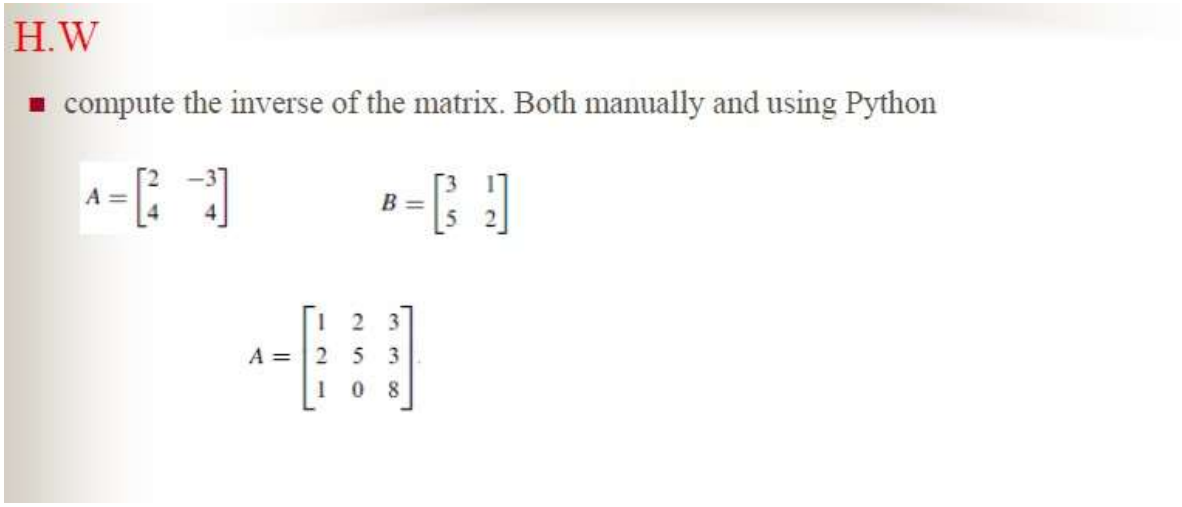


In [2]:
```
import numpy as np
```

In [6]:
```
A = np.array([[2,-3],[4,4]]) # matrix A(2,2)
A
```

Out[6]:
```
array([[ 2, -3],
       [ 4,  4]])
```

In [11]:
```
A_inverse = np.linalg.inv(A)   # inverse matrix A
A_inverse
```

Out[11]:
```
array([[ 0.2 ,  0.15],
       [-0.2 ,  0.1 ]])
```

In [12]:
```
B = np.array([[3,1],[5,2]])   # matrix B(2,2)
B
```

Out[12]:
```
array([[3, 1],
       [5, 2]])
```

In [13]:
```
B_inverse = np.linalg.inv(B)   # inverse matrix B
B_inverse
```

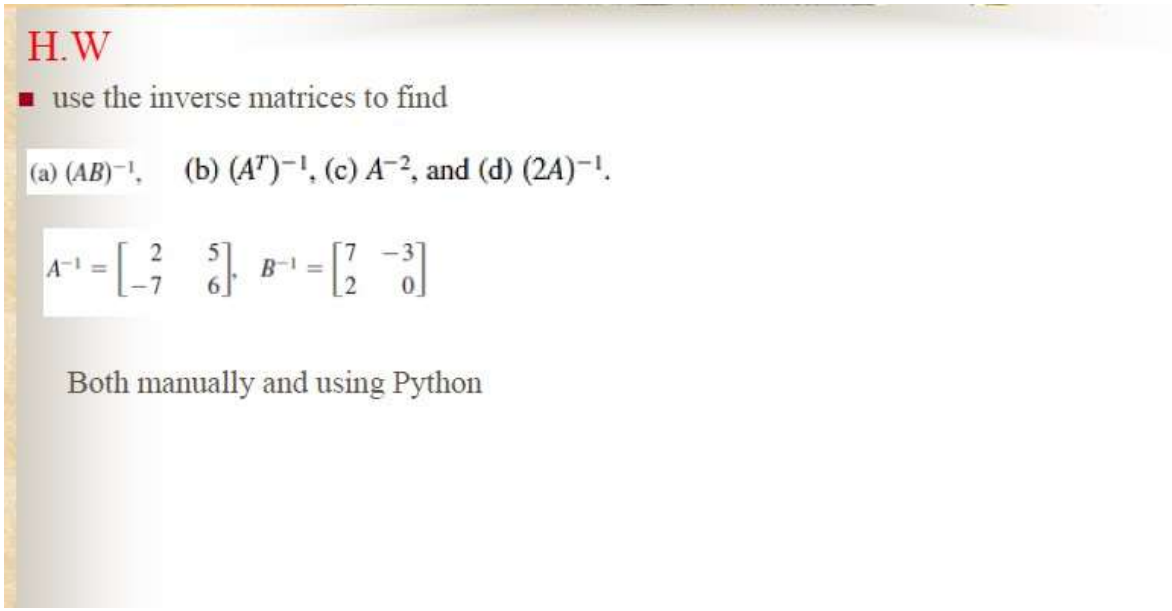
Out[13]:
```
array([[ 2., -1.],
       [-5.,  3.]])
```

In [14]:
```
A = np.array([[1,2,3],[2,5,3],[1,0,8]]) # matrix A(3,3)
A
```

Out[14]:
```
array([[1, 2, 3],
       [2, 5, 3],
       [1, 0, 8]])
```

In [19]:
```
A_inverse = np.linalg.inv(A)   # inverse matrix A
A_inverse
```

Out[19]:
```
array([[-40.,  16.,   9.],
       [ 13.,  -5.,  -3.],
       [  5.,  -2.,  -1.]])
```



In [29]:
```
#a)
AB_inverse = np.dot(np.array([[7,-3],[2,0]]),np.array([[2,5],[-7,6]]))
AB_inverse
```

Out[29]:
```
array([[35, 17],
       [ 4, 10]])
```

In [30]:
```
#b)
A_transpose_inverse = np.array([[2,5],[-7,6]]).T
A_transpose_inverse
```

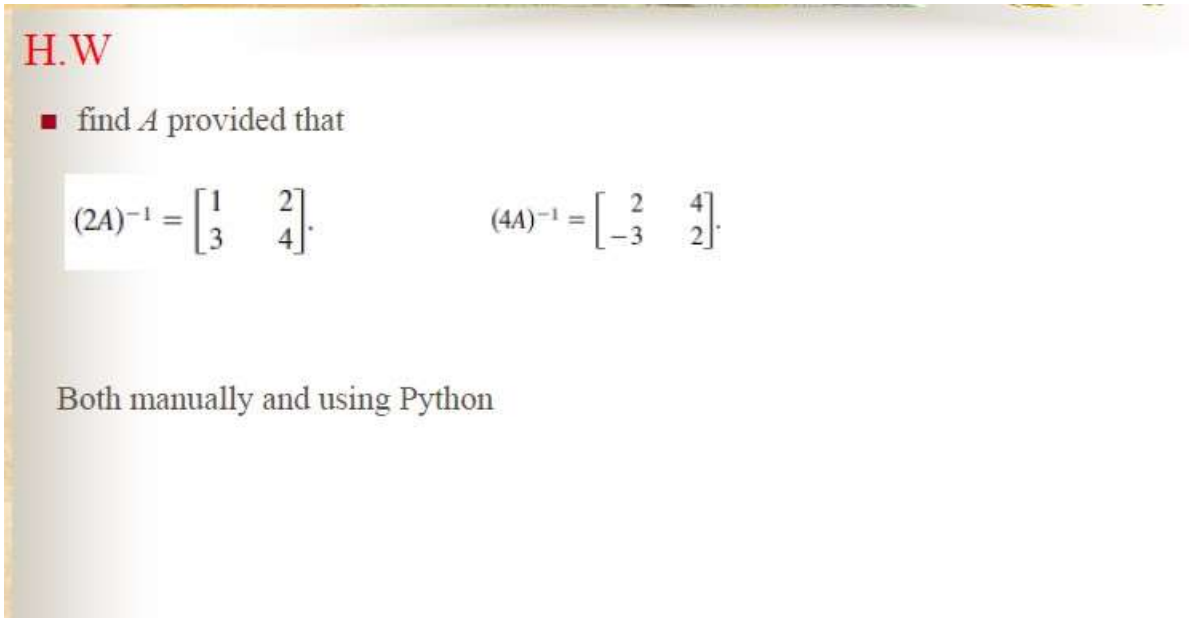
Out[30]:
```
array([[ 2, -7],
       [ 5,  6]])
```

In [31]:
```
#c)
A_pow = np.square(np.array([[2,5],[-7,6]]))
A_pow
```

Out[31]:
```
array([[ 4, 25],
       [49, 36]], dtype=int32)
```

In [33]:
```
#d)
A_inverse_multiply = 1 / 2 * np.array([[2,5],[-7,6]])
A_inverse_multiply
```

Out[33]:
```
array([[ 1. ,  2.5],
       [-3.5,  3. ]])
```



In [34]:
```
#a)
A = 1 / 2 * np.linalg.inv(np.array([[1,2],[3,4]]))
A
```

Out[34]:
```
array([[-1.  ,  0.5 ],
       [ 0.75, -0.25]])
```

In [35]:
```
#b)
A = 1 / 4 * np.linalg.inv(np.array([[2,4],[-3,2]]))
A
```

Out[35]:
```
array([[ 0.03125 , -0.0625  ],
       [ 0.046875,  0.03125 ]])
```

##