UNIT 11
MATPLOTLIB DAN NUMPY

1 .1 Tujuan:

1.1.1 Mahasiswa dapat melakukan data visualisasi pada Python.
1.1.2 Mahasiswa dapat melakukan proses import library pada Python
1.1.3 Mahasiswa dapat mengoperasikan Numpy pada Python
1.1.2 Mahasiswa dapat membuat, memodifikasi, mengoperasikan dan menampilkan array di Jupyter Notebook

1 .2 Dasar Teori

Matplotlib adalah pustaka plot untuk bahasa pemrograman Python dan ekstensi matematika numeriknya NumPy . Ini menyediakan API berorientasi objek untuk menanamkan plot ke dalam aplikasi menggunakan toolkit GUI tujuan umum seperti Tkinter , wxPython , Qt , atau GTK + . Ada juga antarmuka "pylab" prosedural berdasarkan mesin negara (seperti OpenGL), yang dirancang sangat mirip dengan MATLAB , meskipun penggunaannya tidak disarankan. SciPy memanfaatkan Matplotlib.

Penggunaan Matplot umumnya digunakan dengan menggunakan bantuan Jupyter Notebook. *Jupyter Notebook merupakan web-based interactive computation enviorenment,* yang mana menggunakan web sebagai media penggunaannya.

Pada dasarnya untuk membuat sebuah grafik di Python sangatlah mudah. Hal ini karena, di Python sudah disediakan fasilitas untuk menampilkan grafik dari kumpulan data yang ada. Selain grafik yang ditampilkan, dapat juga ditambahkan aksesoris yang lain untuk memberikan informasi yang jelas.

Penggunaan perintah plot, bar atau yang lain bisa menampilkan sebuah grafik dalam bentuk 2D maupun 3D. Tentunya, masing-masing memiliki syarat-syarat yang harus terpenuhi. Misalnya, untuk menampilkan grafik 2D harus tersedia data matriks 2 dimensi. Jika akan menampilkan grafik 3D maka harus tersedia data matriks 3 dimensi pula.

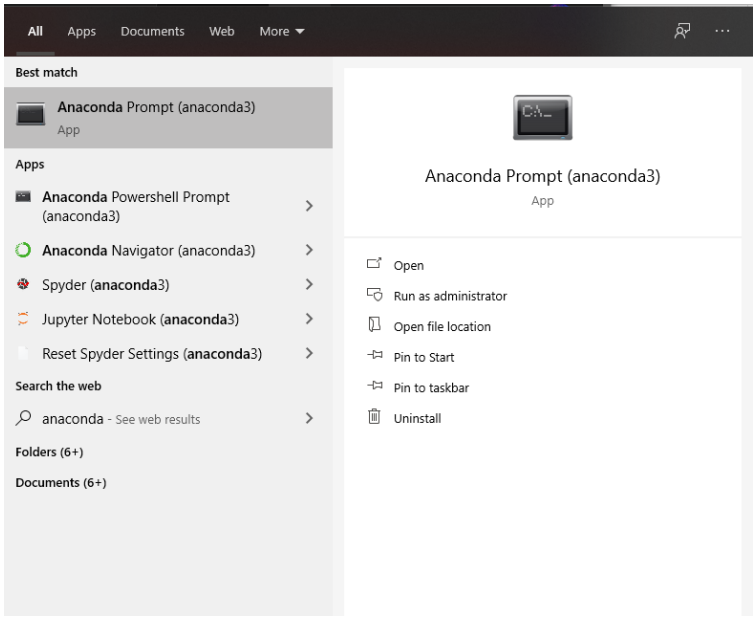
Membuat plot data terdapat beberapa bentuk dari fungsi plot, tergantung pada argumen inputnya. Jika y adalah sebuah vektor, maka sintak plot(y) akan menghasilkan sebuah grafik linier dengan sumbu x merupakan nilai indeks dari elemen y sedangkan sumbu y merupakan nilai y itu sendiri. Jika terdapat dua vektor sebagai argumennya, maka sintaks plot(x,y) akan menghasilkan grafik y versus x.

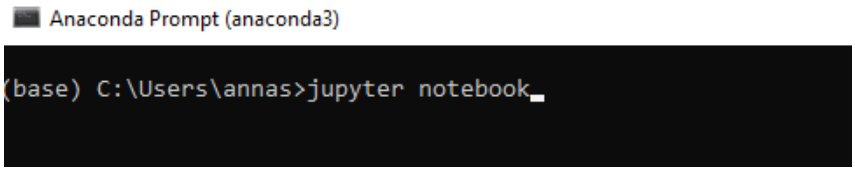
NumPy (Numerical Python) adalah library Python yang fokus pada scientific computing. NumPy memiliki kemampuan untuk membentuk

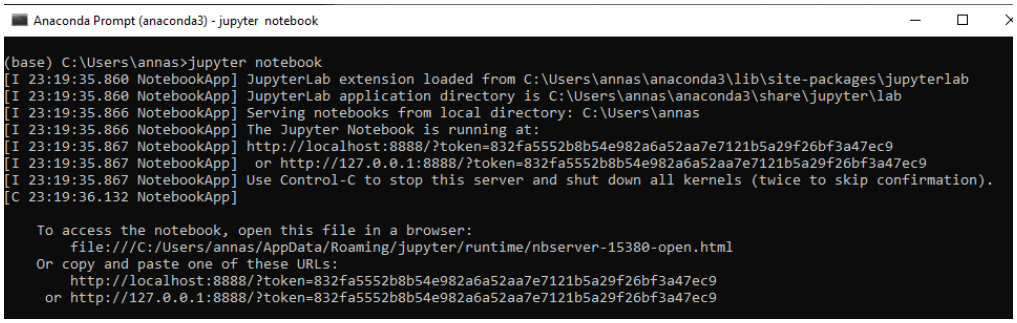
objek N-dimensional array, yang mirip dengan list pada Python. Keunggulan NumPy array dibandingkan dengan list pada Python adalah konsumsi memory yang lebih kecil serta runtime yang lebih cepat. NumPy juga memudahkan kita pada Aljabar Linear, terutama operasi pada Vector (1-d array) dan Matrix (2-d array).

1 .3 Membuka Jupyter Notebook

Untuk dapat menggunakan Matplotlib diperlukan bantuan alat bernama *Jupyter Notebook* dengan mengetik *"Jupyter Notebook"* pada terminal *"Anaconda Prompt"*. Kemudian mengakses alamat http://127.0.0.1:8888/tree atau http://localhost:8888/tree yang didapatkan dari terminal ke halaman browser.

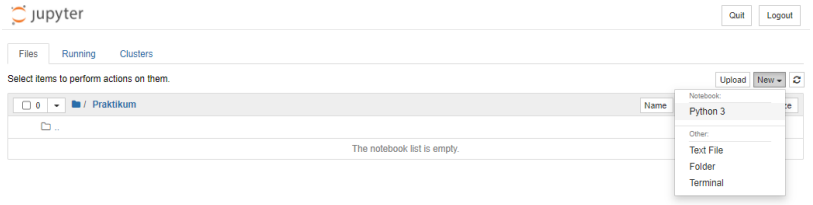




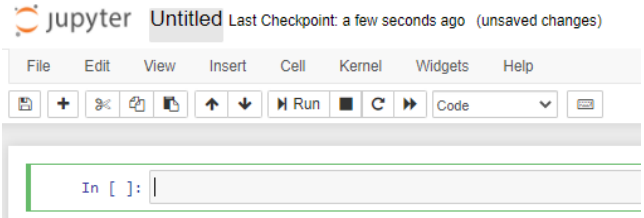


Pilih menu folder untuk membuat file yang akan masuk ke dalam lokasi file C:\users\(NamaPengguna)\

Pilih pilihan Python 3 untuk membuat file program di dalam folder yang telah dibuat setelahnya



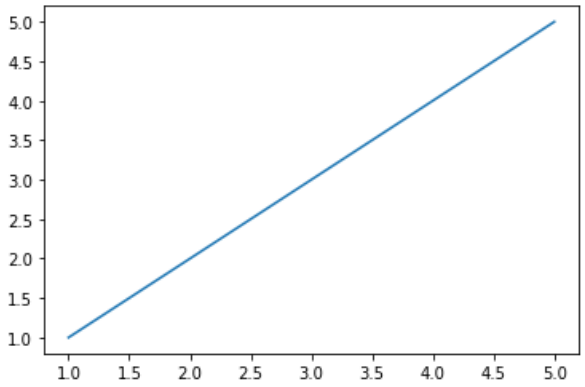
Jika sudah terbuat maka akan terlihat tampilan seperti dibawah ini, tekan untitled untuk mengubah nama program dari nama default menjadi nama yang praktikan inginkan.



1 .4 Membuat Plot Pada Jupyter

Memasukkan nilai kepada variabel dalam array , kemudian memberi nama kedua variabel, dan membentuk grafik linear dengan x melambangkan variabel pada x-axis dan variabel y melambangkan variabel pada y-axis, kemudian memberikan perintah fungsi plt.show() untuk menunjukkan keluaran dalam bentuk grafik.

```
In [5]: import matplotlib.pyplot as plt
        x=[1,2,3,4,5]
        y=[1,2,3,4,5]
        plt.plot(x,y)
        plt.show()
```

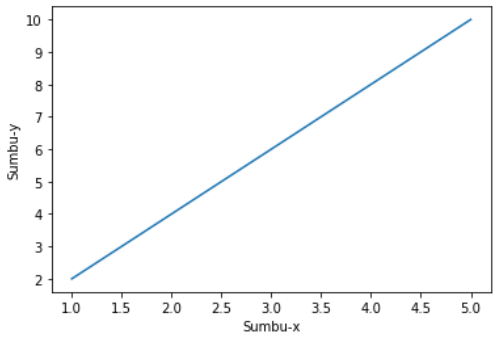


1 .5 Memberikan Label Pada Grafik

Memberikan label pada x-axis dan y-axis menggunakan “String” agar lebih mudah diidentifikasi.

```
Plt.xlabel(‘String’)
Plt.ylabel(‘String’)
```

```
In [7]: import matplotlib.pyplot as plt #masukin lib
        x=[1,2,3,4,5]
        y=[2,4,6,8,10]
        plt.xlabel('Sumbu-x')
        plt.ylabel('Sumbu-y')
        plt.plot(x,y)
        plt.show()
```

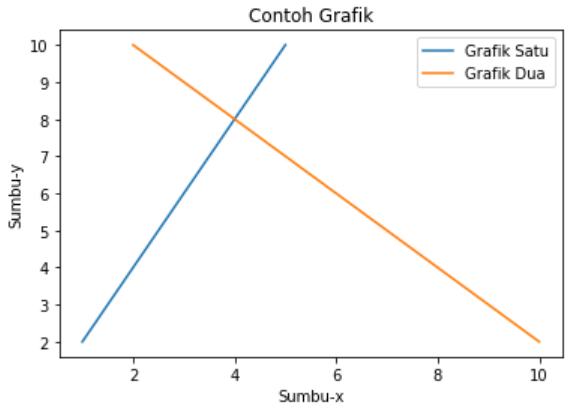


1 .6 Memberikan Judul dan Legenda Pada Grafik

Pemberian legenda untuk memudahkan proses identifikasi apabila terdapat dua garis lebih dalam satu grafik, pemberian judul dan legenda dapat menggunakan fungsi:

```
Plt.legend([“String1”, “String2”])
Plt.tittle(“Tittle”)
```

```
In [20]: import matplotlib.pyplot as plt #masukin lib
x1=[1,2,3,4,5]
y1=[2,4,6,8,10]
x2=[2,4,6,8,10]
y2=[10,8,6,4,2]
plt.xlabel('Sumbu-x')
plt.ylabel('Sumbu-y')
plt.plot(x1,y1)
plt.plot(x2,y2)
plt.legend(["Grafik Satu", "Grafik Dua"])
plt.title("Contoh Grafik")
plt.show()
```

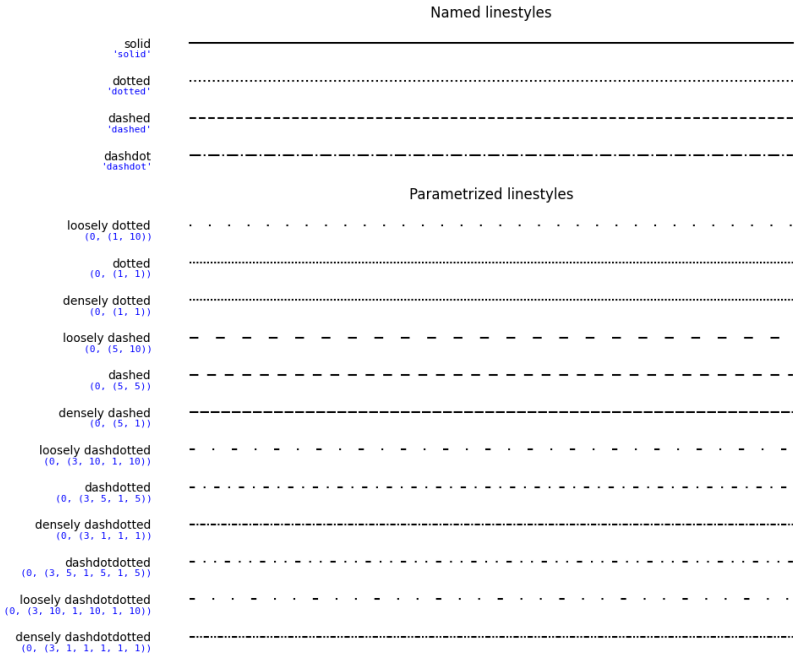


1 .7 Mengganti Jenis Garis Pada Grafik

Mengganti jenis garis grafik ke dalam bentuk lain dapat berupa titik putus-putus, grafik titik serta bentuk-bentuk lain, menggunakan fungsi `linestyle=’’` dengan detail di dalam kutip yang diinginkan.

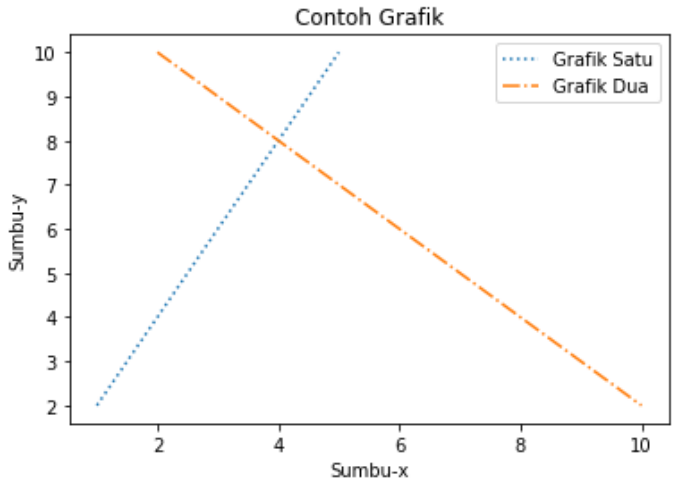
Contoh:

```
Plt.plot(x,y, linestyle=’solid’)
```



marker	symbol	description
"."		point
","		pixel
"o"		circle
"v"		triangle_down
"^"		triangle_up
"<"		triangle_left
">"		triangle_right
"1"		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
"x"		x
"X"		x (filled)
"D"		diamond
"d"		thin_diamond
"|"		vline
"_"		hline

```
In [23]: import matplotlib.pyplot as plt #masukin lib
x1=[1,2,3,4,5]
y1=[2,4,6,8,10]
x2=[2,4,6,8,10]
y2=[10,8,6,4,2]
plt.xlabel('Sumbu-x')
plt.ylabel('Sumbu-y')
plt.plot(x1,y1, linestyle='dotted')
plt.plot(x2,y2, linestyle='dashdot')
plt.legend(["Grafik Satu", "Grafik Dua"])
plt.title("Contoh Grafik")
plt.show()
```

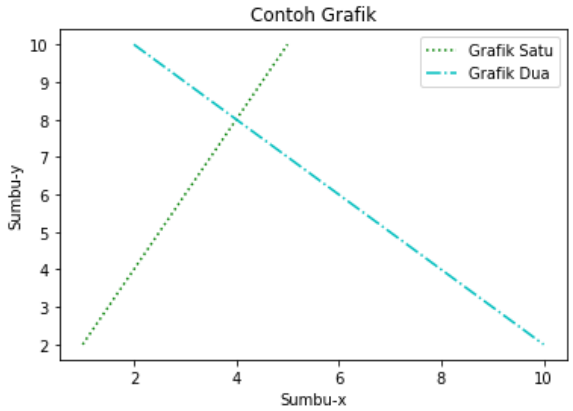


1 .8 Mengganti Warna Garis pada Grafik

Mengganti warna pada masing masing garis dengan memberikan detail khusus pada fungsi `plt.plot()` dengan menambahkan `color=''`, dengan variasi warna tersedia pada tabal dibawah:

b: Biru	k: Hitam
g: Hijau	w: Putih
r: Merah	y: Kuning
c: Sian	m: magenta

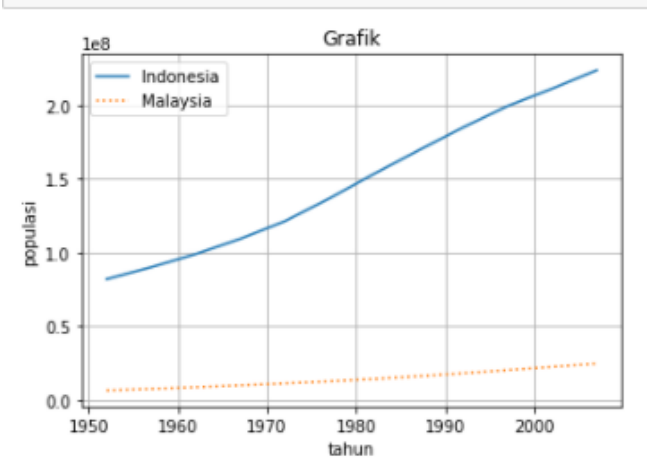
```
In [4]: import matplotlib.pyplot as plt #masukin lib
x1=[1,2,3,4,5]
y1=[2,4,6,8,10]
x2=[2,4,6,8,10]
y2=[10,8,6,4,2]
plt.xlabel('Sumbu-x')
plt.ylabel('Sumbu-y')
plt.plot(x1,y1, linestyle='dotted', color='g')
plt.plot(x2,y2, linestyle='dashdot', color='c')
plt.legend(["Grafik Satu", "Grafik Dua"])
plt.title("Contoh Grafik")
plt.show()
```



1 .9 Menambahkan Grid Pada Grafik

Menambahkan grid untuk mempermudah proses observasi terhadap grafik, dengan menambahkan fungsi `plt.grid()` sebelum fungsi `plt.show()`

```
In [18]: plt.plot(indo.year, indo.population)
         plt.plot(malay.year, malay.population, linestyle='dotted')
         plt.xlabel("tahun")
         plt.ylabel("populasi")
         plt.legend(["Indonesia", "Malaysia"])
         plt.title('Grafik')
         plt.grid()
         plt.show()
```



1 .10 Memberikan Batasan Interval pada Grafik

Memberikan batasan interval pada nilai didalam grafik, untuk memberikan jangkauan nilai yang diinginkan, untuk memberikan interval pada sumbu-x menggunakan

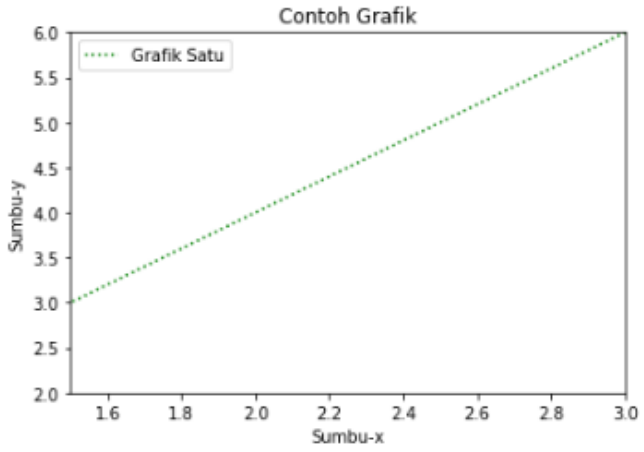
`Plt.xlim(min, max)`

Untuk memberikan interval pada sumbu-y menggunakan

`Plt.ylim(min, max)`

Dengan min menyatakan nilai interval minimum dan maks menyatakan nilai interval maksimum.

```
In [11]: import matplotlib.pyplot as plt #masukin lib
         import numpy as np

         x1=[1,2,3,4,5]
         y1=[2,4,6,8,10]
         plt.xlabel('Sumbu-x')
         plt.ylabel('Sumbu-y')
         plt.plot(x1,y1, linestyle='dotted', color='g')
         plt.xlim(1.5, 3.0) #membatasi interval dari 1.5<=x<=3.0
         plt.ylim(2.0, 6.0) #memnatasi interval ari 1<=y<=5
         plt.legend(["Grafik Satu", "Grafik Dua"])
         plt.title("Contoh Grafik")
         plt.savefig('Praktikum.png')
         plt.show()
```



1 .11 Menyimpan Grafik ke dalam Folder

Menyimpan gambar grafik yang dihasilkan pada matplotlib ke dalam satu folder yang sama dalam bentuk format png, menggunakan fungsi `plt.savefig('nama_file.png')`

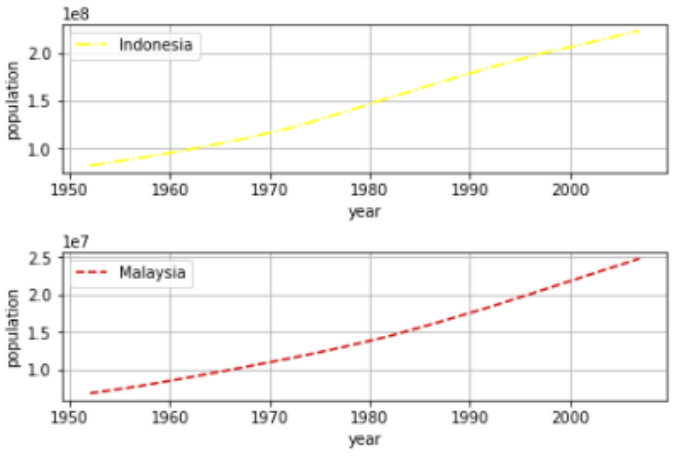
```
In [3]: import matplotlib.pyplot as plt #masukin lib
x1=[1,2,3,4,5]
y1=[2,4,6,8,10]
x2=[2,4,6,8,10]
y2=[10,8,6,4,2]
plt.xlabel('Sumbu-x')
plt.ylabel('Sumbu-y')
plt.plot(x1,y1, linestyle='dotted', color='g', markersize=99)
plt.plot(x2,y2, linestyle='dashdot', color='c')
plt.legend(["Grafik Satu", "Grafik Dua"])
plt.title("Contoh Grafik")
plt.savefig('Praktikum.png')
plt.show()
```

Praktikum.png a minute ago 16.2 kB

1 .12 Membuat Dua Grafik dalam Satu Keluaran

Membuat dua grafik dalam satu keluaran, dengan fungsi `plt.subplot(int, int)` dengan integer awal menyatakan kolom dan integer ke dua menyatakan kolom, pada contoh dibawah menggunakan data yang diperoleh dari data peningkatan populasi Indonesia dan Malaysia dari tahun 1950 hingga 2000.

```
In [18]:
#membuat 2 grafik dalam 1 output
fig, (ax1, ax2)=plt.subplots(2,1)
#(baris, kolom)
ax1.plot(indo.year, indo.population, color='#ffff00', linestyle='dashdot', label='Indonesia')
ax2.plot(malay.year, malay.population, color='#cc0000', linestyle='dashed', label='Malaysia')

ax1.set_xlabel('year')
ax1.set_ylabel('population')
ax1.legend()
ax1.grid()


ax2.set_xlabel('year')
ax2.set_ylabel('population')
ax2.legend()
ax2.grid()
plt.tight_layout()
plt.show()
```



1 .13 Memasukkan Library Numpy pada Jupyter

Memasukkan library Numpy ke dalam Jupyter Notebook dengan memasukkan "`import numpy as np`" kemudian mengeksekusikan library tersebut agar program dibawahnya dapat dijalankan:

```
In [2]: import numpy as np
```

1 .14 Membuat Array

Membuat array melalui fungsi `np.array([])` dapat membuat array 1D, array 2D maupun array 3D tergantung dari data yang kita masukkan ke dalam array.

```
In [7]: matriks1D=np.array([1,2,3]) #array 1 Dimensi
        print(matriks1D)

        [1 2 3]
```

```
In [6]: import numpy as np
        matriks2D=np.array([[1,1,1],[0,0,0]]) #3 kolom 2 baris
        print(matriks2D)

        [[1 1 1]
         [0 0 0]]
```

```
In [9]: matriks3D=np.array([[[1,1],[1,1],[1,1]],[[2,2],[2,2],[2,2]],[[3,3],[3,3],[3,3]]])
        #matriks 3 baris 2 kolom 3 kedalaman
        print(matriks3D)

        [[[1 1]
          [1 1]
          [1 1]]

         [[2 2]
          [2 2]
          [2 2]]

         [[3 3]
          [3 3]
          [3 3]]]
```

1 .15 Mendapatkan Dimensi dari Array

Mengetahui dimensi dari suatu array yang telah diberikan data di dalam suatu variabel `variabel.ndim`

```
In [11]: matriks1D.ndim
Out[11]: 1

In [12]: matriks2D.ndim
Out[12]: 2

In [13]: matriks3D.ndim
Out[13]: 3
```

1 .16 Mendapatkan Informasi Baris, Kolom serta Kedalaman suatu Array

Mengetahui informasi baris, kolom serta kedalaman suatu array melalui fungsi `variabel.shape`

```
In [14]: matriks1D.shape
Out[14]: (3,)

In [15]: matriks1D.shape
Out[15]: (3,)

In [16]: matriks2D.shape
Out[16]: (2, 3)

In [17]: matriks3D.shape
Out[17]: (3, 3, 2)
```

1 .17 Mengetahui serta Mengubah Tipe Data pada Array

Pada umumnya array berada dalam tipe data `int32` tapi kita dapat mengubah tipe data dari suatu array dengan menambahkan detail "`dtype=(tipe data)`" dengan tipe data bermacam-macam dari `int32, int64, int16` dan `float` di dalam fungsi `np.array([], dtype='').` Fungsi yang digunakan untuk melihat tipe data adalah `variabel.dtype.`

```
In [18]: matriks1D.dtype
Out[18]: dtype('int32')

In [20]: matriks1D=np.array([1,2,3], dtype='int8')
         print(matriks1D)
         [1 2 3]
```

1 .18 Mengetahui Banyak Bytes pada Suatu Array

Mengetahui *size* dari suatu array dengan memperhatikan tipe data array itu sendiri, 1 bytes = 8 bit, jika suatu array menggunakan tipe data `int32` berarti array tersebut memiliki size sebesar `4 bytes,` dengan menggunakan fungsi `variabel.itemsize.`

```
In [22]: matriks1D.itemsize
         #menggunakan tipe data int8
Out[22]: 1

In [23]: matriks2D.itemsize
         #menggunakan tipe data int32
Out[23]: 4
```

1 .19 Mengetahui Banyak Elemen pada Array

Mengetahui banyak elemen yang terdaoat pada suatu array dengan menggunakan fungsi khusus `variabel.size,` di dalam contoh matriks 2x3x3 ditemukan terdapat 18 elemen.

```
[[[1 1]
  [1 1]
  [1 1]]

 [[2 2]
  [2 2]
  [2 2]]

 [[3 3]
  [3 3]
  [3 3]]]

In [26]: matriks3D.size

Out[26]: 18
```

1 .20 Membuat Barisan dengan Numpy

Membuat suatu barisan dengan menggunakan Numpy dapat dilakukan dengan dua cara yakni dengan menggunakan `np.linspace` dan `np.arange:`

`np.linspace`	merupakan fungsi yang membagi suku bilangan berdasarkan nilai yang ditentukan.
`np.linspace(mulai, akhir, banyak suku bilangan)`	
`np.arange`	merupakan fungsi yang membagi suku bilangan berdasarkan beda antar suku bilangan yang ditentukan.
`np.arange(mulai, akhir, beda)`	

```
In [8]: import numpy as np
        x=np.linspace(0,10,5)
        print(x)
        y=np.arange(0,10,5)
        print(y)

        [ 0.   2.5  5.   7.5 10. ]
        [0 5]
```

1 .21 Mengetahui Elemen Spesifik dari Array

Mengetahui elemen spesifik dari array menggunakan fungsi yang tersedia dari Numpy, Apabila tersedia matriks seperti dibawah:

```
In [12]: matriks=np.array([[1,1,1],[1,2,3],[0,1,0]])
         matriks

Out[12]: array([[1, 1, 1],
                [1, 2, 3],
                [0, 1, 0]])
```

Maka dapat dicari nilai eleme spesifik secara langsung tanpa harus melihat matriks awal secara manual dengan menggunakan fungsi dibawah dan memerhatikan bahwa Indeks fungsi selalu diawali dari nilai nol (0)

`Variabel[baris,kolom]`

```
In [21]: print(matriks[0,0])
         print(matriks[2,2])
         print(matriks[1,2])

         1
         0
         3
```

Untuk mengetahui isi dari seluruh kolom atau seluruh baris dapat menggunakan fungsi yang sama dengan menambahkan simbol (:) untuk baris/kolom yang ingin di cari.

```
In [27]: print(matriks[:,0])
         print('\n')
         print(matriks[0,:])

         [1 1 0]


         [1 1 1]
```

1 .22 Membuat Matriks Khusus

1.22.1.1. Membuat Matriks dengan seluruh elemen bernilaikan nol (0)

Membuat matriks dengan elemen berisikan angka nol (0) dengan format fungsi:

`np.zeros((baris, kolom))`

```
In [20]: import numpy as np
         np.zeros((3,3))

Out[20]: array([[0., 0., 0.],
                [0., 0., 0.],
                [0., 0., 0.]])
```

1.22.1.2. Membuat Matriks dengan seluruh elemen bernilai satu (1)

Membuat matriks dengan elemen berisikan angka satu (1) dengan format fungsi:

```
np.ones((baris, kolom))
```

```
In [22]: import numpy as np
         np.ones((2,3))

Out[22]: array([[1., 1., 1.],
                [1., 1., 1.]])
```

1.22.1.3. Membuat Matriks dengan seluruh elemen bernilaikan suatu nilai yang dimasukkan

Membuat matriks dengan elemen berisikan angka yang diinginkan dengan format fungsi:

```
np.full((baris, kolom), angka )
```

```
In [27]: import numpy as np
         np.full((2,3), 20)

Out[27]: array([[20, 20, 20],
                [20, 20, 20]])
```

1.22.1.4. Membuat Matriks Identitas

Membuat matriks dengan elemen berisikan elemen satu (1) dengan format fungsi:

```
np.full(dimensi)
```

```
In [29]: import numpy as np
         np.identity(3)

Out[29]: array([[1., 0., 0.],
                [0., 1., 0.],
                [0., 0., 1.]])
```

1 .23 Operasi Matriks pada Numpy

Menggunakan fungsi dalam operasi perhitungan matriks dengan keempat fungsi dasar nemiliki format inti yang sama.

In [51]:
```
import numpy as np
a=np.array([[1,2],[1,2]])
b=np.array([[2,2],[2,2]])
print("Pengurangan \n" +str(np.subtract(a,b)))
print("Pertambahan \n" +str(np.add(a,b)))
print("Pembagian \n" +str(np.divide(a,b)))
print("Perkalian \n" +str(np.multiply(a,b)))
```

```
Pengurangan
[[-1  0]
 [-1  0]]
Pertambahan
[[3 4]
 [3 4]]
Pembagian
[[0.5 1. ]
 [0.5 1. ]]
Perkalian
[[2 4]
 [2 4]]
```

1 .24 Mengakarkan Seluruh Elemen pada Matriks

Mengakarkan seluruh elemen pada suatu matriks dengan mendefinisikan matriks ke dalam bentuk array dan menjadikan array tersebut ke dalam bentuk variabel.

In [54]:
```
import numpy as np
a=np.array([[1,2],[1,2]])
print("Akar \n" +str(np.sqrt(a)))
```

```
Akar
[[1.         1.41421356]
 [1.         1.41421356]]
```

1 .25 Menjumlahkan Seluruh Elemen di dalam Matriks

Menjumlahkan seluruh elemen pada setiap baris dan setiap kolom matriks.

In [56]:
```
import numpy as np
a=np.array([[1,2],[1,2]])
print("Jumlah Seluruh: \n" +str(np.sum(a)))
```

```
Jumlah Seluruh:
6
```

Untuk menjumlahkan seluruh elemen secara vertikal dan horizontal dapat menggunakan fungsi:

```
Axis=0 (Untuk vertikal)
Axis=1 (Untuk Horizontal)
```

```
In [61]: import numpy as np
         a=np.array([[1,2],[2,2]])
         print(a)
         #Menjumlahkan secara vertikal
         print("Jumlah masing-masing baris : \n" +str(np.sum(a, axis=0)))
         #Menjumlahkan secara horizontal
         print("Jumlah masing-masing kolom: \n" +str(np.sum(a, axis=1)))
```

```
[[1 2]
 [2 2]]
Jumlah masing-masing baris :
[3 4]
Jumlah masing-masing kolom:
[3 4]
```

1 .26 Transpose Matriks dengan Numpy

Menggunakan numpy dalam men-transpose suatu matriks yang sudah di definisikan di dalam variabel dalam bentuk array dalam format:

`Variabel.transpose()`

```
In [5]: import numpy as np
        a=np.array([[1,2],[2,2]])
        a.transpose()
```

```
Out[5]: array([[1, 2],
               [2, 2]])
```

1 .27 Perkalian Silang dan Perkalian dot Vektor

Dengan menggunakan Numpy dapat ditemukan hasil dari perkalian vektor antara dua array atau lebih dengan format d-berikut:

```
np.dot(a,b)
np.cross(a,b)
```

```
In [9]: import numpy as np
        a=np.array([[1,2],[1,2]])
        b=np.array([[2,2],[2,2]])
        print('Perkalian dot \n' +str(np.dot(a,b)))
        print('\nPerkalian cross\n' +str(np.cross(a,b)))

        Perkalian dot
        [[6 6]
         [6 6]]

        Perkalian cross
        [-2 -2]
```

1 .28 Fungsi Trigonometrik pada Numpy

Numpy pada dasarnya digunakan untuk perhitungan matematis dengan berbagai fungsi didalamnya, salah satu fungsi transenden yang dapat digunakan

np.sin(x)	Membuat fungsi sinus
np.cos(x)	Membuat fungsi cosinus
np.tan(x)	Membuat fungsi tangen
np.arcsin(x)	Membuat fungsi invers dari sin untuk mengetahui nilai sudut asal
np.arccos(x)	Membuat fungsi invers dari cos untuk mengetahui nilai sudut asal
np.arctan(x)	Membuat fungsi invers dari tan untuk mengetahui nilai sudut asal
np.deg2rad	Mengubah bentuk derajat ke dalam bentuk radian
np.rad2deg	Mengubah bentuk radian ke dalam bentuk derajat

Dengan nilai x merupakan nilai dari variabel yang ingin dimetakan ke dalam fungsi awal yang dapat dilakukan dengan fungsi

`np.linspace() dan np.arange()`

1 .29 Fungsi Logaritmik pada Numpy

Numpy juga menyediakan perintah fungsi untuk fungsi logaritmik

np.log(x)	Membuat fungsi logaritma natural
np.log10(x)	Membuat fungsi logaritma dengan basis 10
np.log2(x)	Membuat fungsi logaritma dengan basis 2

Untuk fungsi dengan basis selain euler, 10 dan 2 dapat dicari dengan menggunakan manipulasi matematis melalui fungsi

$^{c}\log b = {}^{a}\log b \,/\, {}^{a}\log c$

```
In [46]: import numpy as np
         a=np.log(27)/np.log(3)
         print(a)

         3.0
```

1 .30 Fungsi Kompleks pada Numpy

Menggunakan fungsi kompleks pada numpy dalam bentuk kompleks untuk mengetahui fungsi dalam polar.

`np.angle(f(z))`

Mengetahui nilai derajat dari fungsi kompleks untuk dimasukkan dalam fungsi polar yang menghasilkan nilai dalam bentuk radian.

```
In [51]: np.angle(1+1j)
Out[51]: 0.7853981633974483
```

Untuk mengubah nilai radian ke dalam nilai derajat dapat menambahkan argumen `deg=True` di dalam tanda kurung fungsi.

```
In [52]: np.angle(1+1j, deg=True)
Out[52]: 45.0
```

Mengetahui nilai bilangan *real* dan *Imajiner* dari fungsi kompleks menggunakan fungsi

`np.real(variabel)`

dan

`np.imag(variabel)`

```
In [57]: kompleks=np.array([1+1j, 5+8j, -2+4j])
         print('Nilai Real adalah: \n'+str(np.real(kompleks)))
         print('\nNilai Imajiner adalah: \n'+str(np.imag(kompleks)))

         Nilai Real adalah:
         [ 1.  5. -2.]

         Nilai Imajiner adalah:
         [1. 8. 4.]
```

Mencari nilai kompleks konjugasi dari suatu fungsi kompleks

`np.conjugate(kompleks),`

dinyatakan dalam bentuk array untuk mencari nilai di dalam variabel[0]

```
In [61]: kompleks=np.array([1+1j, 5+8j, -2+4j])
         a=np.conjugate(kompleks[0])
         print(a)

         (1-1j)
```

1 .31 Derivasi menggunakan Numpy

Numpy dapat digunakan untuk menentukan besar turunan dari suatu fungsi yang diberikan, fungsi tersebut di inisiasikan sebagai array terlebih dahulu untuk dapat di derviasikan:

```
In [10]: #turunan
         import numpy as np
         y=np.poly1d([1,1,2,5])
         print("Fungsi polinomial\n" +str(y))

         turunan=y.deriv()
         print('Turunan adalah: \n'+str(turunan))
         #Turunan pada x=2
         print(turunan(2))
         #menurunkan dua kali
         turunan_ke2=y.deriv(2)
         print('Turunan ke dua adalah' + str(turunan_ke2))

         Fungsi polinomial
            3     2
         1 x + 1 x + 2 x + 5
         Turunan adalah:
            2
         3 x + 2 x + 2
         18
         Turunan ke dua adalah
         6 x + 2
```

Fungsi di inisiasikan dalam bentuk polinomial menggunakan fungsi

`np.poly1d([5,3,1])`

Yang mana angka di dalam array merepresentasikan nilai konstanta dari suatu fungsi polinomial, dengan bagian kiri merepresentasikan nilai konstanta dengan nilai x^0 , kemudian yang tengah merepresentasikan nilai konstanta dari nilai x dan bagian kiri merupakan konstanta dengan nilai x^2, yang menghasilkan fungsi:

`Polinomial = 5X`^{2}`+3X+1`

Kemudian apabila fungsi polinomial telah dideklarasikan di dalam suatu variabel, maka fungsi derivasi dapat dinyatakan dalam:

`turunan=var.deriv()`

Untuk mengetahui nilai turunan pada titik tertentu dapat menggunakan

`Variabel(int)`

Untuk menghitung derivasi ke-n $d^n(y)/dx^n$ dapat dilakukan dengan mengisi nilai di dalam kurung pada var.deriv dengan suatu integer

```
Var.deriv(int)
```

1 .32 Mengaplikasikan Numpy pada Matplotlib

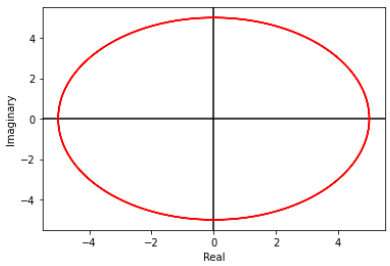
1.32.1 Agrand Diagram pada Numpy

Mengombinasikan numpy dengan matplotlib dalam memproyeksikan fungsi kompleks pada diagram agrand, program diawai dengan memasukkan nilai magnitude pada fungsi kompleks dalam bentuk eksponensial

$e^{j\theta}=r(\cos\theta + j\sin\theta)$

```
In [13]: import numpy as np
         import matplotlib.pyplot as plt

         r=int(input('Masukkan nilai magnitude: '))
         theta= np.linspace(-2*np.pi, 2*np.pi,200)
         y1=r*np.exp(1j*theta)
         y1=np.array(y1)
         plt.axvline(x=0,color='k')
         plt.axhline(y=0,color='k')
         plt.xlabel("Real")
         plt.ylabel('Imaginary')
         plt.plot(np.real(y1), np.imag(y1), color='r')
         plt.show()
         x1=np.real(y1)
         x2=np.imag(y1)
```

```
Masukkan nilai magnitude: 5
```



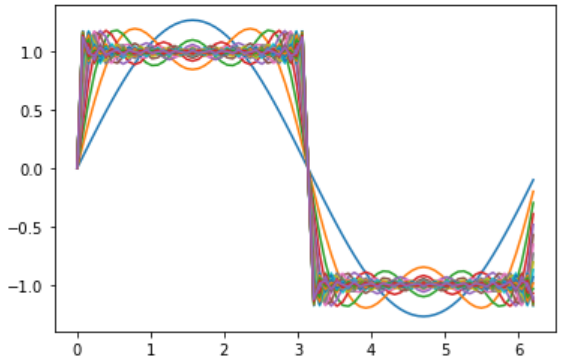
1.32.2 Deret Fourier dengan Numpy

Penerapan matplotlib dan numpy dapat digunakan pada dunia teknik, salah satunya dengan fungsi taylor series yang digunakan untuk menghasilkan fungsi diskrit apabila penjumlahan dilakukan dengan nilai n mendekati tak hingga dari penjumlahan deret fungsi sinusoidal, dengan fungsi awal:

$$\sum_{n=0}^{k} \frac{4}{\pi} \times \frac{1}{n \times \sin(nx)}, n = genap$$

In [40]:
```
import numpy as np
import matplotlib.pyplot as plt
sum=0
x = np.arange(0, 2 * np.pi, np.pi / 40)
n=int(input('Polinomial Series : '))
for angka in range (0, n+1):
    if angka % 2 != 0:
        sum=sum+ 4/np.pi*1/angka*np.sin(angka*x)
        plt.plot(x, sum)
```

```
Polinomial Series : 50
```



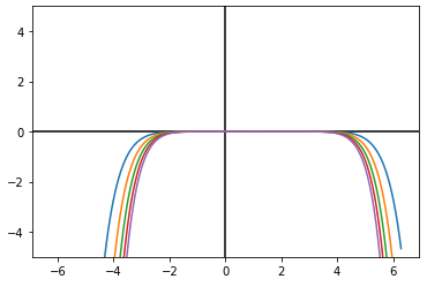
1 .33 Deret McLaurin dan Taylor dengan Numpy

Menggunakan fungsi yang tersedia pada numpy, kita dapat mengunakannya dalam mencari nilai pada suatu fungsi pada titik tertentu yang telah direpresentasikan dalam bentuk deret, sebagai contoh menggunakan numpy dalam menemukan nilai tertentu pada fungsi sinusoidal:

$$\sum_{n=0}^{k}(-1)^n \times \frac{(x-a)^{2n}}{2n!}$$

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

        sum=0
        plt.axvline(x=0,color='k')
        plt.axhline(y=0, color='k')
        plt.ylim(ymax=5, ymin=-5)
        x=np.linspace(-2*np.pi,2*np.pi,200)
        n=int(input('Masukkan banyak suku barisan: '))
        a=int(input('Masukkan pusat fungsi: '))
        for angka in range (0, n):
            sum=sum+((-1)**n)*((x-a)**(2*n))/np.math.factorial(2*n)
            plt.plot(x, sum)
```

```
Masukkan banyak suku barisan: 5
Masukkan pusat fungsi: 1
```



1 .34 Membuat Grafik Sinusoidal

Membuat Grafik Sinusoidal dengan bantuan library Numpy, diawali dengan memasukkan kedua library yakni library Matplot dan library Numpy untuk memungkinka dalam menggunakan fungsi yang berada di dalamnya.

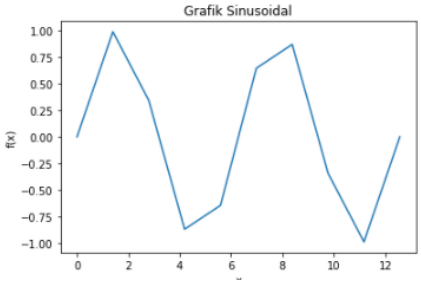
```
In [27]: import matplotlib.pyplot as plt
         import numpy as np
```

Mendeklarasikankan variabel x yang melambangkan nilai x yang akan dimasukkan ke dalam fungsi, menggunakan fungsi

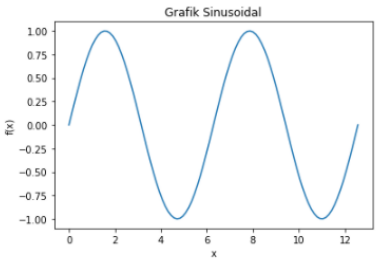
`np.linspace(nilai awal, nilai akhir, banyak suku bilangan)`

```
x=np.linspace(0, 4*np.pi, 1000)
```

Dengan nilai awal adalah nol hingga 4π yang terbagi menjadi 1000 suku bilangan untuk mendapatkan grafik yang jelas dan lurus, semakin banyak suku bilangan, semakin akurat hasil yang didapatkan



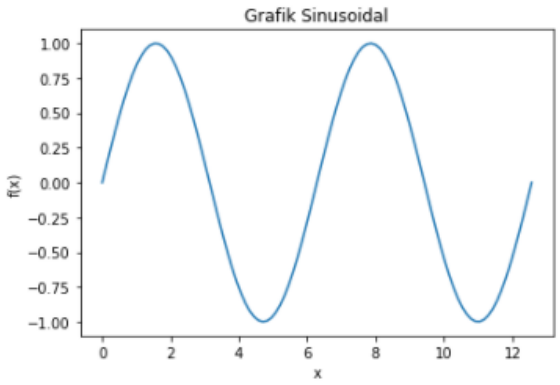
G dengan 10
Suku Bilangan



Grafik dengan 100
Suku Bilangan

Memanggil fungsi sinus dengan menggunakan `np.sin(variabel)` dengan variabel pada dalam adalah variabel yang melambangkan nilai yang ingin dikomposisikan dalam kasus ini nilai x.

```
In [36]: import matplotlib.pyplot as plt
         import numpy as np
         x=np.linspace(0, 4*np.pi, 100)
         y=np.sin(x)
         plt.plot(x,y)
         plt.title('Grafik Sinusoidal')
         plt.xlabel('x')
         plt.ylabel('f(x)')
         plt.show()
```

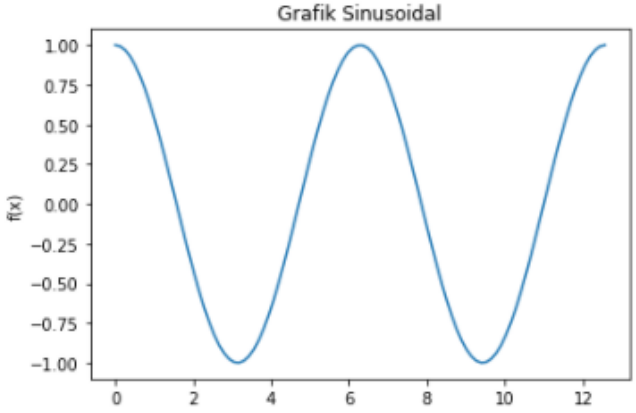


`

Untuk membuat fungsi-fungsi lainnya dapat dengan mengubah fungsi np.sin(x) menjadi fungsi lain.

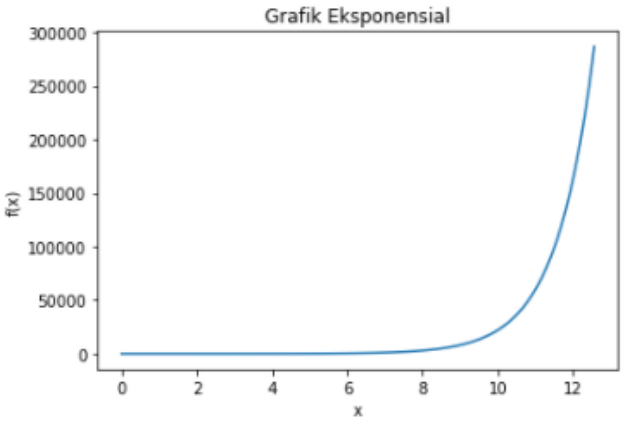
Contoh Grafik Fungsi Sinusoidal

```
In [37]: import matplotlib.pyplot as plt
         import numpy as np
         x=np.linspace(0, 4*np.pi, 100)
         y=np.cos(x)
         plt.plot(x,y)
         plt.title('Grafik Cosinus')
         plt.xlabel('x')
         plt.ylabel('f(x)')
         plt.show()
```



Contoh Fungsi Eksponensial

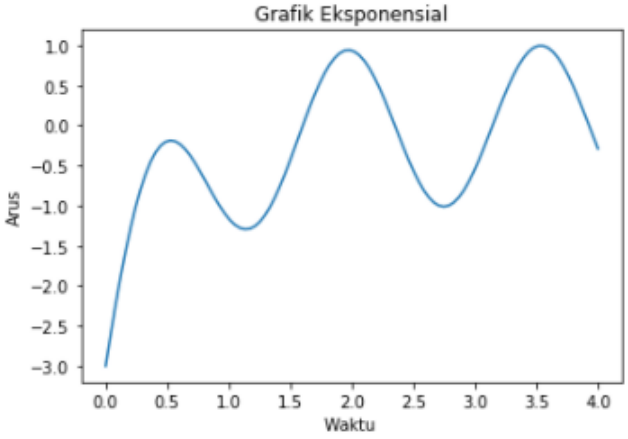
In [41]:
```
import matplotlib.pyplot as plt
import numpy as np
x=np.linspace(0, 4*np.pi, 100)
y=np.exp(x)
plt.plot(x,y)
plt.title('Grafik Eksponensial')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.show()
```



Contoh Grafik Fungsi

$I(t) = \sin(4t) - 3e^{-2t}$

In [68]:
```
import matplotlib.pyplot as plt
import numpy as np

t=np.linspace(0, 4, 100)
Arus=np.sin(4*t)-3*np.exp(-2*t)
plt.plot(t,Arus)
plt.title('Grafik Eksponensial')
plt.xlabel('Waktu ')
plt.ylabel('Arus')
plt.show()
```



1 .35 Membuat Grafik Pie

Membuat grafik pie dengan menggunakan matplotlib, pada contoh dibawah menggambarkan “Pelajaran Tersulit menurut Mahasiswa Teknik Elektro UPN Veteran Jakarta”

fungsi :

`label=’String’, ‘String’`

Digunakan untuk memberikan identitas nama pada data yang diwakilkan

Fungsi:

`Sizes=[45, 30,15, 10]`

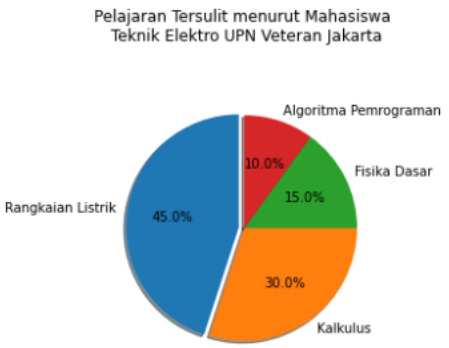
Digunakan untuk menyatakan besar nilai dari data yang di representasikan dalam bentuk persentase dengan total persentase adalah 100%

Fungsi:

`Explode=(0.08,0,0,0)`

Digunakan untuk memberikan efek memisah pada bagian sub grafik yang ingin dipisahkan, untuk memudahkan proses identifikasi terhadap data yang ingin diberikan perhatian khusus/

```
In [17]: import matplotlib.pyplot as plt


labels = 'Rangkaian Listrik', 'Kalkulus', 'Fisika Dasar', 'Algoritma Pemrograman'
sizes = [45, 30, 15, 10]
explode = (0.05,0, 0, 0)
fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.set_title("Pelajaran Tersulit menurut Mahasiswa \n Teknik Elektro UPN Veteran Jakarta\n\n")
plt.show()
```



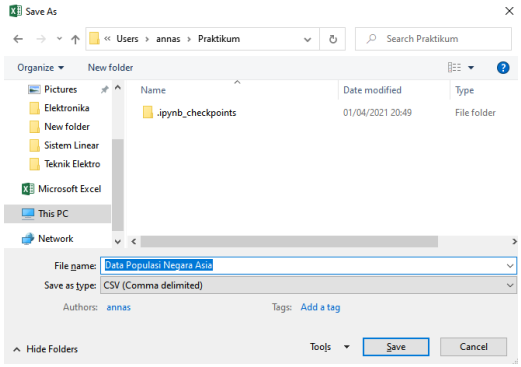
1 .36 Memasukkan Data dengan Panda

Memasukkan kumpulan data yang dimuat dalam format (.csv) untuk dimasukkan ke dalam program agar lebih mudah dalam menyiapkan data.

Buat data dengan bantuan ms.excel kemudian pisahkan data dengan koma (,) untuk membedakan kolom pada data.

	A	B	C
1	country,year,population		
2	Indonesia,1952,82052000		
3	Indonesia,1957,90124000		
4	Indonesia,1962,99028000		
5	Indonesia,1967,109343000		
6	Indonesia,1972,121282000		
7	Indonesia,1977,136725000		
8	Indonesia,1982,153343000		
9	Indonesia,1987,169276000		
10	Indonesia,1992,184816000		
11	Indonesia,1997,199278000		
12	Indonesia,2002,211060000		
13	Indonesia,2007,223547000		

Simpan folder csv ke dalam folder yang sama dengan program berada, umumnya folder akan disimpan secara default di C:\users\(NamaPengguna)\(Nama Folder)



Impor library untuk menggunakan fungsi-fungsi yang dibutuhkan di dalam program seperti membaca dan meroganisir data. Kemudian gunakan program

`pd.read_csv('Nama_folder.csv')`

Untuk membuat program membaca file CSV yang telah diletakkan di satu folder yang sama

```
In [2]: import pandas as pd
        data=pd.read_csv('Data Populasi Negara Asia.csv')
```

Maka Output akan terlihat seperti ini setelah nama variabel dipanggil.

```
In [3]: data
Out[3]:
```

	country	year	population
0	Indonesia	1952	82052000
1	Indonesia	1957	90124000
2	Indonesia	1962	99028000
3	Indonesia	1967	109343000
4	Indonesia	1972	121282000
5	Indonesia	1977	136725000
6	Indonesia	1982	153343000
7	Indonesia	1987	169276000
8	Indonesia	1992	184816000
9	Indonesia	1997	199278000
10	Indonesia	2002	211060000
11	Indonesia	2007	223547000
12	Malaysia	1952	6748378
13	Malaysia	1957	7739235
14	Malaysia	1962	8906385
15	Malaysia	1967	10154878
16	Malaysia	1972	11441462
17	Malaysia	1977	12845381

Jika ingin mengimport data tanpa harus perlu untuk memindahkan data ke dalam folder yang sama, dapat digunakan program seperti ini

```
pasar=pd.read_csv('C:\\Users\\annas\\Downloads\\Data Marketing.csv')
```

Untuk dapat melihat data lakukan pemanggilan variabel yang telah dideklarasikan sebelumnya

	Tanggal	Minggu	Minggu Total	Bulan	Bulan Total	Tahun	Nama Hari	Pengunjung	Penghasilan	Biaya Pemasaran	Promo
0	09/11/2020	46	34	11	11	2020	Senin	707	465	651.375000	Tidak Promo
1	10/11/2020	46	34	11	11	2020	Selasa	1455	10386	1298.250000	Promo Tipe-1
2	11/11/2020	46	34	11	11	2020	Rabu	1520	12475	1559.375000	Promo Tipe-2
3	12/11/2020	46	34	11	11	2020	Kamis	1726	11712	1801.750000	Tidak Promo
4	13/11/2020	46	34	11	11	2020	Jumat	2134	10000	2614.500000	Tidak Promo
...	...	...	...	...	...	...	...	...	...	...	...
177	05/05/2021	19	60	5	17	2021	Rabu	1400	7284	1119.600000	Tidak Promo
178	06/05/2021	19	60	5	17	2021	Kamis	2244	13021	2067.888889	Promo Tipe-1
179	07/05/2021	19	60	5	17	2021	Jumat	2023	4587	1450.200000	Tidak Promo
180	08/05/2021	19	60	5	17	2021	Sabtu	1483	5927	1121.875000	Tidak Promo
181	09/05/2021	20	61	5	17	2021	Minggu	1303	3861	871.000000	Tidak Promo

1 .37 Membuat grafik 3 dimensi dengan Numpy dan Matplotlib

Library matplotlib dapat digunakan untuk membuat grafik tiga dimensi yang merepresentasikan fungsi multivariabel. Program di bawah merupakan program untuk memetakan fungsi f(x,y) = sin(x) + cos (y)

`np.arange :` digunakan untuk memetakan nilai x dan nilai y dengan interval dari -5 pada setiap 0.1 kenaikan hingga 5.

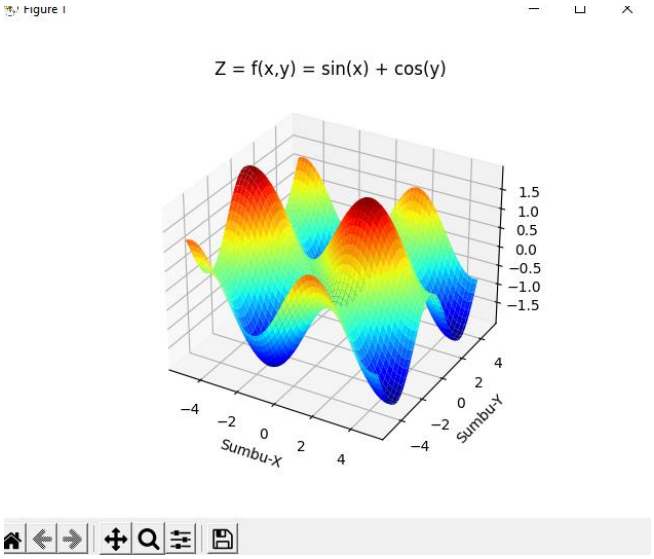
`np.meshgrid(x, y) :` digunakan untuk membuat pasangan koordinat antara nilai x dan nilai y.

`ax.plot_surface(x, y, z, cmap = “jet”) :`

digunakan untuk memplot grafik 3d, dengan memasukkan tiga variabel utama yakni x,y,z dengan parameter cmap untuk menentukan warna dari grafik .

```
import numpy as np
import matplotlib.pyplot as plt
ax = plt.axes(projection='3d')
xn = np.arange(-5, 5, 0.1)
yn = np.arange(-5, 5, 0.1)
X, Y = np.meshgrid(xn, yn)
ax.set_xlabel("Sumbu-X")
ax.set_ylabel("Sumbu-Y")
ax.set_title("Z = f(x,y) = sin(x) + cos(y)")
Z = np.sin(X) + np.cos(Y)
ax.plot_surface(X, Y, Z, cmap="jet")
plt.show()
```

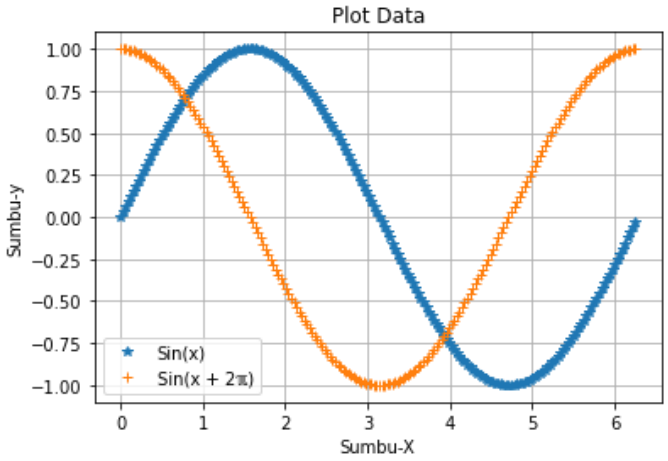
Setelah program di jalankan *pop-out* grafik akan muncul sebagai bentuk visualisasi dari fungsi yang telah dibuat, Pada text editor atau IDE seperti PyCharm atau VisualStudioCode sudut pandang dari grafik ini dapat diatur dengan mengubah sudut azimuth dan altitude namun pada Python notebook hanya akan muncul gambar default, hal ini bertujuan untuk meringankan beban komputasi yang berat pada proses visualisasi.



Analisis Praktikum

1. Fungsi Grafik Eksponen

```
import numpy as np
import matplotlib.pyplot as plt
# Buatlah nilai x dengan interval [0, 2π] dan step π/100
x = np.arange(__, ___, ____)
# Masukkan fungsi y1 = sin(x) dan y2 = cos(x + 2π)
y1 = _____
y2 = _____
plt.plot(__, ___, '*', label='Sin(x)')
plt.plot(__, ___, '+', label='cos(x + 2π)')
plt.xlabel('Sumbu-X')
plt.ylabel('Sumbu-y')
plt.title('Plot Data')
plt.grid()
plt.legend()
plt.show()
```



2. Mencari nilai dari gelombang segitiga dengan menggunakan deret Fourrier

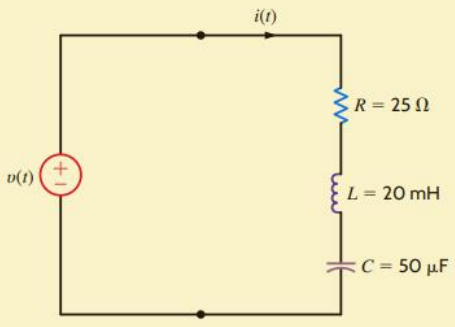
$$\sum_{angka\,=\,1}^{n} \frac{Amplitudo}{\pi} \times \frac{\sin(angka \times \frac{2\pi}{T} t)}{angka}$$

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

t = np.linspace(__ , ___ , ___)
plt.axvline(x = 0, color = 'k')
plt.axhline(y = 0, color = 'k')
plt.xlabel('Time (s)')
plt.ylabel('f(t)')

sum = 0
amplitudo = int(input("Masukkan nilai amplitudo: "))
periode = int(input("Masukkan nilai T : "))
n = int(input("Masukkan banyaknya  aproksimasi : "))

for angka in range(1, n):
  # Masukkan fourier transform di bawah
  sum = sum + _____
  # Masukkan nilai t sebagai x dan Sum sebagai Y
  plt.plot(__, __)
```

3. Mencari nilai kompleks pada suatu rangkaian menggunakan numpy

Determine the equivalent impedance of the network shown in Fig. 8.10 if the frequency is $f = 60$ Hz. Then compute the current $i(t)$ if the voltage source is $v(t) = 50\cos(\omega t + 30°)$ V. Finally, calculate the equivalent impedance if the frequency is $f = 400$ Hz.

Figure 8.10
Series ac circuit.

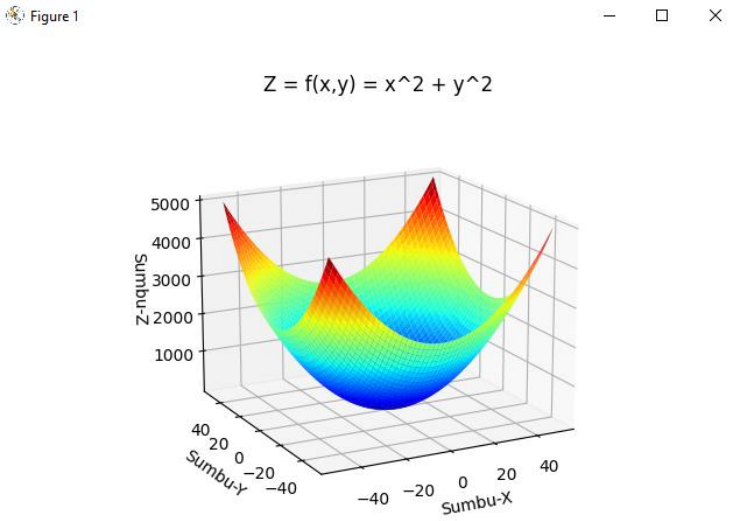


$$\mathbf{I} = \frac{\mathbf{V}}{\mathbf{Z}} = \frac{50\angle 30°}{25 - j45.51} = \frac{50\angle 30°}{51.93\angle -61.22°} = 0.96\angle 91.22° \text{ A}$$

```
import numpy as np
# Diketahui v(t) = 50∠30, L = 20.10^(-3), C = 50.10^(-6)
# Masukkan nilai C, L dan f
f = ___
C = ___
L = ___
W = ___
R = ___
# Menghitung nilai masing-masing impendansi
# Ingat ZL = 1j*W*L, ZC = -1j/(W*C), ZR = R
ZL = ___
ZC = ___
ZR = ___

def ke_kompleks(amplitudo, sudut):
  # A∠θ = A(cos(θ) + j*sin(θ))
  real = amplitudo * np.cos(sudut * (np.pi/180))
  imajiner = 1j*amplitudo * np.sin(sudut * np.pi/180)
  # Tambahkan bilangan real dan imajiner
  bilangan_kompleks = ___ + ___
  return bilangan_kompleks

def ke_polar(fungsi_kompleks):
  # Fungsi kompleks = A(cos(θ) + j*sin(θ))
  #Amplitudo (A) = akar(real^2(fungsi_kompleks) + imajiner^2(fungsi_kompleks))
  # Sudut(θ) = tan^(-1)(bilangan_imajiner/bilangan_real)
  amplitudo = np.sqrt(np.real(___)**2 + np.imag(___)**2)
  sudut = np.angle(fungsi_kompleks, deg=True)
  # Membulatkan 2 digit dibelakang 0
  bilangan_polar = str(round(amplitudo, 2)) + '∠' + str(round(sudut, 2))
  return bilangan_polar

# Ztot = ZL + ZC + ZR
Ztot = ______
# Mengubah v ke dalam kompleks
v = ke_kompleks(50, 30)

# Arus = v / ztot
arus = ____
arus_dalam_polar = ke_polar(arus)
print(f"Nilai arus dalam polar adalah : {arus_dalam_polar}")
```

4. Buatlah fungsi 3d dari $z = f(x,y) = x^2 + y^2$

```
import numpy as np
import matplotlib.pyplot as plt
ax = plt.axes(projection='3d')
# Buatlah interval dari sumbu x dan y [-100,100] dengan kenaikan per 100
xn = np.linspace(___, ___, ___)
yn = np.linspace(___, ___, ___)
X, Y = np.meshgrid(xn, yn)
ax.set_xlabel("Sumbu-X")
ax.set_ylabel("Sumbu-Y")
ax.set_title("Z = f(x,y) = x^2 + y^2")
# Buatlah fungsi Z = x^2 + y^2 dari nilai yang telah di meshgrid
Z = ____
ax.plot_surface(X, Y, Z, cmap="jet")
plt.show()
```

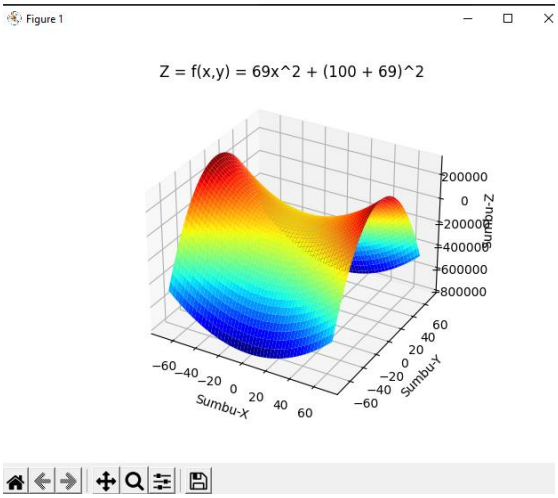


Tugas Akhir:

(Program beserta gambar keluaran dilampirkan pada saat pengumpulan tugas akhir)

1. Buatlah suatu program yang dapat memetakan fungsi multivariabel dari x dan y, dengan interval 2 digit NIM terakhit [-NIM, NIM] dan fungsi :
   $Z = f(x,y) = (NIM)x^2 + (NIM + 100)\ y^2$

Contoh Output:
NIM : 200314069
2 digit terakhir = 69
Interval [-69, 69]
$Z = f(x,y) = (69)x^2 + (169)\ y^2$



contoh

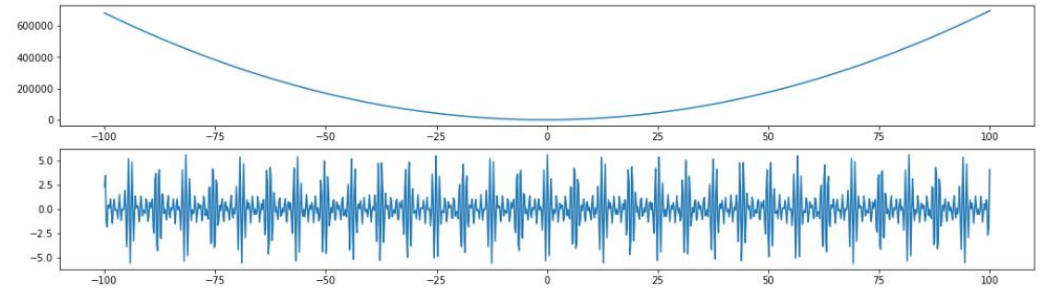
2. Buatlah program dari masing-masing fungsi dibawah dengan NIM sebagai 2 digit terakhir dari Nomer Induk Mahasiswa, Kemudian buatlah plot grafik dari masing-masing fungsi dalam satu keluaran!

$Y = (NIM)\ x^2 + (NIM)\ x + NIM$

$Y = \sum_{n=0}^{3} sin\left((NIM + n)x\right) + \cos((NIM + n)x)$

Contoh Output:
NIM : 200314069
2 digit terakhir = 69
Grafik plot yang dihasilkan pada satu keluaran:
$Y = 69x^2 + 69x + 69$
$Y = \sum_{n=0}^{3} sin\left((69 + n)x\right) + \cos((69 + n)x)$

```
plt.show()
```


600000
400000
200000
0
−100
−75
−50
−25
0
25
50
75
100
5.0
2.5
0.0
−2.5
−5.0
−100
−75
−50
−25
0
25
50
75
100