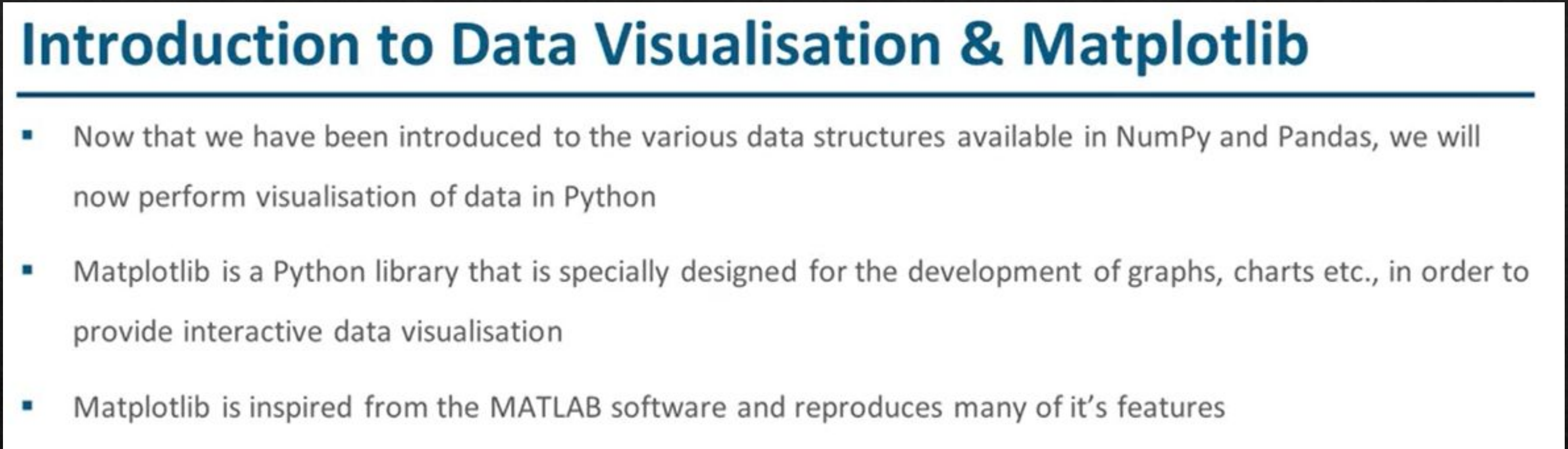
Data Science with Python

Noble Xavier



Introduction to Data Visualisation & Matplotlib

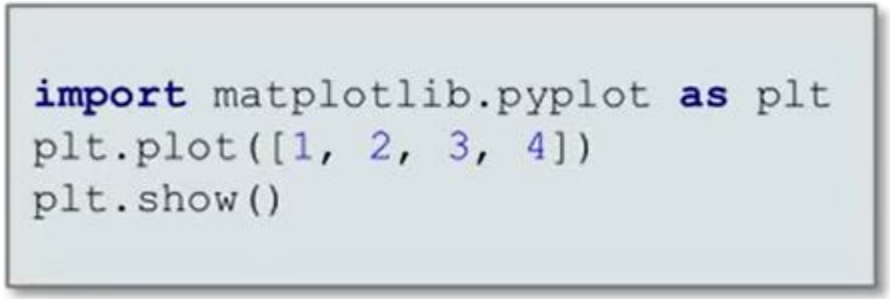
- Now that we have been introduced to the various data structures available in NumPy and Pandas, we will now perform visualisation of data in Python
- Matplotlib is a Python library that is specially designed for the development of graphs, charts etc., in order to provide interactive data visualisation
- Matplotlib is inspired from the MATLAB software and reproduces many of it’s features

First Plot with Matplotlib

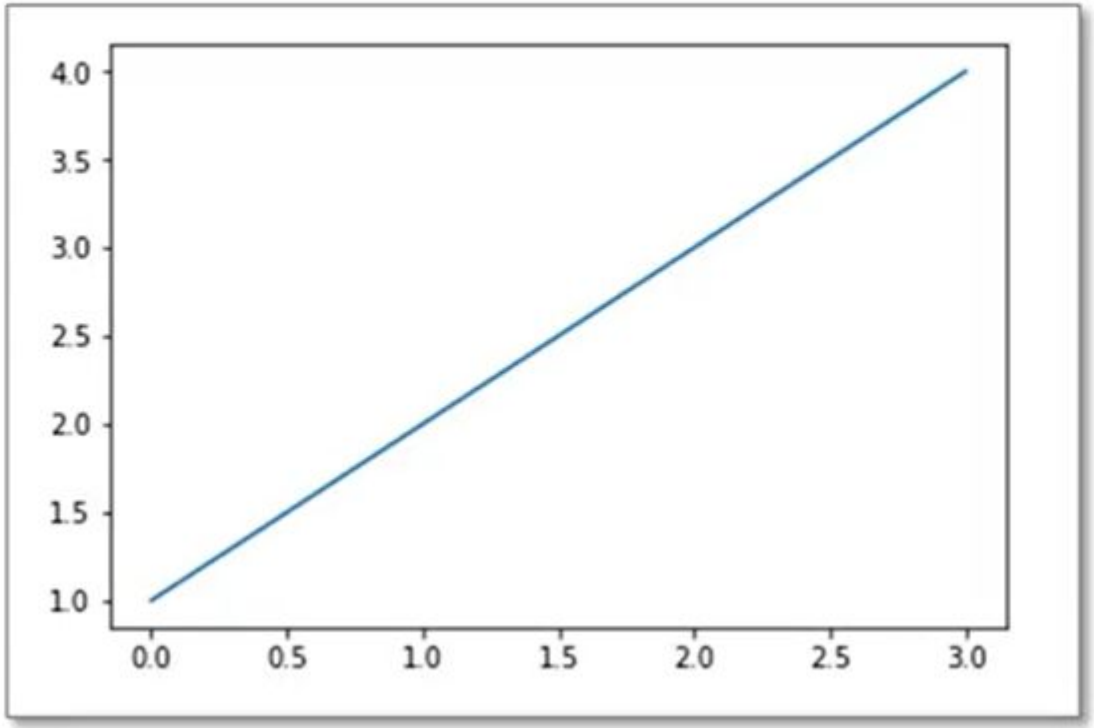
- Let us plot a simple graph on matplotlib

Code

Plot

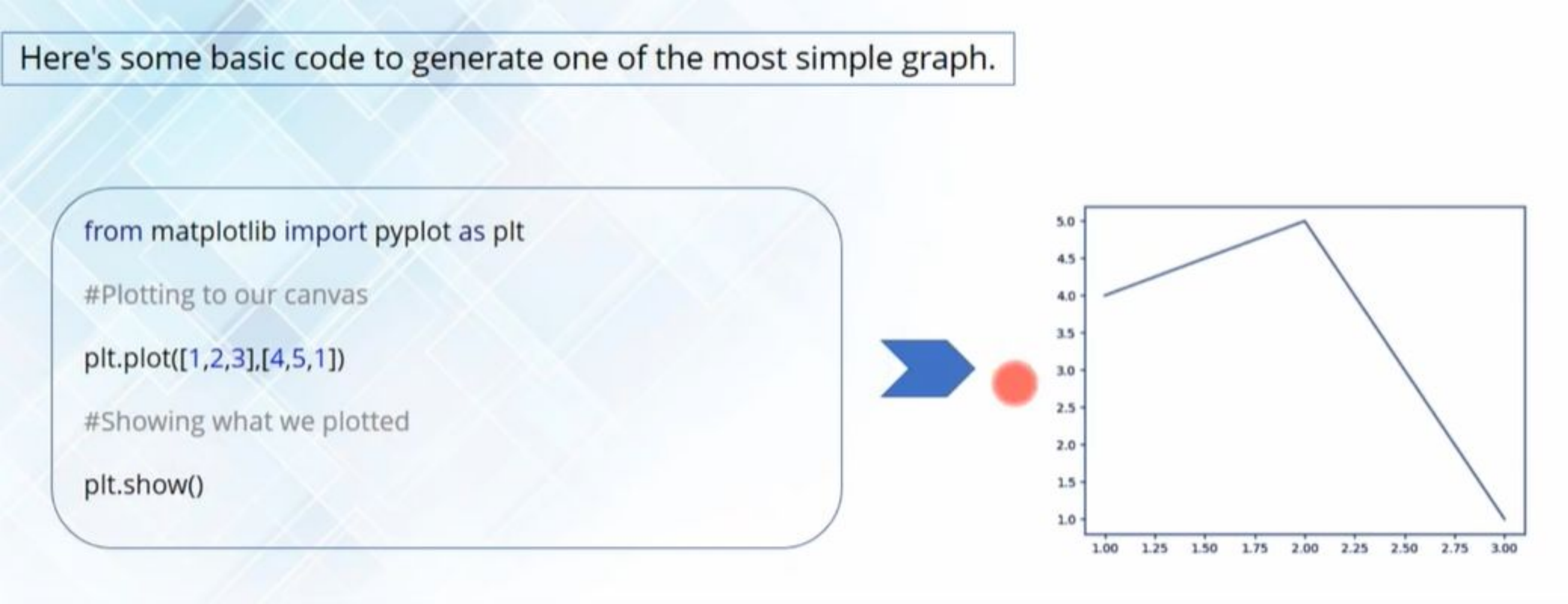


```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.show()
```



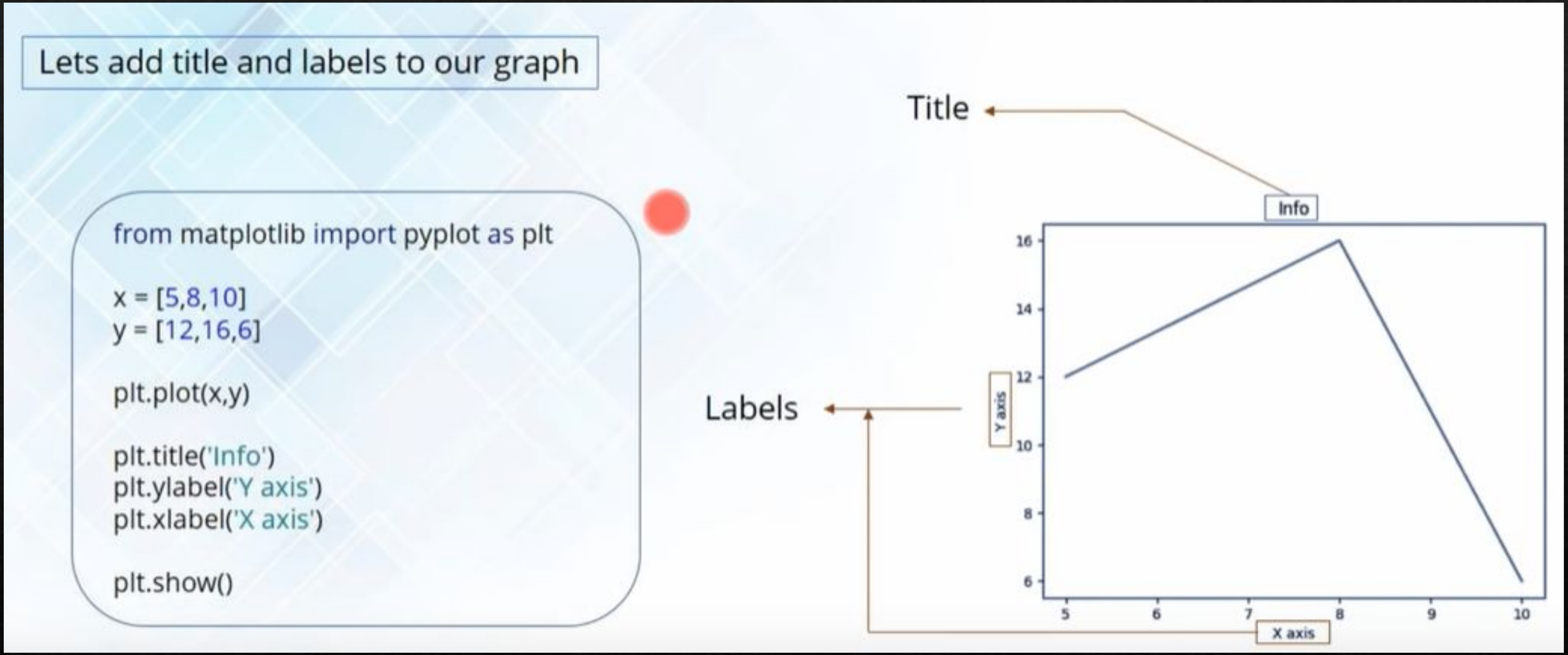
Types of Plots

Bar graph
Histograms
Scatter Plot
Pie Plot
Hexagonal Bin Plot
Area Plot



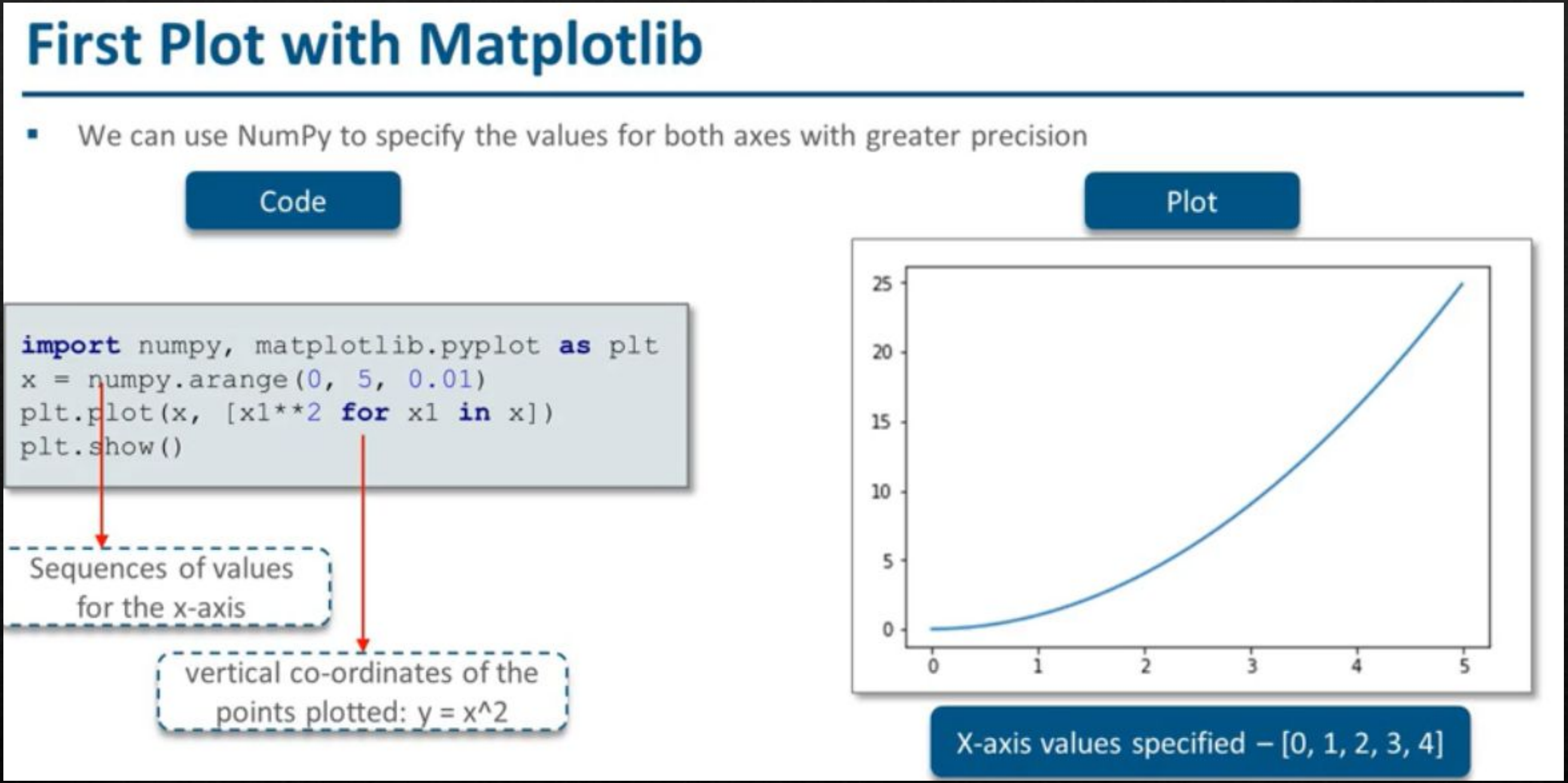
Here's some basic code to generate one of the most simple graph.

```
from matplotlib import pyplot as plt

#Plotting to our canvas

plt.plot([1,2,3],[4,5,1])

#Showing what we plotted

plt.show()
```



Lets add title and labels to our graph

```
from matplotlib import pyplot as plt

x = [5,8,10]
y = [12,16,6]

plt.plot(x,y)

plt.title('Info')
plt.ylabel('Y axis')
plt.xlabel('X axis')

plt.show()
```



First Plot with Matplotlib

- We can use NumPy to specify the values for both axes with greater precision

Code

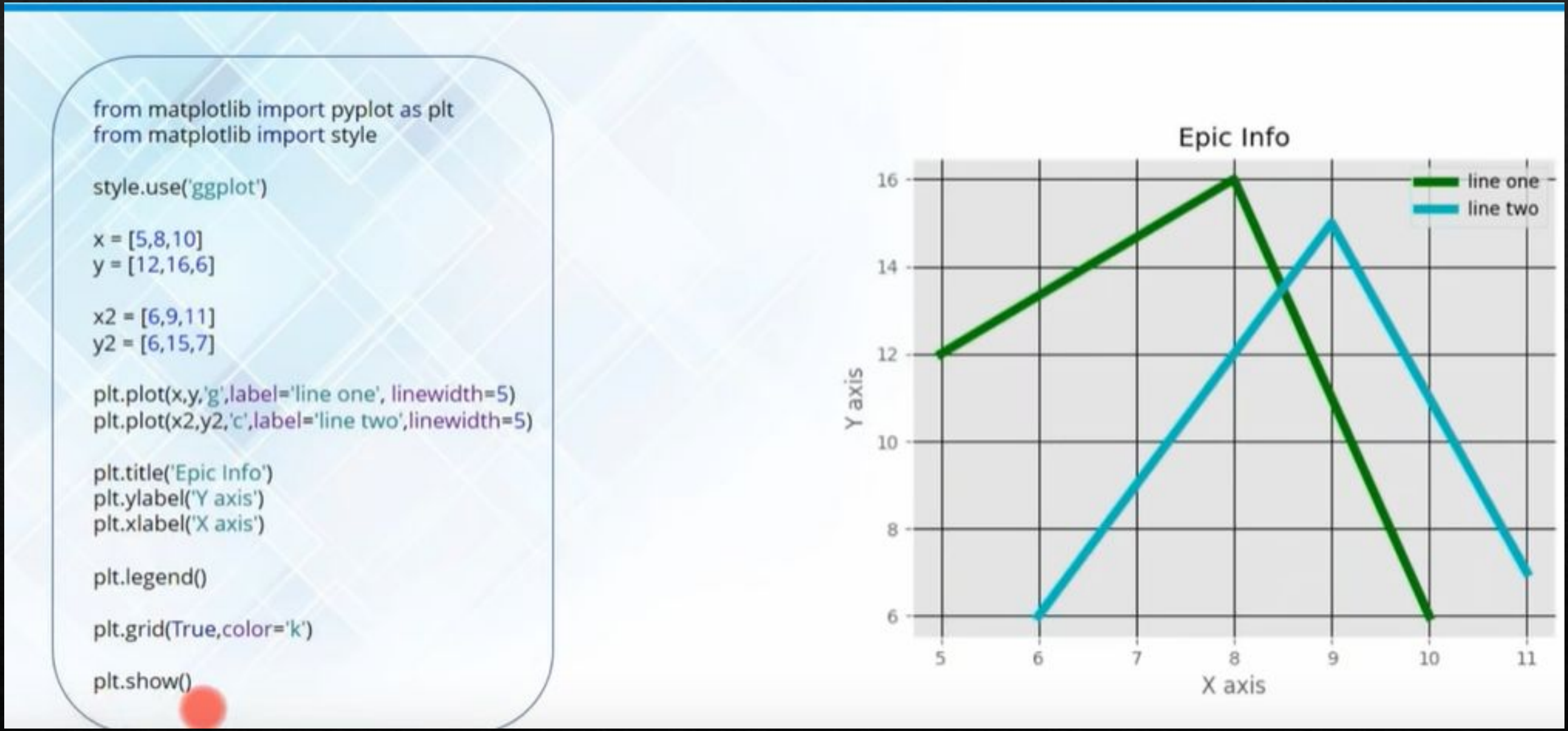
Plot

```
import numpy, matplotlib.pyplot as plt
x = numpy.arange(0, 5, 0.01)
plt.plot(x, [x1**2 for x1 in x])
plt.show()
```

Sequences of values for the x-axis

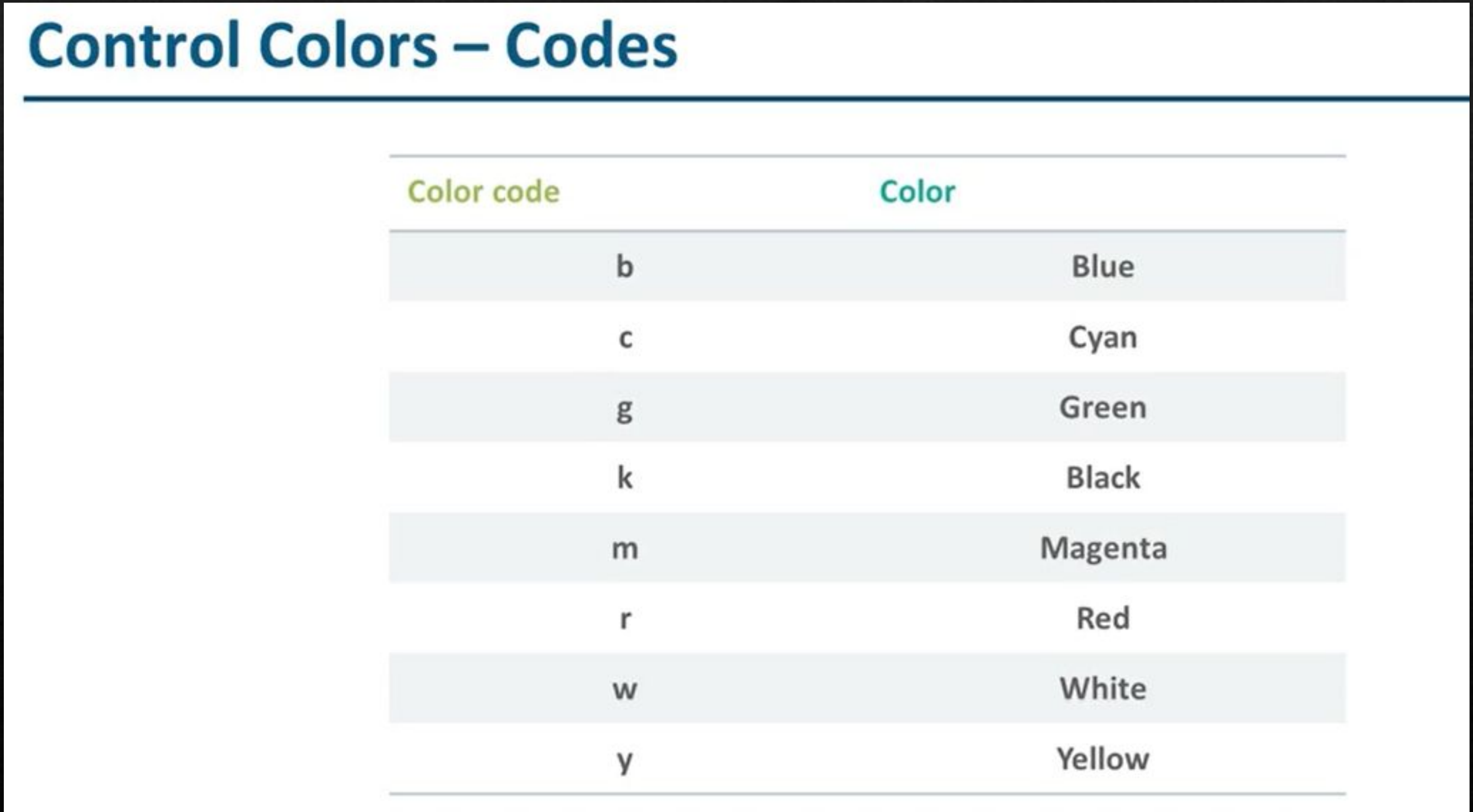
vertical co-ordinates of the points plotted: y = x^2

X-axis values specified – [0, 1, 2, 3, 4]



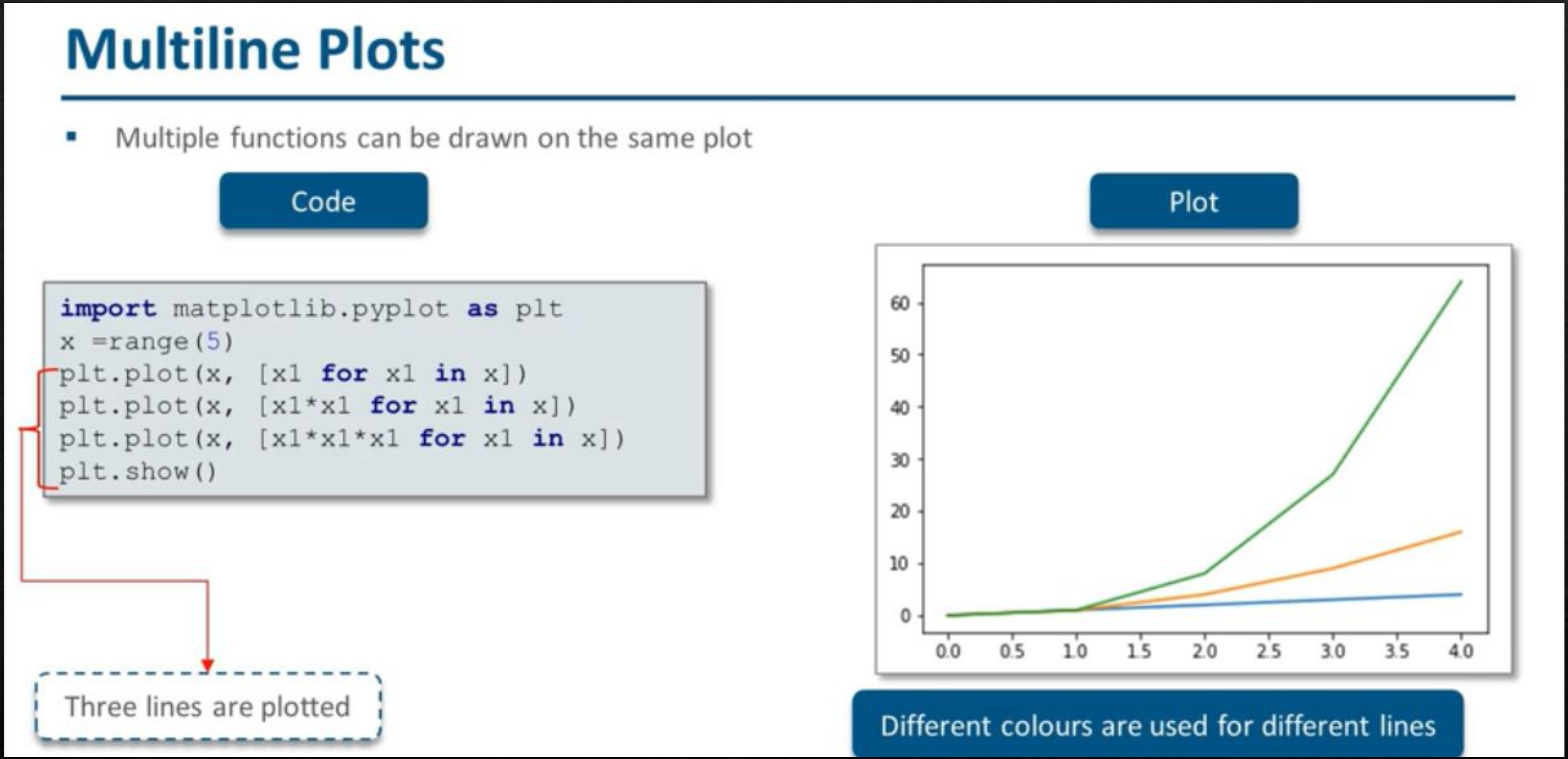
```
from matplotlib import pyplot as plt
from matplotlib import style

style.use('ggplot')

x = [5,8,10]
y = [12,16,6]

x2 = [6,9,11]
y2 = [6,15,7]

plt.plot(x,y,'g',label='line one', linewidth=5)
plt.plot(x2,y2,'c',label='line two',linewidth=5)

plt.title('Epic Info')
plt.ylabel('Y axis')
plt.xlabel('X axis')

plt.legend()

plt.grid(True,color='k')

plt.show()
```

https://matplotlib.org/stable/gallery/color/named_colors.html



Control Colors – Codes

Color code	Color
b	Blue
c	Cyan
g	Green
k	Black
m	Magenta
r	Red
w	White
y	Yellow



Multiline Plots

- Multiple functions can be drawn on the same plot

Code

```
import matplotlib.pyplot as plt
x =range(5)
plt.plot(x, [x1 for x1 in x])
plt.plot(x, [x1*x1 for x1 in x])
plt.plot(x, [x1*x1*x1 for x1 in x])
plt.show()
```

Three lines are plotted

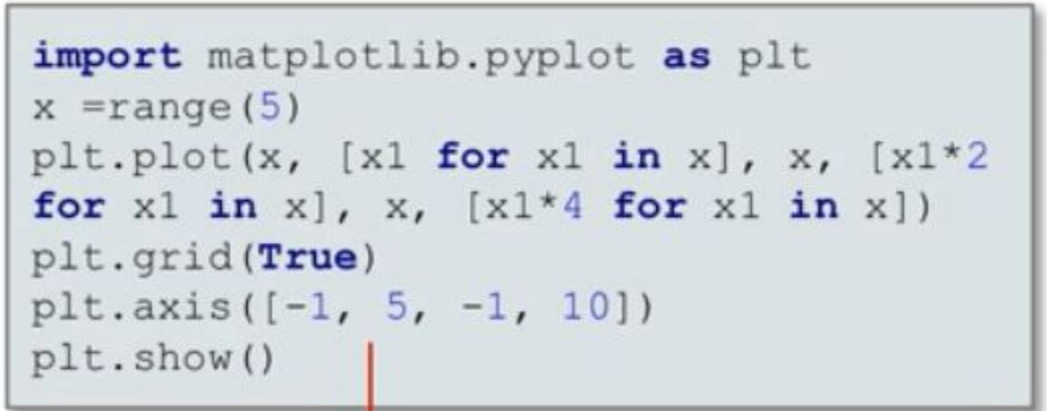
Plot

Different colours are used for different lines

Limiting the Axes

The scale of the plot can be set using axis()

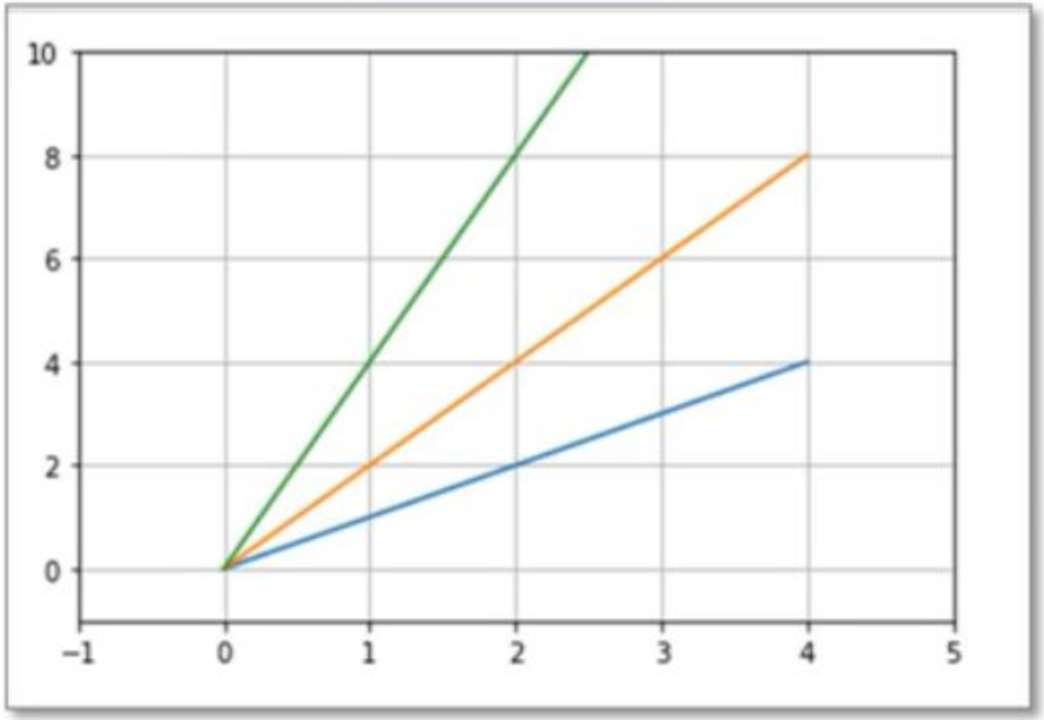
Code



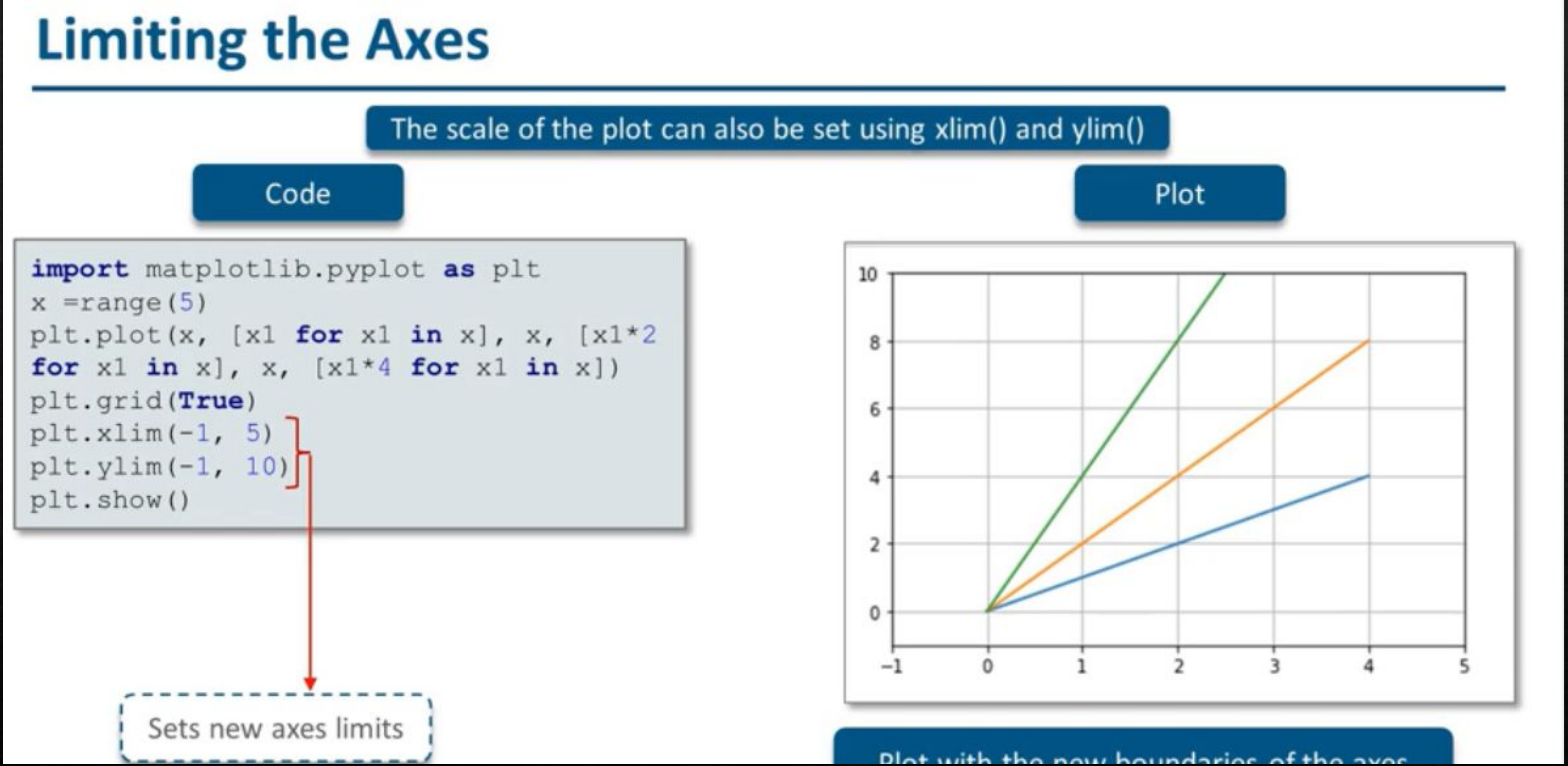
```
import matplotlib.pyplot as plt
x =range(5)
plt.plot(x, [x1 for x1 in x], x, [x1*2
for x1 in x], x, [x1*4 for x1 in x])
plt.grid(True)
plt.axis([-1, 5, -1, 10])
plt.show()
```

Sets new axes limits

Plot



Plot with the new boundaries of the axes



Limiting the Axes

The scale of the plot can also be set using xlim() and ylim()

Code

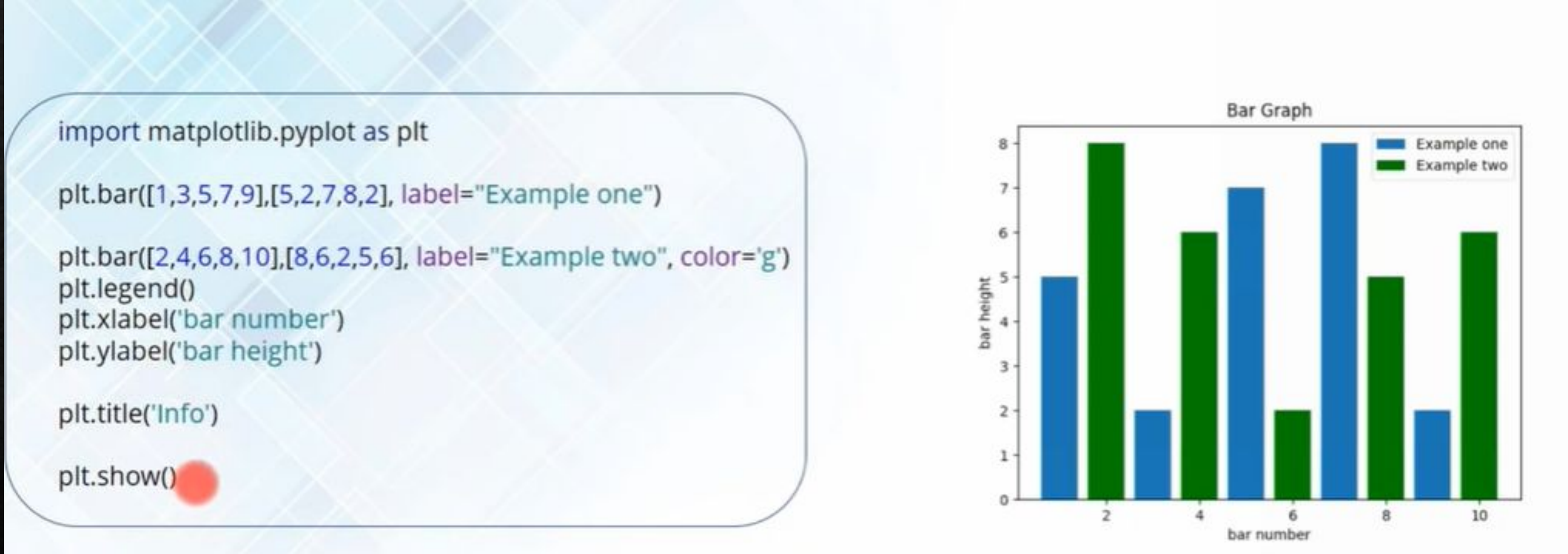
```
import matplotlib.pyplot as plt
x =range(5)
plt.plot(x, [x1 for x1 in x], x, [x1*2
for x1 in x], x, [x1*4 for x1 in x])
plt.grid(True)
plt.xlim(-1, 5)
plt.ylim(-1, 10)
plt.show()
```

Sets new axes limits

Plot

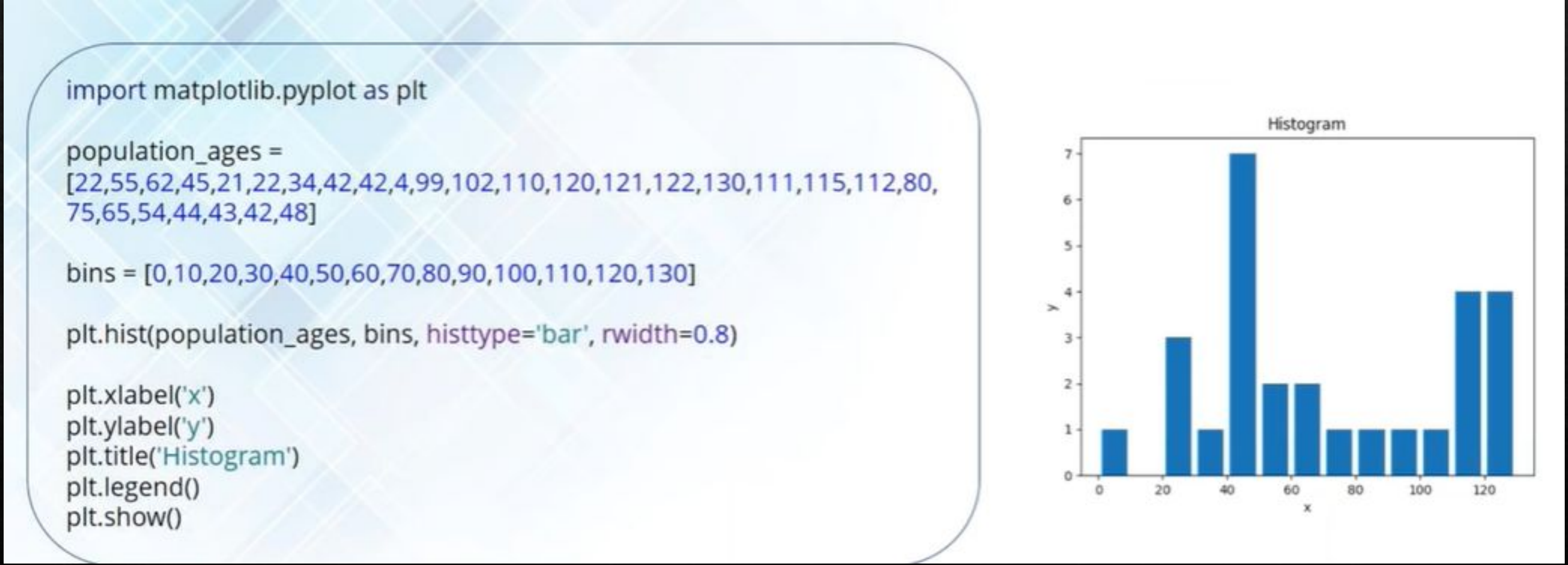
Plot with the new boundaries of the axes

Bar Graph



```
import matplotlib.pyplot as plt

plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")

plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two", color='g')
plt.legend()
plt.xlabel('bar number')
plt.ylabel('bar height')

plt.title('Info')

plt.show()
```

Histogram



```
import matplotlib.pyplot as plt

population_ages =
[22,55,62,45,21,22,34,42,42,4,99,102,110,120,121,122,130,111,115,112,80,
75,65,54,44,43,42,48]

bins = [0,10,20,30,40,50,60,70,80,90,100,110,120,130]

plt.hist(population_ages, bins, histtype='bar', rwidth=0.8)

plt.xlabel('x')
plt.ylabel('y')
plt.title('Histogram')
plt.legend()
plt.show()
```

Histogram

Histograms display the distribution of a variable over a range of frequencies or values

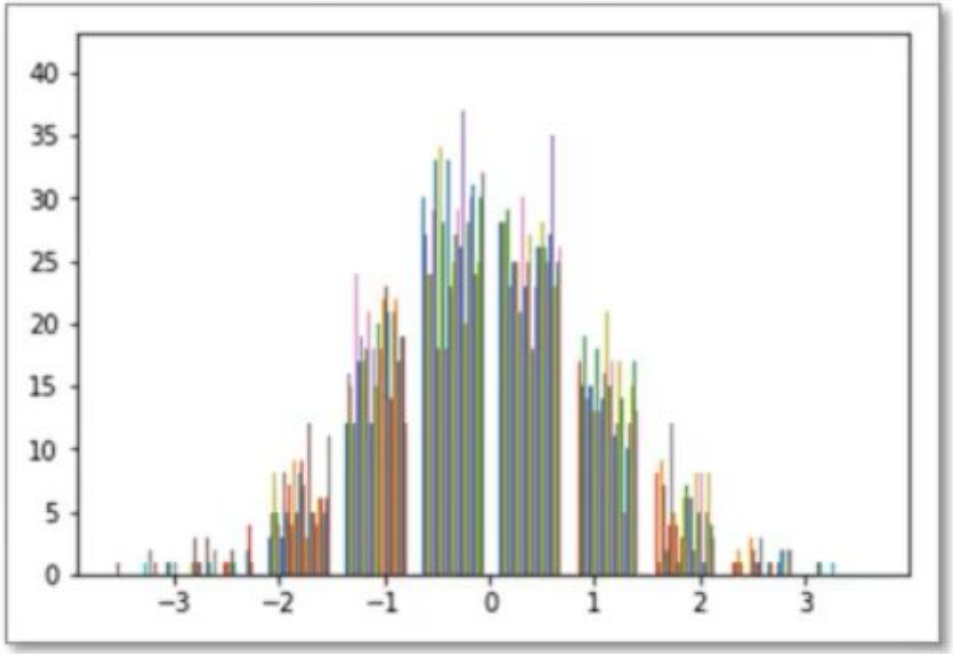
Code

Plot

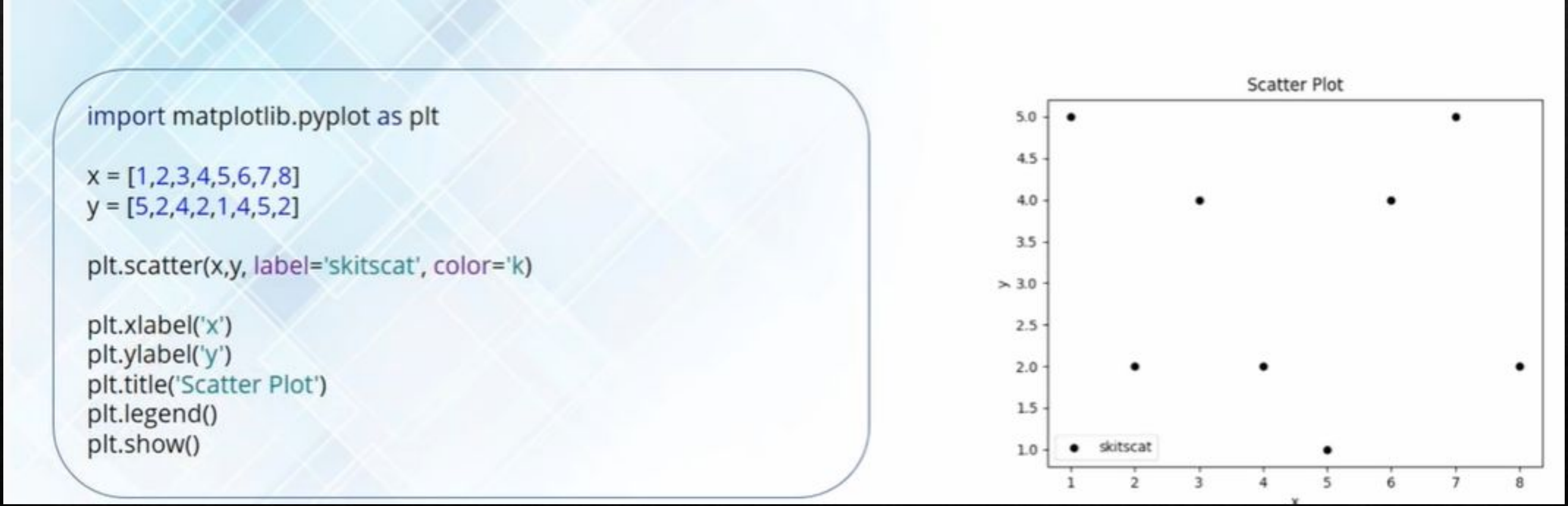
```
import matplotlib.pyplot as plt, numpy

y = numpy.random.randn(100, 100)
plt.hist(y)
plt.show()
```

Function to plot the histogram takes the dataset as the parameter

100x100 array of a Gaussian distribution

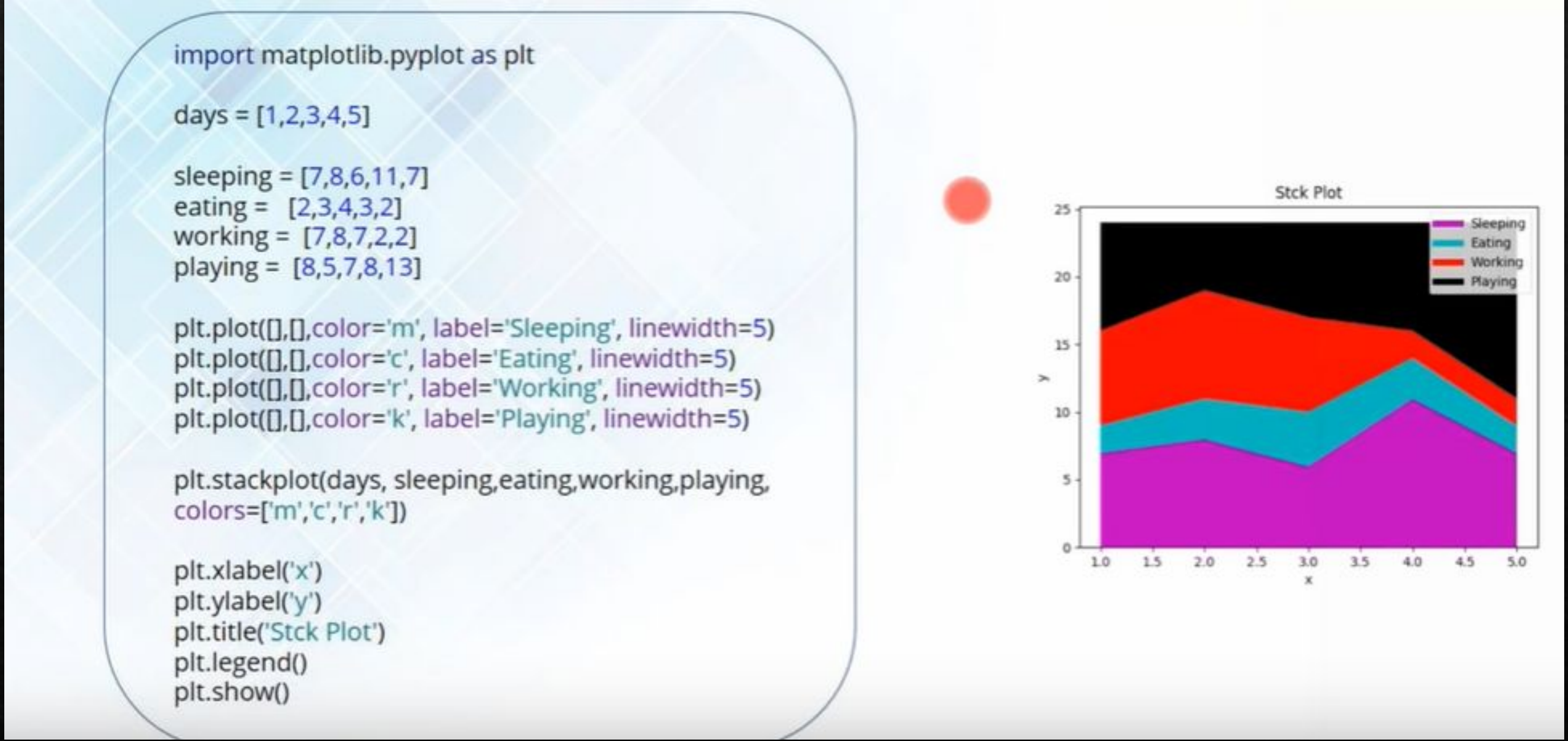


Scatter Plot



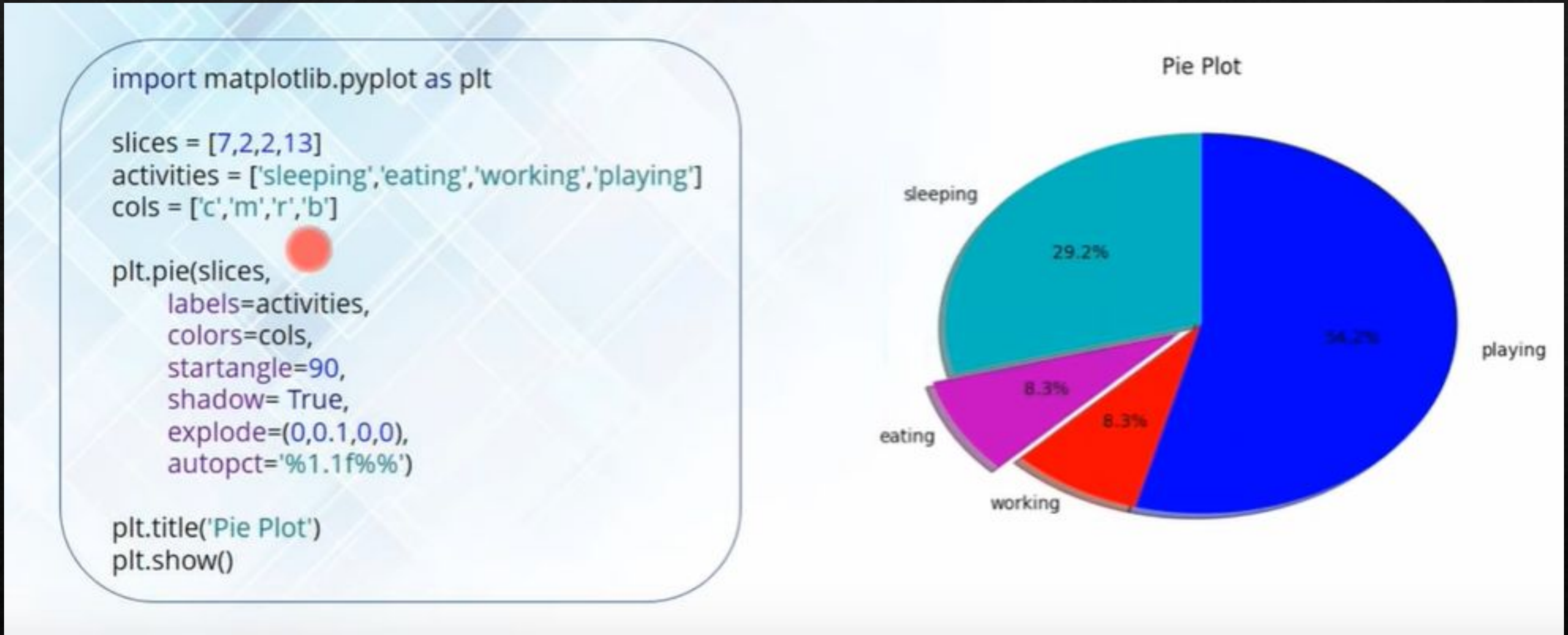
```
import matplotlib.pyplot as plt

x = [1,2,3,4,5,6,7,8]
y = [5,2,4,2,1,4,5,2]

plt.scatter(x,y, label='skitscat', color='k')

plt.xlabel('x')
plt.ylabel('y')
plt.title('Scatter Plot')
plt.legend()
plt.show()
```

Stack \Area Graph



```
import matplotlib.pyplot as plt

days = [1,2,3,4,5]

sleeping = [7,8,6,11,7]
eating =   [2,3,4,3,2]
working =  [7,8,7,2,2]
playing =  [8,5,7,8,13]

plt.plot([],[],color='m', label='Sleeping', linewidth=5)
plt.plot([],[],color='c', label='Eating', linewidth=5)
plt.plot([],[],color='r', label='Working', linewidth=5)
plt.plot([],[],color='k', label='Playing', linewidth=5)

plt.stackplot(days, sleeping,eating,working,playing,
colors=['m','c','r','k'])

plt.xlabel('x')
plt.ylabel('y')
plt.title('Stck Plot')
plt.legend()
plt.show()
```

Pie Chart



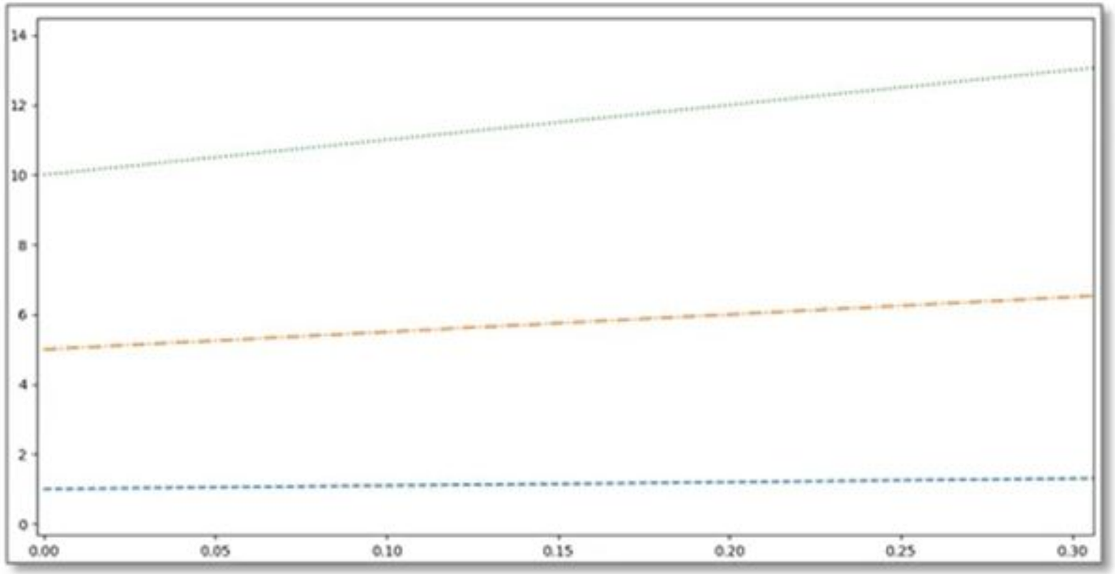
```
import matplotlib.pyplot as plt

slices = [7,2,2,13]
activities = ['sleeping','eating','working','playing']
cols = ['c','m','r','b']

plt.pie(slices,
        labels=activities,
        colors=cols,
        startangle=90,
        shadow= True,
        explode=(0,0.1,0,0),
        autopct='%1.1f%%')

plt.title('Pie Plot')
plt.show()
```

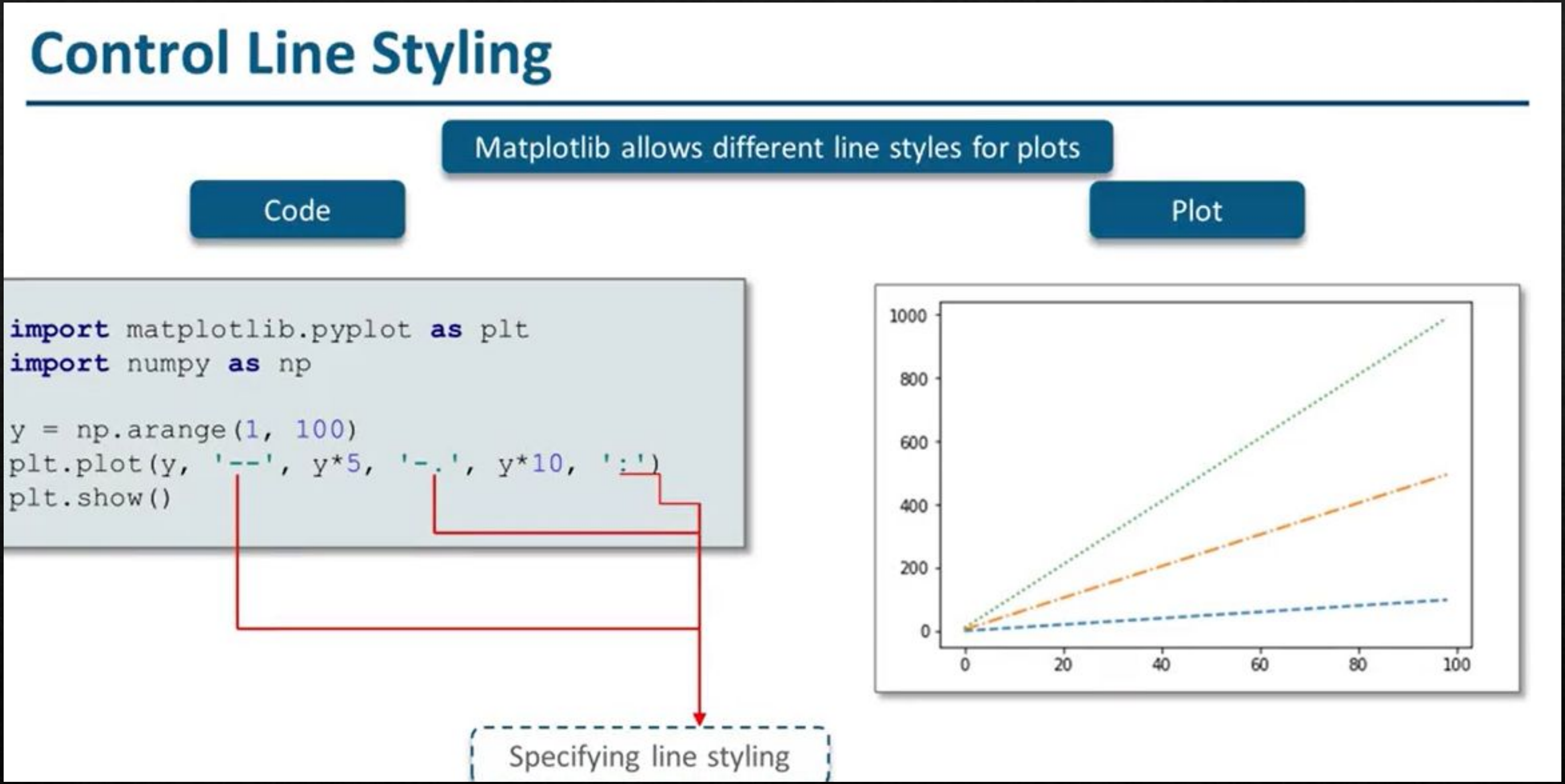
Control Line Styling

Matplotlib allows different line styles for plots

Plot

Style	Style Name
-	Solid line
--	Dashed line
-.	Dash-Dot line
:	Dotted Line





Control Line Styling

Matplotlib allows different line styles for plots

Code

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

y = np.arange(1, 100)
plt.plot(y, '--', y*5, '-.', y*10, ':')
plt.show()
```

Specifying line styling

Plot

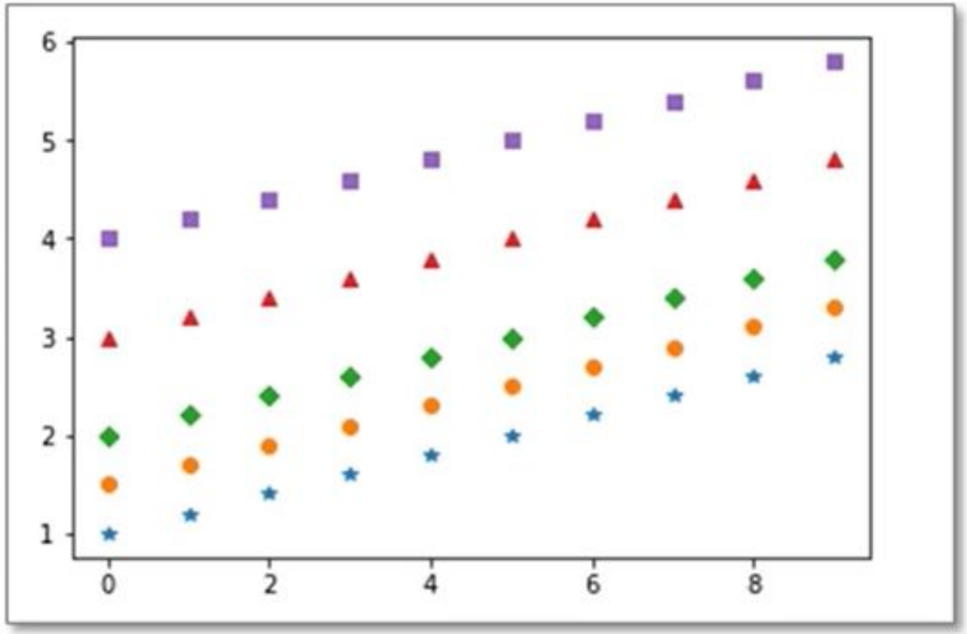
Control Marker Styling

Matplotlib provides customization options for markers

Code

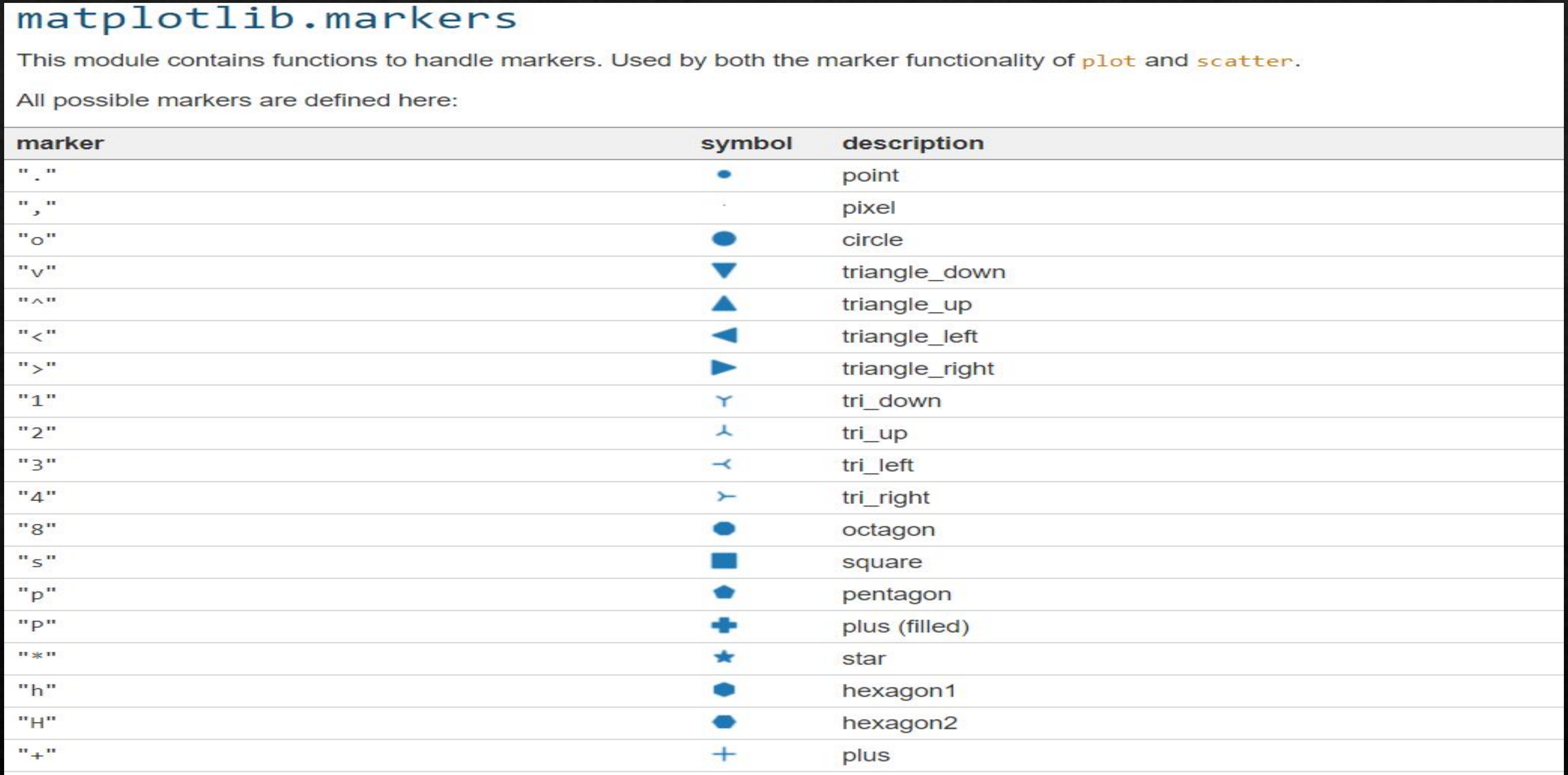
Plot

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

y = np.arange(1, 3, 0.2)
plt.plot(y, '*', y+0.5, 'o', y+1, 'D',
y+2, '^', y+3, 's')
plt.show()
```



Specifying line styling

https://matplotlib.org/stable/api/markers_api.html

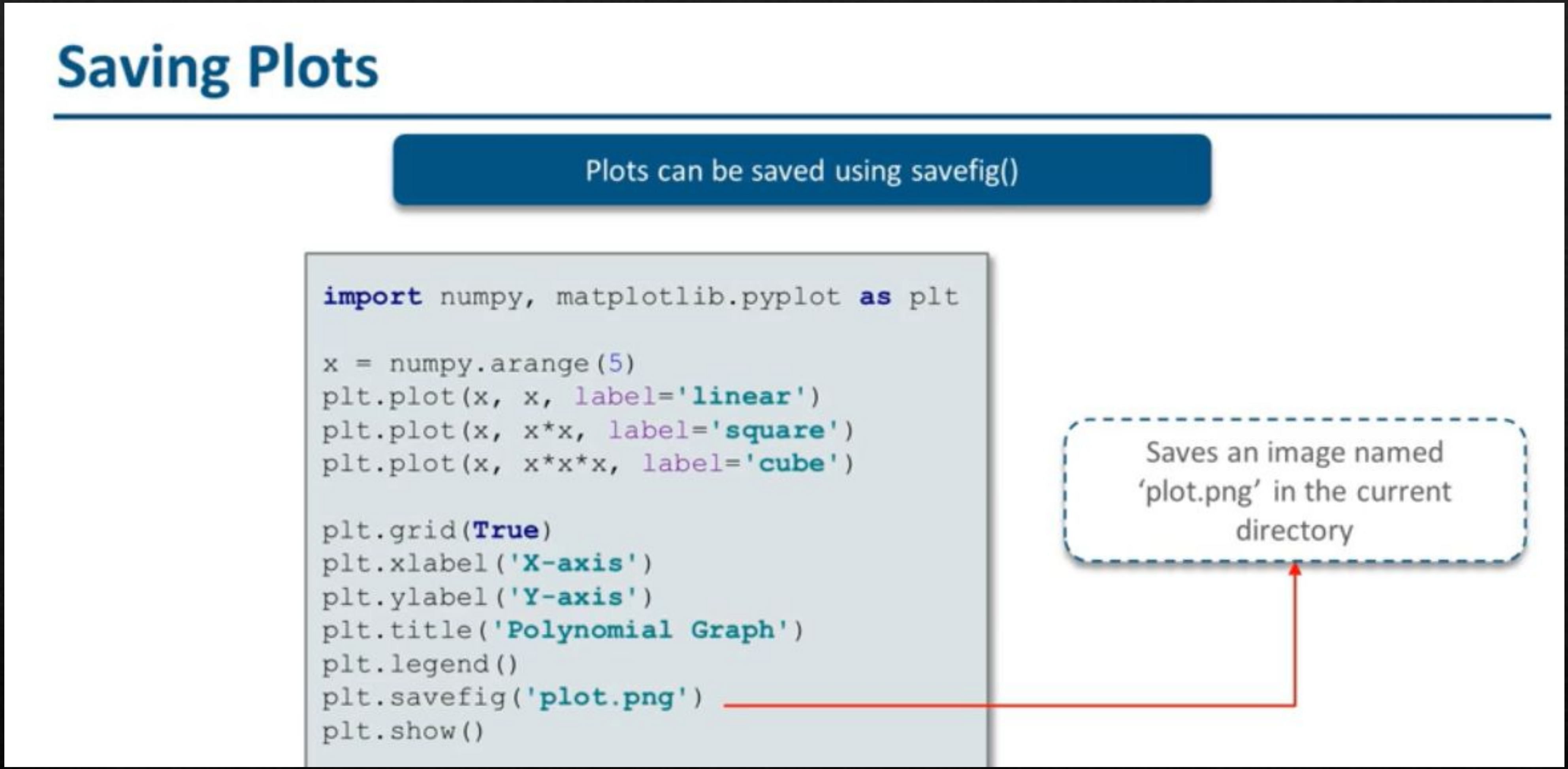


matplotlib.markers

This module contains functions to handle markers. Used by both the marker functionality of `plot` and `scatter`.

All possible markers are defined here:

marker	symbol	description
"."	•	point
","	·	pixel
"o"	●	circle
"v"	▼	triangle_down
"^"	▲	triangle_up
"<"	◀	triangle_left
">"	▶	triangle_right
"1"	Y	tri_down
"2"	⅄	tri_up
"3"	≺	tri_left
"4"	≻	tri_right
"8"	⯃	octagon
"s"	■	square
"p"	⬟	pentagon
"P"	✚	plus (filled)
"*"	★	star
"h"	⬢	hexagon1
"H"	⬣	hexagon2
"+"	+	plus



Saving Plots

Plots can be saved using savefig()

```
import numpy, matplotlib.pyplot as plt

x = numpy.arange(5)
plt.plot(x, x, label='linear')
plt.plot(x, x*x, label='square')
plt.plot(x, x*x*x, label='cube')

plt.grid(True)
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Polynomial Graph')
plt.legend()
plt.savefig('plot.png')
plt.show()
```

Saves an image named 'plot.png' in the current directory