

```

#mcm dp
def matrix_chain(p):
    max_val = 99999999
    n = len(p) - 1 #n is calculated as the length of p
    minus 1
    M = [[0] * (n + 1) for i in range(n + 1)] #M is
    initialized as a 2D list of size (n+1) x (n+1) filled with
    zeros
    S = [[0] * (n) for i in range(2, n + 1)] #S is
    initialized as a 2D list of size (n-1) x n filled with zeros

    for i in range(1, n + 1):
        M[i][i] = 0 #Initializes the main diagonal of
        matrix M with zeros. This is because when multiplying a
        single matrix, the result is zero.

        #The nested loops iterate over the lower triangle of
        matrix M
        for l in range(2, n + 1): #This outer loop iterates
        over the length of the matrix chain, starting from 2 and
        going up to n.
            for i in range(1, n - l + 2): #This inner loop
            iterates over the possible starting points of the
            subchains. The range ensures that the subchain has at least
            two matrices.
                j = i + l - 1 #Calculates the end point of
                the current subchain.
                M[i][j] = max_val
                for k in range(i, j): #This loop iterates over
                the possible split points within the current subchain.
                    q = M[i][k] + M[k + 1][j] + p[i - 1] * p[k]
                    * p[j] #Calculates the result of multiplying the matrices
                    in the current subchain and adding the cost of multiplying
                    the resulting subchains.
                    # M[i][k] represents the result of the
                    subchain from i to k.
                    # M[k + 1][j] represents the result of the
                    subchain from k+1 to j.

```

```

        # p[i - 1] * p[k] * p[j] represents the
actual matrix multiplication result.
        if M[i][j] > q: #Checks if the calculated
result q is less than the current result stored in M[i][j].
            M[i][j] = q #Updates the result matrix
with the minimum result for the current subchain
            S[i-1][j-1] = k #Updates the split
matrix S with the optimal split point for the current
subchain. The indices are adjusted to start from 0.

```

```

print("\nmatrix M:")
for i in M:
    print(i)

```

```

print("\nmatrix S:")
for j in S:
    print(j)

```

```

    return M[1][n] #The index [1][n] specifically
represents the cost of multiplying the matrices from the
first matrix to the nth matrix.

```

```

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

result = matrix_chain(p)
print("result:",result)

```

File Edit Selection View Go Run Terminal Help

5th sem codes(daa)

EXPLORER

- 5th sem codes(daa)
 - knapsack.cpp
 - knapsack.exe
 - knapsack new.cpp
 - knapsack new.exe
 - lcs.py
 - mat_mul_large_dim.py
 - matrix_mul_daa.cpp
 - max_heap_all_op_new.cpp
 - max_heap_all_op_new.exe
 - max_heap_all_op.cpp
 - max_heap_all_op.exe
 - max_heapify.cpp
 - max_heapify.exe
 - max_heapify_p2.cpp
 - max_heapify_p2.exe
 - mcm_dp.py
 - merge_sort_daa.cpp
 - merge_sort_daa.exe
 - merge_sort_other_daa.cpp
 - merge_sort_other_daa.exe
 - merge_sort_p2.cpp
 - merge_sort_p2.exe
 - prims_algo.cpp
 - prims_algo.exe
 - prims.py
 - print_subsequence_of_lcs.py
 - quick_sort_daa.cpp
 - quick_sort_daa.exe
 - quick_sort_p2.cpp
 - quick_sort_p2.exe
- OUTLINE
- TIMELINE

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

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

matrix M:
[0, 0, 0, 0, 0]
[0, 0, 20000, 15000, 10000]
[0, 0, 0, 10000, 50000]
[0, 0, 0, 0, 8000]
[0, 0, 0, 0, 0]

matrix S:
[0, 1, 1, 3]
[0, 0, 2, 3]
[0, 0, 0, 3]
[0, 0, 0, 3]
[0, 0, 0, 3]

result: 19000
PS C:\Users\HP\OneDrive\Desktop\5th sem codes(daa)>
```

Ln 46, Col 24 (2554 selected) Spaces: 4 UTF-8 CRLF Python 3.11.1 64-bit Go Live

18:29 30-11-2023