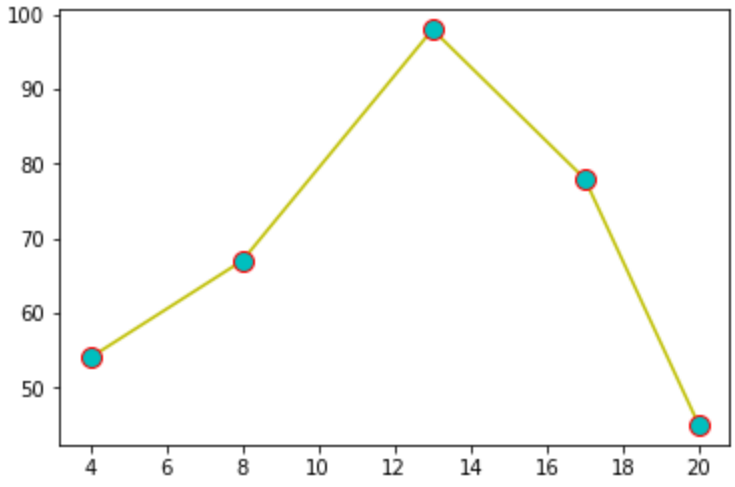


Gráfico de dispersión

- Utiliza las coordenadas cartesianas para mostrar los valores de dos variables para un conjunto de datos.
- La función `scatter(x, y)` se usa para dibujar puntos (marcadores). Recibe como argumentos secuencias numéricas con las coordenadas horizontales y verticales de los puntos. Ambos parámetros son obligatorios.

In [6]:
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
```

```
plt.scatter(x, y)
plt.title("Gráfico de dispersión")
plt.show()
```

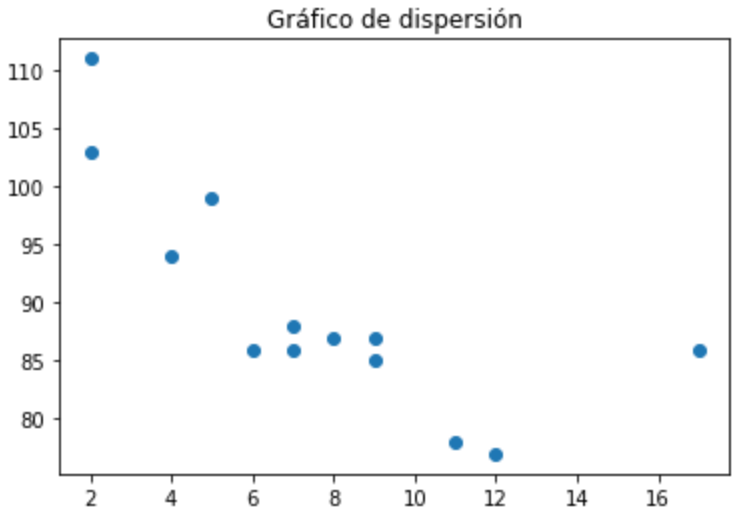
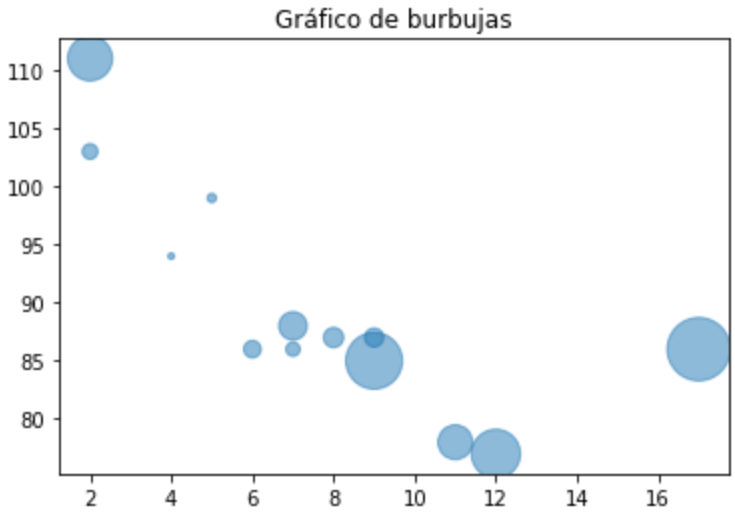


Gráfico de burbujas

- Es una variación de un gráfico de dispersión en el que los puntos de datos se reemplazan por burbujas y una dimensión adicional de los datos se representa en el tamaño de las burbujas.
- La función `scatter(x, y, s)` recibe de manera opcional un tercer argumento `s` con el tamaño de los marcadores. Por defecto, es un entero que establece el mismo tamaño para todos los puntos, pero acepta una secuencia con distintos valores. Además, un parámetro opcional `alpha`, que oscila entre 0 (transparente) y 1 (opaco), establece el nivel de transparencia de los marcadores.

In [7]:
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
s = [20,50,100,200,500,1000,60,90,10,300,600,800,75]
plt.scatter(x, y, s, alpha=0.5)
plt.title("Gráfico de burbujas")
plt.show()
```



Burbujas de colores

- Permite representar una cuarta dimensión con los colores de las burbujas.
- La función `scatter(x, y, s, c)` recibe de manera opcional un cuarto argumento `c` con el color de los marcadores. Puede ser un escalar o secuencia de números que serán mapeados a una escala de colores `cmap`, una secuencia de colores RGB o RGBA, o un simple texto con el nombre del color.

In [8]:
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
s = [20,50,100,200,500,1000,60,90,10,300,600,800,75]
c = [89,75,37,69,15,63,61,18,72,92,3,70,56]
plt.scatter(x, y, s, c, alpha=0.5, cmap='nipy_spectral')
plt.title("Gráfico de burbujas de colores")
plt.colorbar()
plt.show()
```

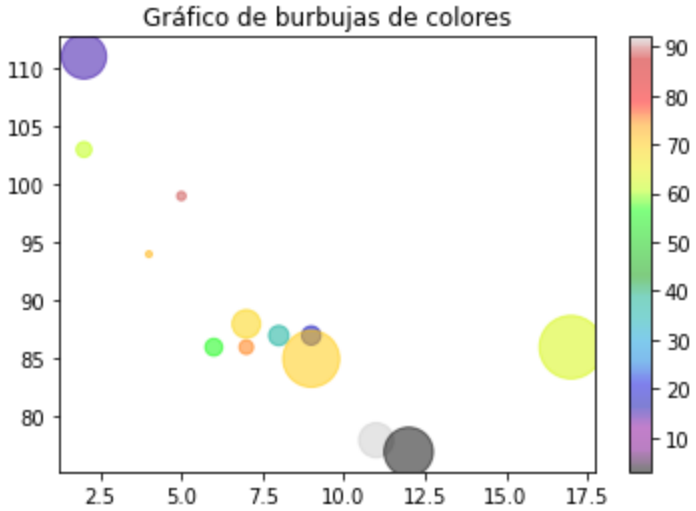


Gráfico de barras

- Permiten comparar, con el tamaño de la barras, frecuencias de una variable categórica.
- La función `bar(x, height)` se usa para dibujar barras. Recibe como primer argumento, `x`, una secuencia con las etiquetas horizontales de las barras y como segundo argumento `height`, el tamaño vertical de las barras. Ambos parámetros son obligatorios.

In [9]:
```
x = ["A", "B", "C", "D"]
height = [3, 8, 1, 10]
plt.bar(x, height)
plt.title("Gráfico de barras")
plt.show()
```

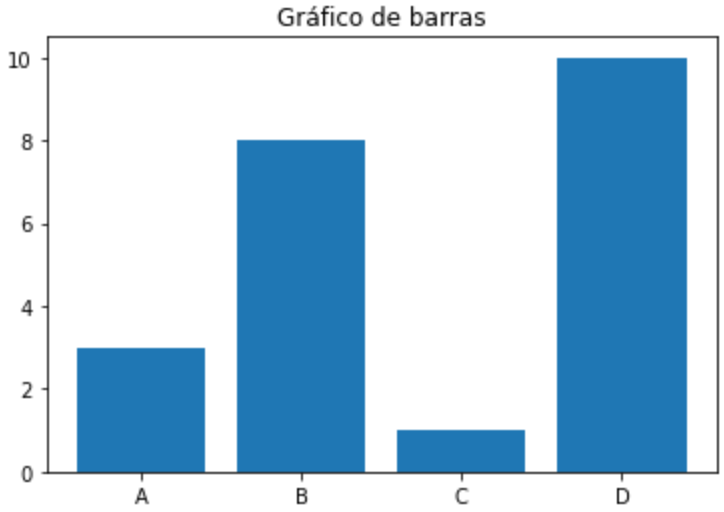
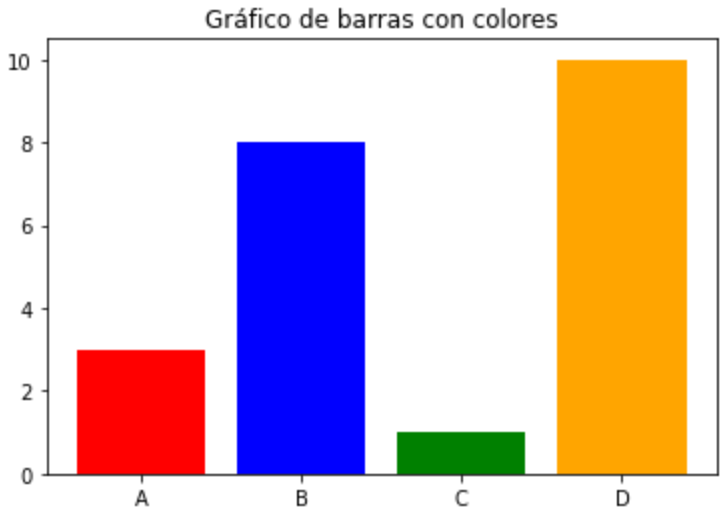


Gráfico de barras de colores

La función `bar(x, height)` además permite otros parámetros como el ancho de la barra y el color. Para conocer todos los parámetros en profundidad puede visitar la documentación oficial de la función https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar.html.

In [10]:
```
x = ["A", "B", "C", "D"]
height = [3, 8, 1, 10]
plt.bar(x, height, color=["red", "blue", "green", "orange"])
plt.title("Gráfico de barras con colores")
plt.show()
```



Histogramas

- Representa la distribución de frecuencias de los valores de una variable cuantitativa, agrupados en intervalos numéricos.
- La función `hist(x)` se usa para dibujar *bins*. Recibe como primer argumento obligatorio, `x`, una secuencia con todos los valores numéricos que se agruparán en una cierta cantidad de barras o "bins". Este argumento es opcional, así como el color y la orientación. Para conocer todos los parámetros en profundidad puede visitar la documentación oficial de la función https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html.

In [11]:
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
plt.hist(x, bins=3, edgecolor = "black")
plt.title("Histograma")
plt.show()
```

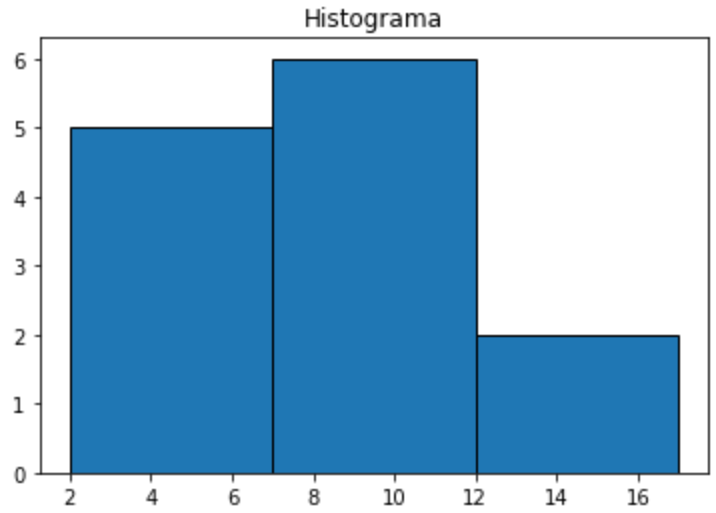
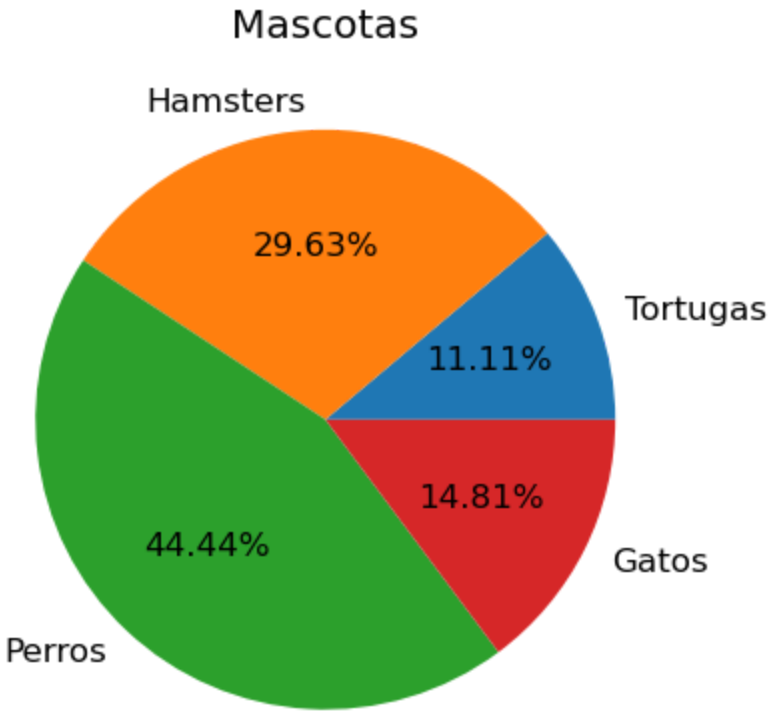


Gráfico circular

- Conocido popularmente como "gráfico de torta" o "*pie*", se utiliza para representar porcentajes y proporciones de los datos con respecto al total.
- La función `pie(x)` se usa para dibujar sectores en un gráfico circular.
  - Recibe como primer argumento obligatorio, `x`, una secuencia con todos los valores numéricos que se agruparán en una cierta cantidad de trozos de la torta.
  - Como segundo parámetro, opcional, pero recomendable, está `labels`, que es una secuencia de textos con las etiquetas de cada sector.
  - Un tercer argumento utilizado es `autopct` que permite etiquetar los sectores con un valor numérico. Generalmente, se usa para etiquetar cada sector con el porcentaje del valor con respecto al total.
- Para conocer todos los parámetros en profundidad puede visitar la documentación oficial de la función https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.pie.html.

In [12]:
```
plt.figure(dpi=120)
x = [15, 40, 60, 20]
etiquetas = ["Tortugas","Hamsters","Perros","Gatos"]
plt.pie(x, labels=etiquetas, autopct="%.2f%%")
plt.title("Mascotas")
plt.show()
```


Mascotas
Hamsters
29.63%
Tortugas
11.11%
14.81%
Gatos
44.44%
Perros