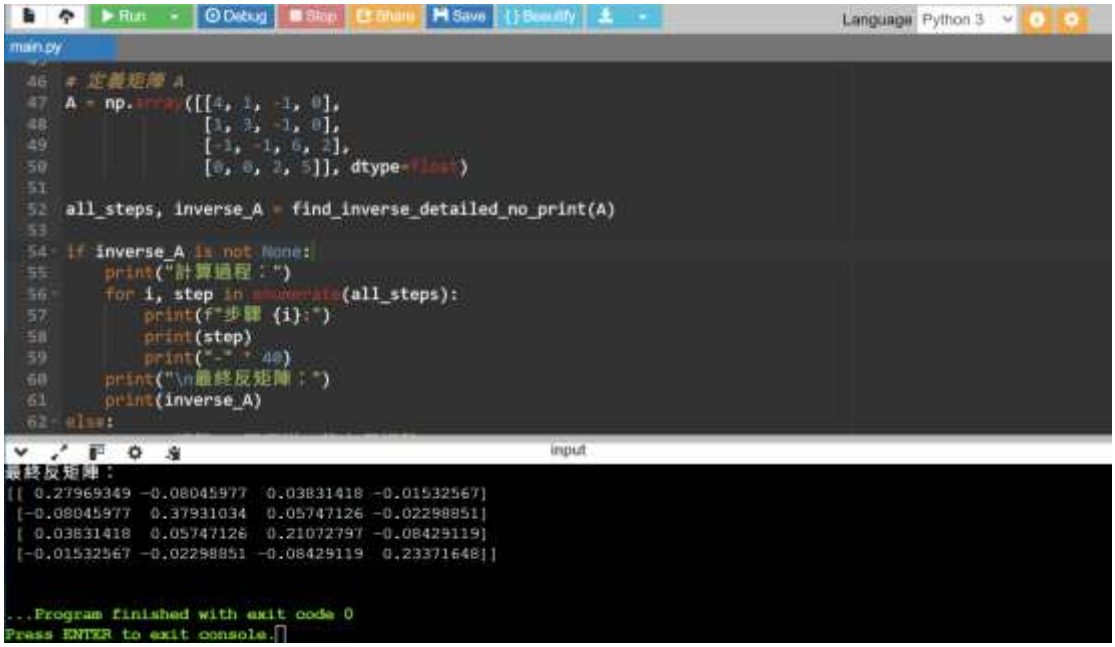
1.

```
    # Back substitution
    x = [0 for _ in range(n)]
    for i in range(n - 1, -1, -1):
        x[i] = matrix[i][n]
        for j in range(i + 1, n):
            x[i] -= matrix[i][j] * x[j]
        x[i] /= matrix[i][i]

    return x

# 增廣矩陣（每列加一欄為等號右邊常數）
aug_matrix = [
    [1.19, 2.11, -100, 1, 1.12],
    [14.2, -0.112, 12.2, -1, 3.44],
    [0, 100, -99.9, 1, 2.15],
    [15.3, 0.110, -13.1, -1, 4.16]
]

solution = gaussian_elimination(aug_matrix)
print("解為：", solution)
```

```
input
解為： [0.1767763295731456, 0.012692102894198573, -0.020661201210554453, -1.1832642903542474]

...Program finished with exit code 0
Press ENTER to exit console.
```

2.



```
# 定義矩陣 A
A = np.array([[4, 1, -1, 0],
              [1, 3, -1, 0],
              [-1, -1, 6, 2],
              [0, 0, 2, 5]], dtype=float)

all_steps, inverse_A = find_inverse_detailed_no_print(A)

if inverse_A is not None:
    print("計算過程：")
    for i, step in enumerate(all_steps):
        print(f"步驟 {i}:")
        print(step)
        print("-" * 40)
    print("\n最終反矩陣：")
    print(inverse_A)
else:
```

```
input
最終反矩陣：
[[ 0.27969349 -0.08045977  0.03831418 -0.01532567]
 [-0.08045977  0.37931034  0.05747126 -0.02298851]
 [ 0.03831418  0.05747126  0.21072797 -0.08429119]
 [-0.01532567 -0.02298851 -0.08429119  0.23371648]]

...Program finished with exit code 0
Press ENTER to exit console.
```

3.


Run
Debug
Stop
Share
Save
{} Beautify
Language
Python 3
main.py
# 前代 Ly = d
y[0] = d[0] / l[0]
for i in range(1, n):
y[i] = (d[i] - c[i] * y[i - 1]) / l[i]
# 回代 Ux = y
x[n - 1] = y[n - 1]
for i in range(n - 2, -1, -1):
x[i] = y[i] - u[i] * x[i + 1]
return x
# 三對角元素
a = [3, 3, 3, 3] # 對角線
b = [-1, -1, -1] # 上對角線
c = [0, -1, -1, -1] # 下對角線（第一個是 0）
d = [2, 3, 4, 1] # 常數項
solution = crout_tridiagonal(a, b, c, d)
print("解 x =", ["{:.5f}".format(val) for val in solution])
input
解 x = ['1.43636', '2.30909', '2.49091', '1.16364']
...Program finished with exit code 0
Press ENTER to exit console.