In [1]:

```
u=2.5+3j
v=2
w=u+v
```

In [2]:

```
from math import sin
r=sin(w)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-2-f2478e858a9e> in <module>()
      1 from math import sin
----> 2 r=sin(w)

TypeError: can't convert complex to float
```

In [20]:

```
from cmath import sin,sinh
from numpy.lib.scimath import *
from cmath import *
from math import sqrt
from cmath import sqrt
import numpy
r=sin(w)
r
```

Out[20]:

```
(-9.841442815345065-2.1117259539209345j)
```

In [11]:

```
r1=sin(8j)
r1
```

Out[11]:

```
1490.4788257895502j
```

In [12]:

```
r2=1j*sinh(8)
r2
```

Out[12]:

```
1490.4788257895502j
```

In [15]:

```
q=8
exp(1j*q)
```

Out[15]:

```
(-0.14550003380861354+0.9893582466233818j)
```

In [16]:

```
cos(q)+1j*sin(q)
```

Out[16]:

```
(-0.14550003380861354+0.9893582466233818j)
```

In [19]:

```
sqrt(-1)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-19-e94865f03ce3> in <module>()
----> 1 sqrt(-1)

ValueError: math domain error
```

In [21]:

```
sqrt(-1)
```

Out[21]:

```
1j
```

Type *Markdown* and LaTeX: α^2

In [8]:

```
import numpy as np
```

In [11]:

```
x=np.array([[1,2],[4,5]])
y=np.array([[1,3],[5,6]])
print(np.add(x,y))
```

```
[[ 2  5]
 [ 9 11]]
```

In [12]:

```
print(np.subtract(x,y))
```

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

In [13]:

```
print(np.divide(x,y))
```

```
[[ 1.         0.66666667]
 [ 0.8        0.83333333]]
```

In [14]:

```
print(np.multiply(x,y))
```

```
[[ 1  6]
 [20 30]]
```

In [15]:

```
print(np.dot(x,y))
```

```
[[11 15]
 [29 42]]
```

In [1]:

```
import numpy as np
matrix = np.matrix([[1,4],[2,0]])
det=np.linalg.det(matrix)
print(det)
```

```
-8.0
```

In [2]:

```
A=([[1,5,6,7],[8,9,1,0],[2,3,4,5],[4,5,2,3]])
A_det=np.linalg.det(A)
print(A)
print(A_det)
```

```
[[1, 5, 6, 7], [8, 9, 1, 0], [2, 3, 4, 5], [4, 5, 2, 3]]
-86.0
```

In [3]:

```
inverse=np.linalg.inv(matrix)
A_inv=np.linalg.inv(A)
print(inverse)
print(A_inv)
B=np.linalg.inv(A_inv)
print(B)
```

```
[[ 0.     0.5  ]
 [ 0.25  -0.125]]
[[-0.39534884  0.09302326  0.65116279 -0.1627907 ]
 [ 0.34883721 -0.02325581 -0.6627907   0.29069767]
 [ 0.02325581  0.46511628  0.75581395 -1.31395349]
 [-0.06976744 -0.39534884 -0.26744186  0.94186047]]
[[  1.00000000e+00   5.00000000e+00   6.00000000e+00   7.00000000e+00]
 [  8.00000000e+00   9.00000000e+00   1.00000000e+00   2.35922393e-16]
 [  2.00000000e+00   3.00000000e+00   4.00000000e+00   5.00000000e+00]
 [  4.00000000e+00   5.00000000e+00   2.00000000e+00   3.00000000e+00]]
```

In [4]:

```
A=([[1,3,2],[2,3,1],[4,2,1]])
B=([[2,4,6],[3,2,1],[7,6,2]])
det_A=np.linalg.det(A)
det_B=np.linalg.det(B)
print(det_A)
print(det_B)
inv_A=np.linalg.inv(A)
inv_B=np.linalg.inv(B)
print(inv_A)
print(inv_B)
AB=np.dot(A,B)
AB_INV=np.linalg.inv(AB)
B_INV_A_INV=np.dot(inv_B,inv_A)
print(B_INV_A_INV)
print(AB_INV)
```

```
-9.0
24.0
[[-0.11111111 -0.11111111  0.33333333]
 [-0.22222222  0.77777778 -0.33333333]
 [ 0.88888889 -1.11111111  0.33333333]]
[[-0.08333333  1.16666667 -0.33333333]
 [ 0.04166667 -1.58333333  0.66666667]
 [ 0.16666667  0.66666667 -0.33333333]]
[[-0.5462963   1.28703704 -0.52777778]
 [ 0.93981481 -1.97685185  0.76388889]
 [-0.46296296  0.87037037 -0.27777778]]
[[-0.5462963   1.28703704 -0.52777778]
 [ 0.93981481 -1.97685185  0.76388889]
 [-0.46296296  0.87037037 -0.27777778]]
```

In [5]:

```
A=([[1,3],[4,9]])
B=([[4,5],[6,8]])
C=([[6,7],[8,1]])
A_B=np.dot(A,B)
B_C=np.dot(B,C)
A_BC=np.dot(A,B_C)
AB_C=np.dot(A_B,C)
print(A_BC)
print(AB_C)
```

```
[[ 364  183]
 [1156  582]]
[[ 364  183]
 [1156  582]]
```

In [6]:

```
E=np.matrix([[1,4,7],[9,1,9],[0,0,1]])
eigvals=np.linalg.eigvals(E)
print(eigvals)
```

```
[ 7. -5.  1.]
```

In [7]:

```
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a)
tri_upper_diag=np.triu(a,k=0)
print(tri_upper_diag)
tri_upper_diag_no_diag=np.triu(a,k=1)
print(tri_upper_diag_no_diag)
tri_upper_diag_no_diag=np.triu(a,k=2)
print(tri_upper_diag_no_diag)
tri_upper_diag_no_diag=np.triu(a,k=3)
print(tri_upper_diag_no_diag)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[1 2 3]
 [0 5 6]
 [0 0 9]]
[[0 2 3]
 [0 0 6]
 [0 0 0]]
[[0 0 3]
 [0 0 0]
 [0 0 0]]
[[0 0 0]
 [0 0 0]
 [0 0 0]]
```

In [8]:

```
A=np.array([[3,1],[1,2]])
B=np.array([9,8])
X=np.linalg.solve(A,B)
print(X)
```

```
[ 2.  3.]
```

In [ ]:

```

```