Matplotlib

Programación II

Matplotlib

Programación II

Matplotlib es una biblioteca de Python que se utiliza para crear gráficos y visualizaciones en 2D. Esta biblioteca es una de las más utilizadas para la creación de gráficos en Python, ya que ofrece una gran variedad de herramientas y https://pypi.org/project/matplotlib/ características para ayudar a los usuarios a crear gráficos con la mayor facilidad posible.

Esta biblioteca es compatible con diversos formatos de archivos, como PNG, JPG, EPS, SVG, PDF y muchos más. Esta biblioteca también ofrece una gran variedad de funciones para manipular los datos, así como para ajustar y personalizar los gráficos.

(Matplotlib, 2023)

Matplotlib (Algunas Funcionalidades)

Programación II

1. Estilos de gráficos personalizables: Matplotlib proporciona una amplia variedad de estilos de gráficos, desde gráficos de líneas simples hasta gráficos de barras, gráficos de dispersión, gráficos de área, gráficos de histogramas, gráficos de caja y bigotes, y gráficos de contorno.

2. Personalización de gráficos: Matplotlib proporciona una variedad de herramientas para personalizar los gráficos, desde etiquetas y leyendas hasta anotaciones y ejes.

3. Exportación de gráficos: Matplotlib le permite exportar gráficos a una variedad de formatos, como PNG, PDF, SVG y EPS.

Matplotlib

Programación II

4. Soporte para trabajar con datos de pandas: Matplotlib proporciona una API que le permite trabajar directamente con objetos de pandas para crear gráficos. Esto le ayuda a ahorrar tiempo al no tener que convertir los datos a un formato diferente antes de crear los gráficos.

5. Soporte para trabajar con varias figuras: Matplotlib le permite crear y trabajar con varias figuras al mismo tiempo, lo que le permite crear gráficos más complejos y complejos.

6. Soporte para animaciones: Matplotlib proporciona una API para crear animaciones, lo que le permite crear gráficos animados para representar los cambios en los datos a lo largo del tiempo.

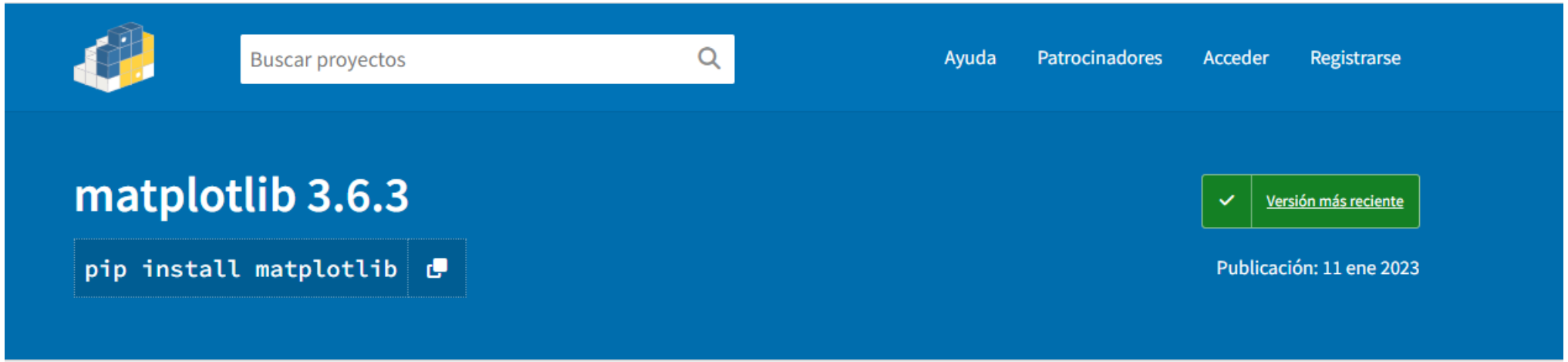
Matplotlib

Programación II

- *Documentación Oficial*
  - *https://matplotlib.org/*
- *Guia de matplotlib*
  - *https://pypi.org/project/matplotlib/*

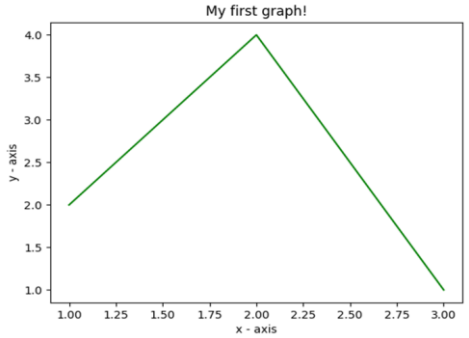
Matplotlib (Instalación)

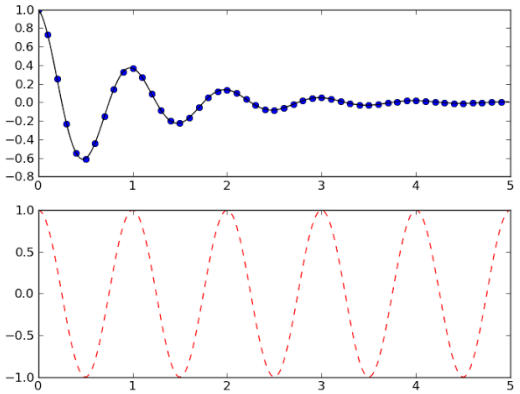
Programación II

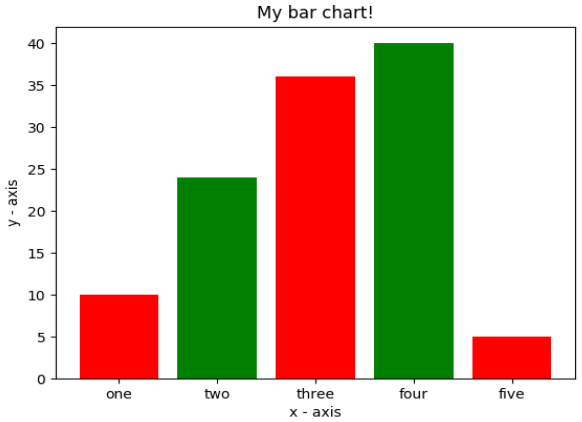


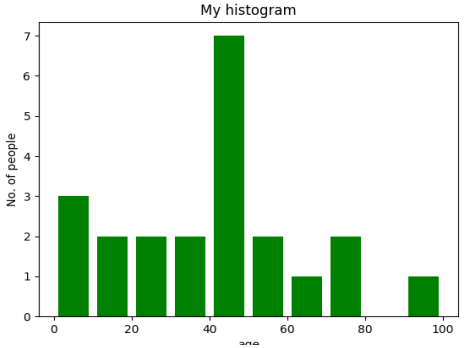
Matplotlib

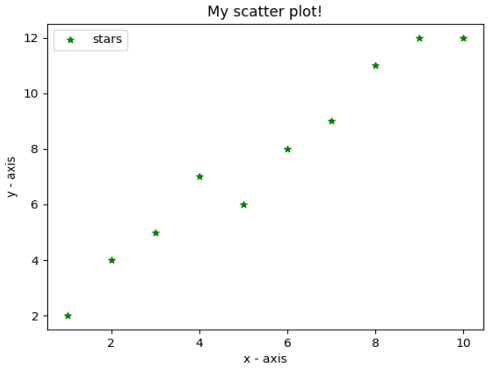
Programación II

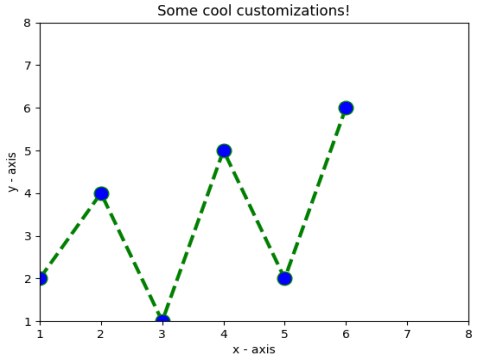


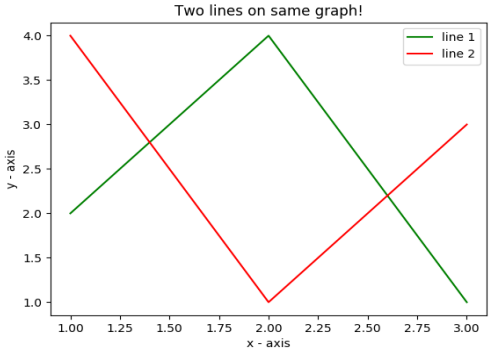








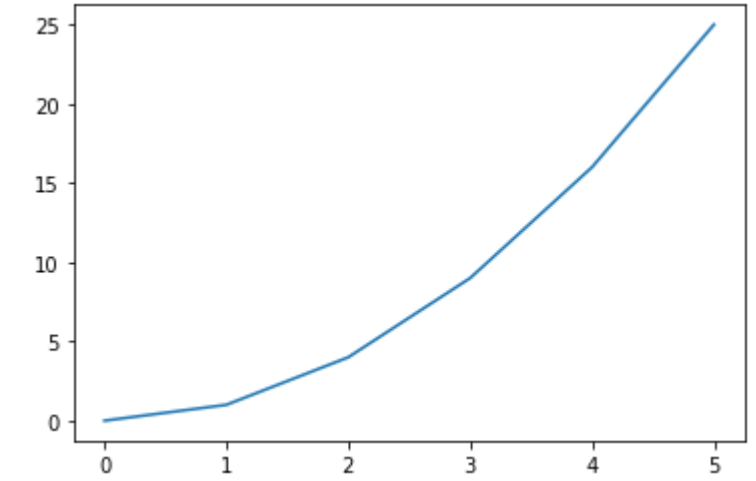




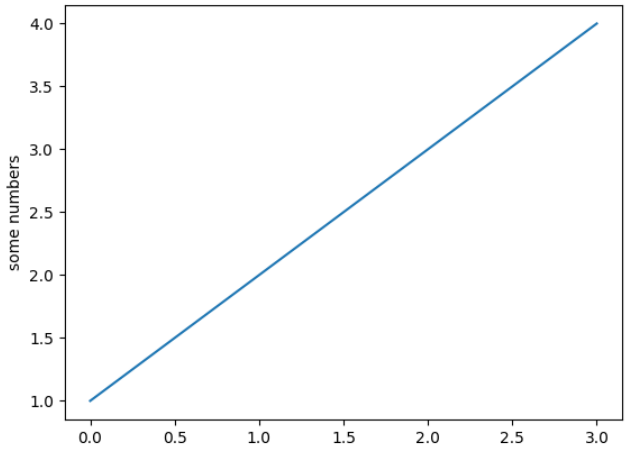
Matplotlib (Grafíca de linea)

Programación II

```
import matplotlib.pyplot as plt

#create data for plotting
x_values = [0, 1, 2, 3, 4, 5 ]
y_values = [0, 1, 4, 9, 16,25]

#the default graph style for plot is a line
plt.plot(x_values, y_values)

#display the graph
plt.show()
```




```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```



Matplotlib

Programación II

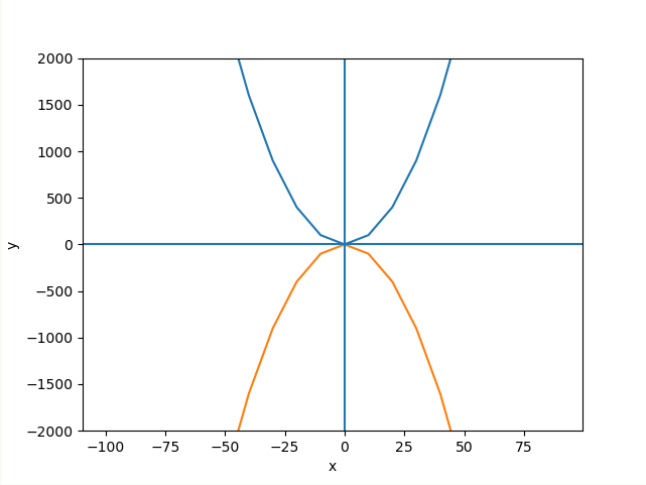
- ***text()*** *: Añade texto en una localización arbitraría*
- ***xlabel()*** *: Añade texto en eje-x*
- ***ylabel()*** *: Añade texto en eje-y*
- ***title()*** *: Añade titulo al grafico*
- ***clear()*** *: Remueve todos las grafícas y ejees*
- ***savefig()****: Guardad la figura en un archivo*
- ***legend()*** *: Muestra las legendas en el grafíco*

Todos los métodos están disponibles en ***pyplot*** *y en la instancia de ejes en general.*

Matplotlib

Programación II

```
import matplotlib.pyplot as plt
y1 =[]
y2 =[]
x = range(-100,100,10)
for i in x: y1.append(i**2)
for i in x: y2.append(-i**2)

plt.plot(x, y1)
plt.plot(x, y2)
plt.xlabel("x")
plt.ylabel("y")
plt.ylim(-2000, 2000)
plt.axhline(0) # horizontal line
plt.axvline(0) # vertical line

plt.savefig("quad.png")

plt.show()
```

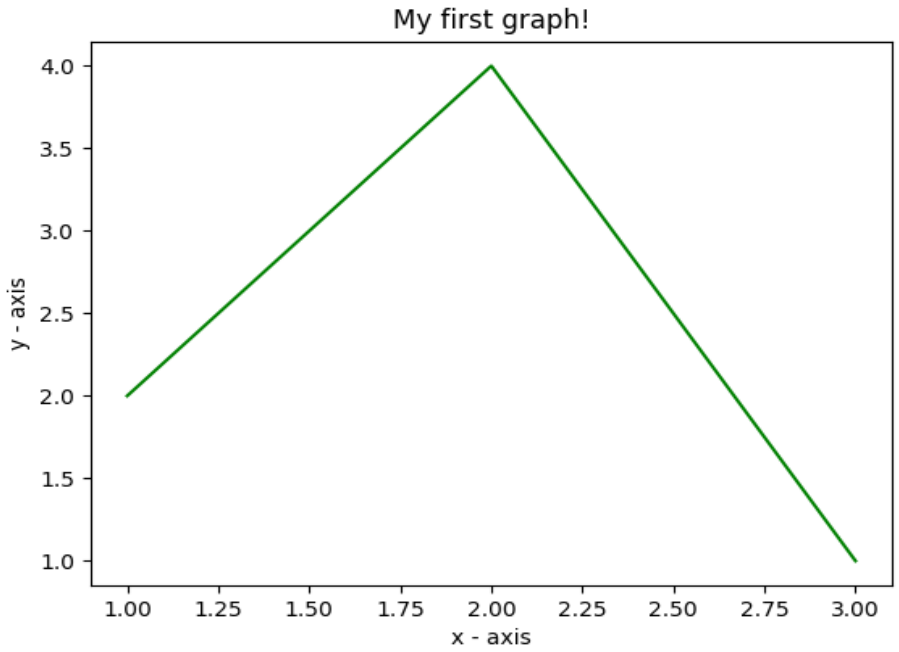


Matplotlib

Programación II

Simple line

```
# importing the required module
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3]
# corresponding y axis values
y = [2,4,1]

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('My first graph!')

# function to show the plot
plt.show()
```



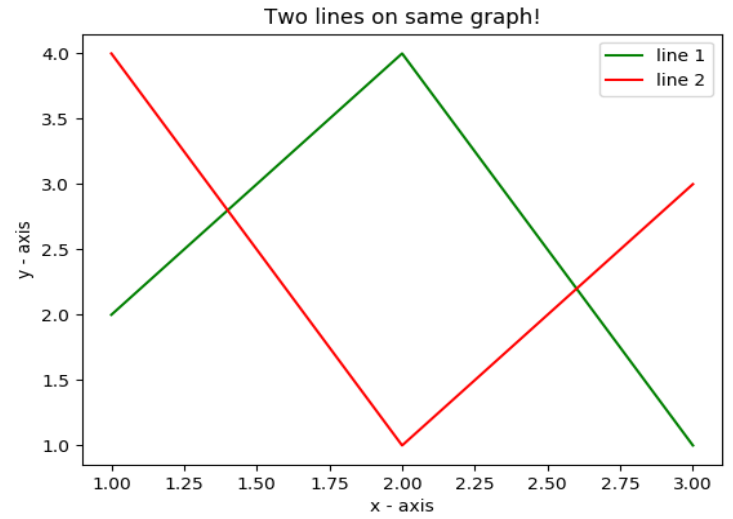
- Define the **`x-axis`** and corresponding **`y-axis`** values as lists.
- Plot them on canvas using **`.plot()`** function.
- Give a name to x-axis and y-axis using **`.xlabel()`** and **`.ylabel()`** functions.
- Give a title to your plot using **`.title()`** function.
- Finally, to view your plot, we use **`.show()`** function.

Matplotlib

Programación II

Simple 2 lines

```
import matplotlib.pyplot as plt

# line 1 points
x1 = [1,2,3]
y1 = [2,4,1]
# plotting the line 1 points
plt.plot(x1, y1, label="line 1")

# line 2 points
x2 = [1,2,3]
y2 = [4,1,3]
# plotting the line 2 points
plt.plot(x2, y2, label = "line 2")

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')
# giving a title to my graph
plt.title('Two lines on same graph!')

# show a legend on the plot
plt.legend()

# function to show the plot
plt.show()
```



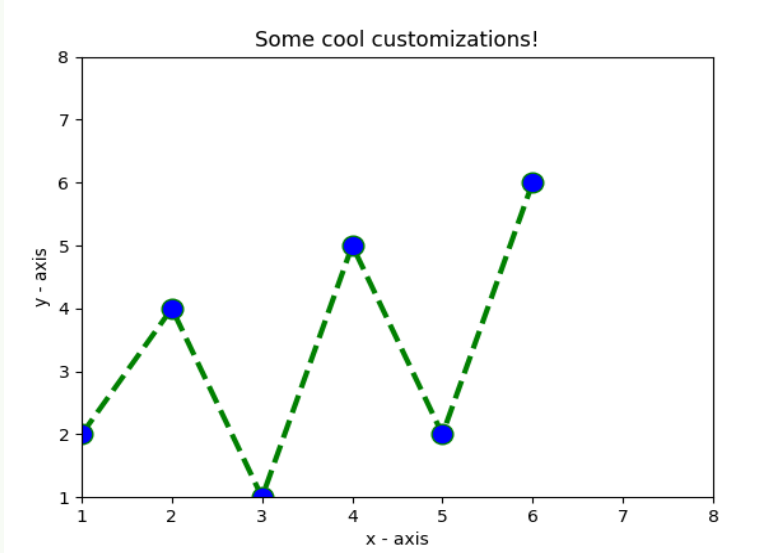
- Here, we plot two lines on same graph. We differentiate between them by giving them a name(label) which is passed as an argument of `.plot()` function.
- The small rectangular box giving information about type of line and its color is called legend. We can add a legend to our plot using `.legend()` function.

Matplotlib

Programación II

Customization of Plots

```
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3,4,5,6]
# corresponding y axis values
y = [2,4,1,5,2,6]

# plotting the points
plt.plot(x, y, color='green', linestyle='dashed', linewidth = 3,
         marker='o', markerfacecolor='blue', markersize=12)

# setting x and y axis range
plt.ylim(1,8)
plt.xlim(1,8)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('Some cool customizations!')

# function to show the plot
plt.show()
```

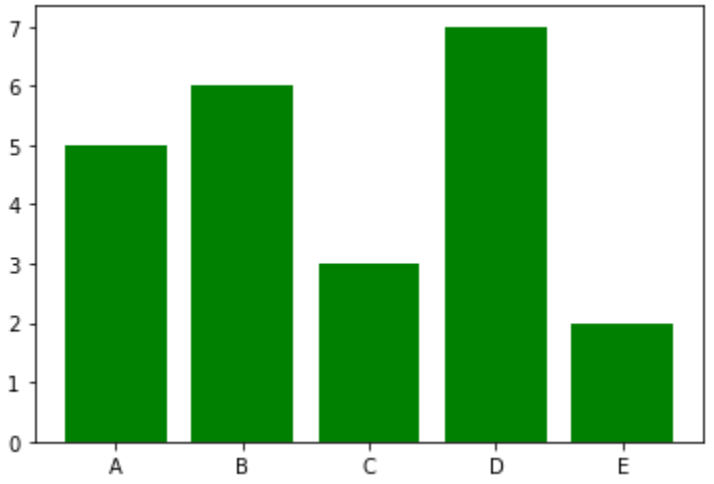


Matplotlib

Programación II

Bar graphs

```
import matplotlib.pyplot as plt

#Create data for plotting
values = [5, 6, 3, 7, 2]
names  = ["A", "B", "C", "D", "E"]

plt.bar(names, values, color="green")
plt.show()
```



- When using a bar graph, the change in code will be from `plt.plot()` to `plt.bar()` changes it into a bar chart.

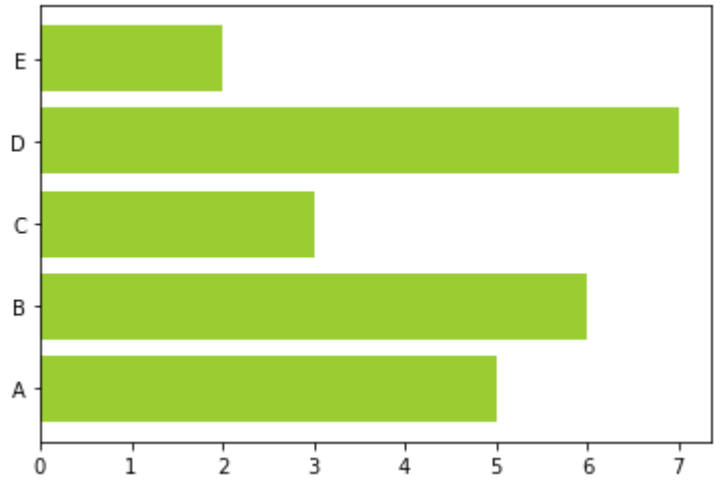
Matplotlib

Programación II

Bar graphs

We can also flip the bar graph horizontally with the following

```
import matplotlib.pyplot as plt

#Create data for plotting
values = [5,6,3,7,2]
names  = ["A", "B", "C", "D", "E"]

# Adding an "h" after bar will flip the graph
plt.barh(names, values, color="yellowgreen")
plt.show()
```

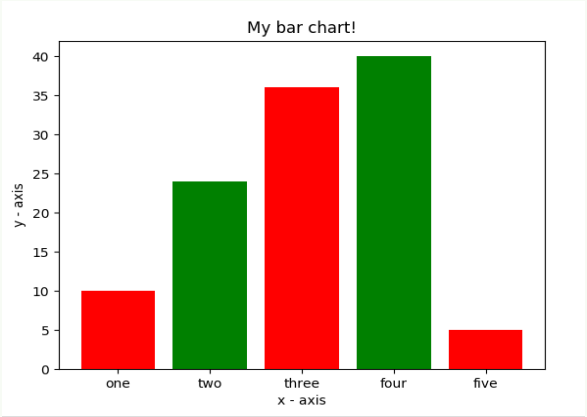


Matplotlib

Programación II

Bar Chart

```
import matplotlib.pyplot as plt


# heights of bars
height = [10, 24, 36, 40, 5]

# labels for bars
names = ['one','two','three','four','five']

# plotting a bar chart
c1 =['red', 'green']
c2 =['b', 'g'] # we can use this for color
plt.bar(names, height, width=0.8, color=c1)

# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('My bar chart!')

# function to show the plot
plt.show()
```



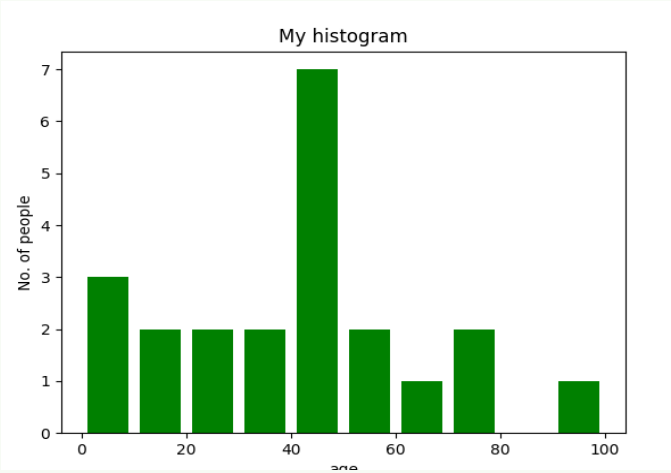
- Here, we use `plt.bar()` function to plot a bar chart.
- you can also give some name to x-axis coordinates by defining tick_labels

Matplotlib

Programación II

Histogram

```
import matplotlib.pyplot as plt

# frequencies
ages=[2,5,70,40,30,45,50,45,43,40,44,60,7,13,57,18,90,77,32,21,20,40]

# setting the ranges and no. of intervals
range = (0, 100)
bins = 10

# plotting a histogram
plt.hist(ages, bins, range, color='green',histtype='bar',rwidth=0.8)

# x-axis label
plt.xlabel('age')
# frequency label
plt.ylabel('No. of people')
# plot title
plt.title('My histogram')

# function to show the plot
plt.show()
```



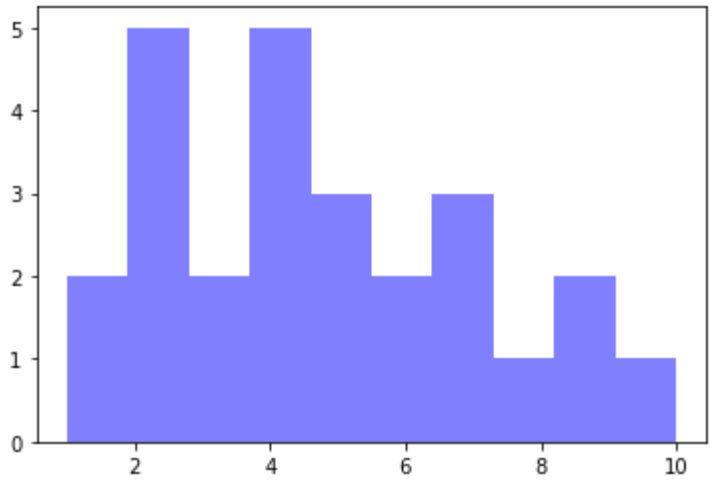
Matplotlib

Programación II

Histograms

```
import matplotlib.pyplot as plt

#generate fake data
x = [2,1,6,4,2,4,8,9,4,2,4,10,6,4,5,7,7,3,2,7,5,3,5,9,2,1]

#plot for a histogram
plt.hist(x, bins = 10, color='blue', alpha=0.5)
plt.show()
```

- *Looking at the code snippet, I added two new arguments:*
  - ***Bins*** *— is an argument specific to a histogram and allows the user to customize how many bins they want.*
  - ***Alpha*** *— is an argument that displays the level of transparency of the data points.*



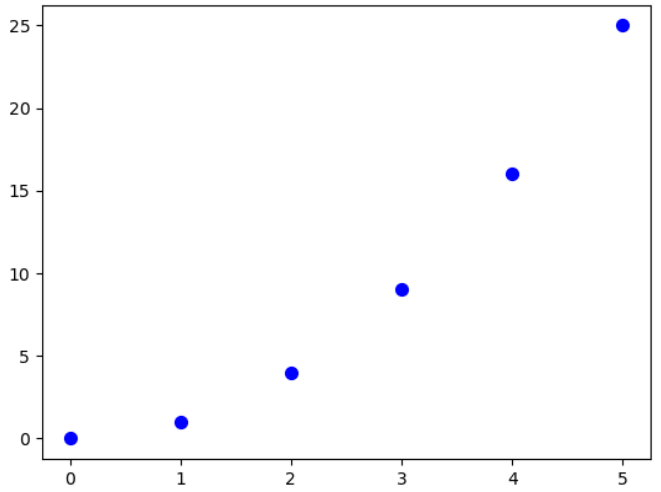
Matplotlib

Programación II

Scatter Plots

```
import matplotlib.pyplot as plt

#create data for plotting

x_values = [0,1,2,3,4,5]
y_values = [0,1,4,9,16,25]

plt.scatter(x_values, y_values, s=30, color="blue")
plt.show()
```

- *Can you see the pattern? Now the code changed from **plt.bar()** to **plt.scatter()**.*

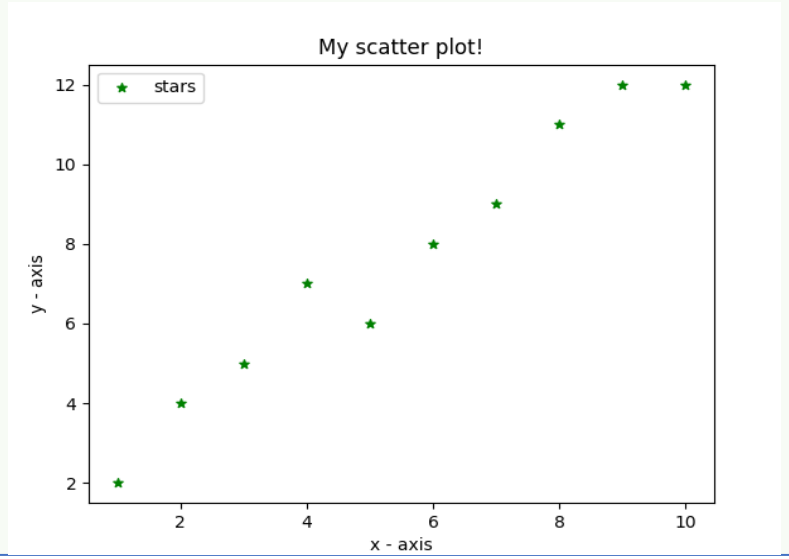


Matplotlib

Programación II

Scatter plot

```
import matplotlib.pyplot as plt

# x-axis values
x = [1,2,3,4,5,6,7,8,9,10]
# y-axis values
y = [2,4,5,7,6,8,9,11,12,12]

# plotting points as a scatter plot
plt.scatter(x, y, label= "stars", color="green", marker="*", s=30)

# x-axis label
plt.xlabel('x - axis')
# frequency label
plt.ylabel('y - axis')
# plot title
plt.title('My scatter plot!')
# showing legend
plt.legend()

# function to show the plot
plt.show()
```

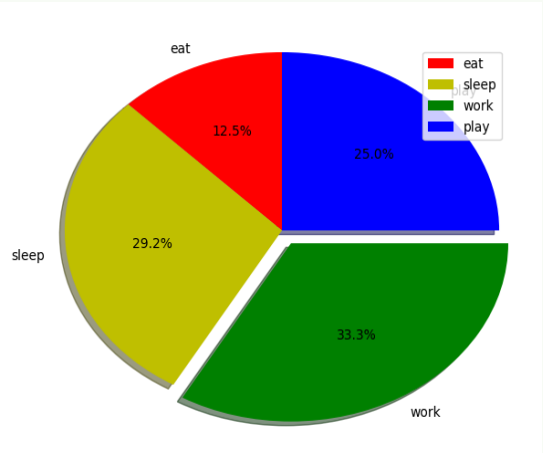


Matplotlib

Programación II

Pie-chart

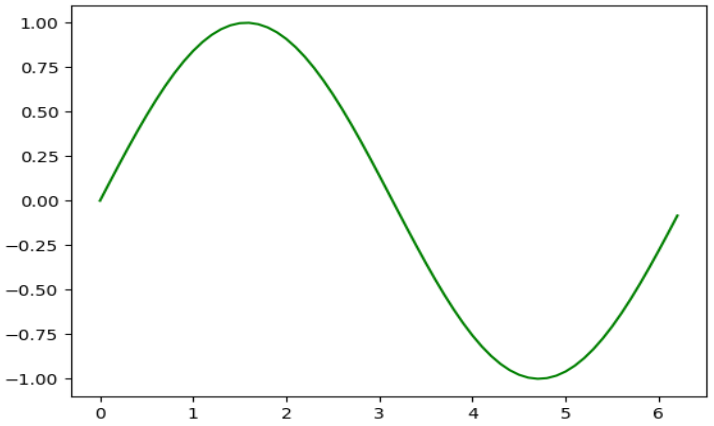
```
import matplotlib.pyplot as plt

# defining labels
activities = ['eat', 'sleep', 'work', 'play']

# portion covered by each label
slices = [3, 7, 8, 6]

# color for each label
colors = ['r', 'y', 'g', 'b']

# plotting the pie chart
plt.pie(slices, labels = activities, colors=colors,
        startangle=90, shadow = True, explode = (0, 0, 0.1, 0),
        radius = 1.2, autopct = '%1.1f%%')

# plotting legend
plt.legend()

# showing the plot
plt.show()
```



Matplotlib (Plotting curves of given equation)

Programación II

```
# importing the required modules
import matplotlib.pyplot as plt
import numpy as np

# setting the x - coordinates
x = np.arange(0, 2*(np.pi), 0.1)
# setting the corresponding y - coordinates
y = np.sin(x)

# potting the points
plt.plot(x, y)

# function to show the plot
plt.show()
```



Examples taken from:
Graph Plotting in Python | Set 1

Matplotlib

Programación II

- *Matplotlib: Visualization with Python*
  - *https://matplotlib.org/index.html*
- *matplotlib.pyplot*
  - *https://matplotlib.org/3.2.1/api/pyplot_summary.html*
- *Tutorials*
  - *https://matplotlib.org/tutorials/index.html*
- *Gallery & Examples*
  - *https://matplotlib.org/gallery/index.html*
- *Videos*
  - *https://www.youtube.com/watch?v=3Fp1zn5ao2M&feature=plcp*
- *Book: Mastering matplotlib*
  - *https://www.packtpub.com/big-data-and-business-intelligence/mastering-matplotlib*