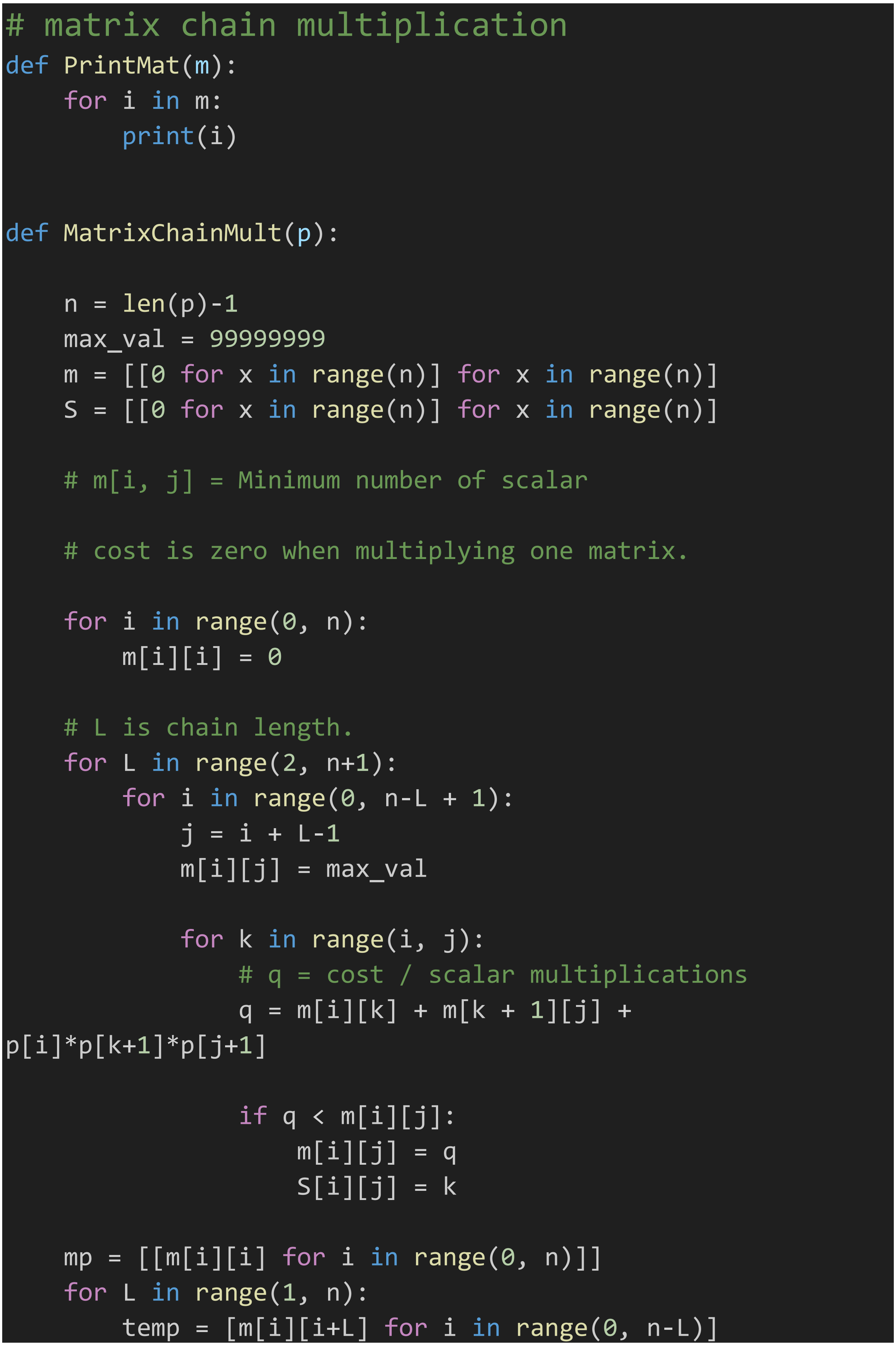
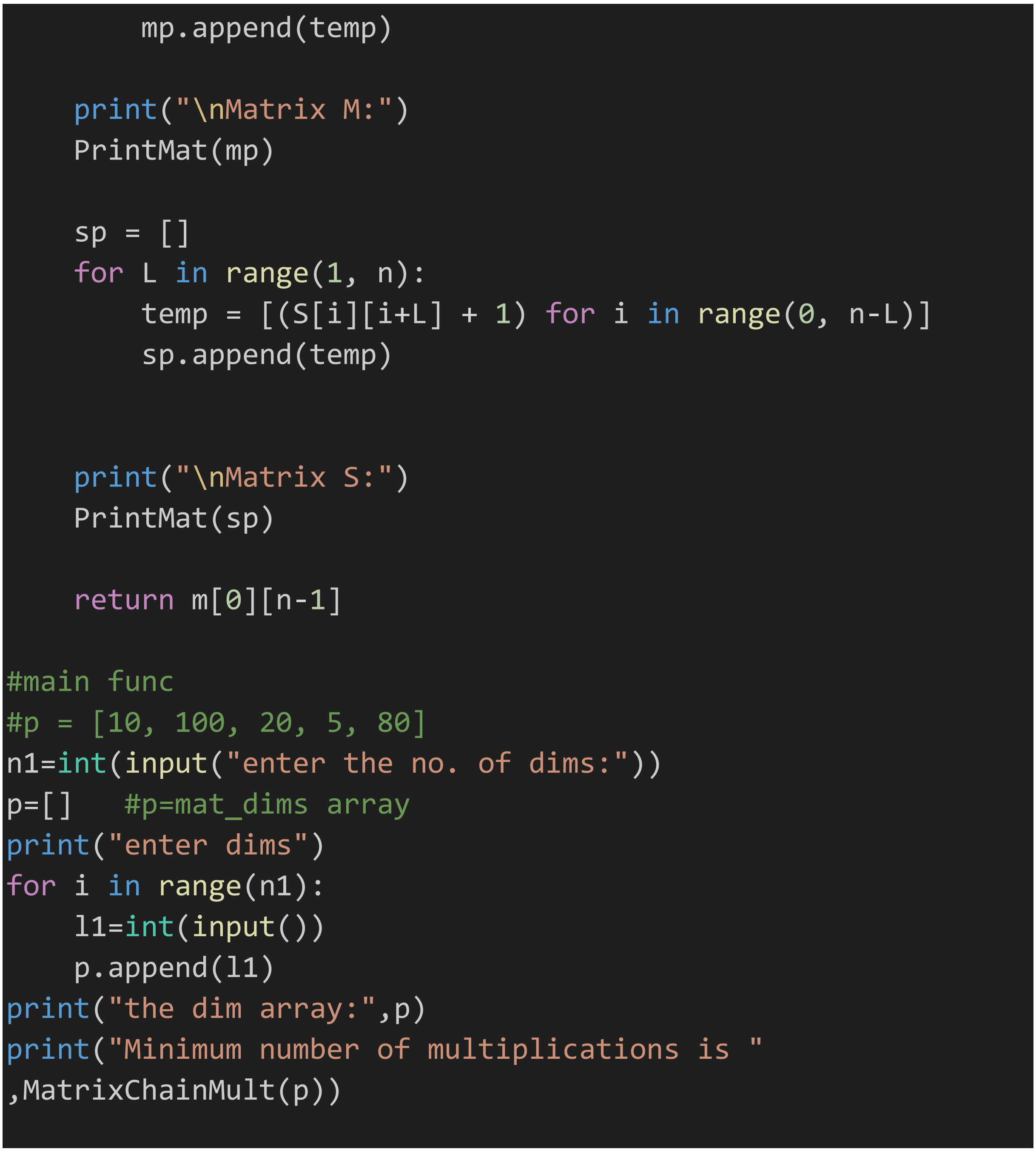
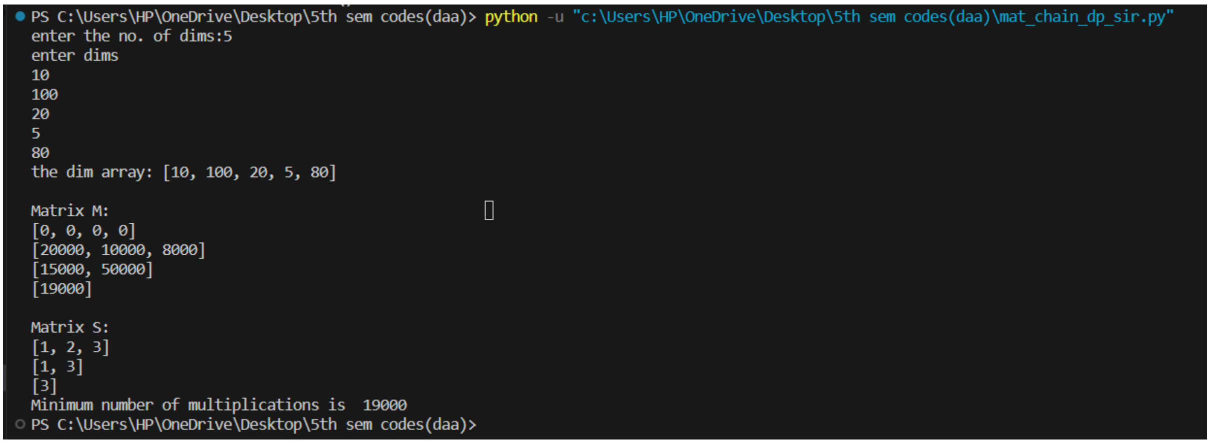


```
# matrix chain multiplication
def PrintMat(m):
    for i in m:
        print(i)


def MatrixChainMult(p):

    n = len(p)-1
    max_val = 99999999
    m = [[0 for x in range(n)] for x in range(n)]
    S = [[0 for x in range(n)] for x in range(n)]

    # m[i, j] = Minimum number of scalar

    # cost is zero when multiplying one matrix.

    for i in range(0, n):
        m[i][i] = 0

    # L is chain length.
    for L in range(2, n+1):
        for i in range(0, n-L + 1):
            j = i + L-1
            m[i][j] = max_val

            for k in range(i, j):
                # q = cost / scalar multiplications
                q = m[i][k] + m[k + 1][j] +
p[i]*p[k+1]*p[j+1]

                if q < m[i][j]:
                    m[i][j] = q
                    S[i][j] = k

    mp = [[m[i][i] for i in range(0, n)]]
    for L in range(1, n):
        temp = [m[i][i+L] for i in range(0, n-L)]
```



```
        mp.append(temp)

    print("\nMatrix M:")
    PrintMat(mp)

    sp = []
    for L in range(1, n):
        temp = [(S[i][i+L] + 1) for i in range(0, n-L)]
        sp.append(temp)


    print("\nMatrix S:")
    PrintMat(sp)

    return m[0][n-1]

#main func
#p = [10, 100, 20, 5, 80]
n1=int(input("enter the no. of dims:"))
p=[]   #p=mat_dims array
print("enter dims")
for i in range(n1):
    l1=int(input())
    p.append(l1)
print("the dim array:",p)
print("Minimum number of multiplications is "
,MatrixChainMult(p))
```



```
● PS C:\Users\HP\OneDrive\Desktop\5th sem codes(daa)> python -u "c:\Users\HP\OneDrive\Desktop\5th sem codes(daa)\mat_chain_dp_sir.py"
  enter the no. of dims:5
  enter dims
  10
  100
  20
  5
  80
  the dim array: [10, 100, 20, 5, 80]

  Matrix M:
  [0, 0, 0, 0]
  [20000, 10000, 8000]
  [15000, 50000]
  [19000]

  Matrix S:
  [1, 2, 3]
  [1, 3]
  [3]
  Minimum number of multiplications is  19000
○ PS C:\Users\HP\OneDrive\Desktop\5th sem codes(daa)>
```