**SUMMARY**

## USC ID/s: 2143749103, 4727109268, 4371197245

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS (Basic) | Time in MS (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 0 | 0 | 14172 | 14532 |
| 64 | 1.997232 | 1.997709 | 14176 | 14396 |
| 128 | 2.997637 | 8.998394 | 14196 | 14412 |
| 256 | 19.0022 | 29.97828 | 14620 | 14444 |
| 384 | 33.00333 | 236.9974 | 15500 | 14564 |
| 512 | 166.9965 | 143.0094 | 16892 | 15016 |
| 768 | 168.0019 | 387.995 | 20296 | 15420 |
| 1024 | 299.0015 | 599.0002 | 24844 | 15692 |
| 1280 | 566.9997 | 1035.977 | 30940 | 15612 |
| 1536 | 1076.993 | 1415.993 | 38280 | 15780 |
| 2048 | 1292.026 | 2401.023 | 57204 | 15344 |
| 2560 | 1947.02 | 4241.996 | 81940 | 15764 |
| 3072 | 3743.002 | 5636.973 | 109924 | 16184 |
| 3584 | 4435.03 | 9122.021 | 143408 | 16596 |
| 3968 | 4987.977 | 8392.024 | 172488 | 16324 |

## Datapoints

## Insights

### Graph1 – Memory vs Problem Size (M+N)

Chart, line chart

Description automatically generated

#### Nature of the Graph (Logarithmic/ Linear/ Exponential)

Basic: Exponential

Efficient: Linear

#### Explanation:

As we can see from the graph, the basic dynamic programming algorithm takes exponentially more memory as the problem size is increased because it requires to create a memorization table of size m\*n whereas the efficient algorithm which uses a divide and conquer approach with dynamic programming, we will use only 2 \* max(m,n) space which explains the linear memory growth.

### Graph2 – Time vs Problem Size (M+N)

Chart, line chart

Description automatically generated

#### Nature of the Graph (Logarithmic/ Linear/ Exponential)

Basic:

Efficient:

#### Explanation:

## Contribution

(Please mention what each member did if you think everyone in the group does not have an equal contribution, otherwise, write “Equal Contribution”)

<USC ID/s>: <Equal Contribution>

2143749103 : Equal Contribution

4727109268 : Equal Contribution

4371197245 : Equal Contribution