**SUMMARY**

## USC ID/s: **2119838999 and XXXXXXXXXX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS (Basic) | Time in MS (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 1.82 | 1.00 | 19740 | 34844 |
| 64 | 3.34 | 4.47 | 19676 | 34700 |
| 128 | 6.32 | 16.83 | 19912 | 34832 |
| 256 | 11.75 | 58.62 | 20052 | 34744 |
| 384 | 30.10 | 145.14 | 20920 | 34828 |
| 512 | 50.93 | 271.94 | 22252 | 34712 |
| 768 | 107.40 | 538.15 | 25480 | 34908 |
| 1024 | 201.10 | 1000.70 | 29968 | 34788 |
| 1280 | 307.89 | 1639.98 | 35852 | 34968 |
| 1536 | 484.76 | 2344.72 | 43696 | 34936 |
| 2048 | 949.25 | 3845.86 | 62212 | 35068 |
| 2560 | 1450.63 | 6009.93 | 85788 | 34900 |
| 3072 | 2005.15 | 7983.77 | 113720 | 35128 |
| 3584 | 2580.36 | 11215.98 | 150520 | 35248 |
| 3968 | 2660.43 | 12296.55 | 176296 | 35132 |

## Datapoints

## Insights

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

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

Basic: polynomial (quadratic)

Efficient: linear

#### Explanation:

The basic algorithm takes up a polynomial factor of memory space with respect to input size where m and n are the lengths of the input strings. This is due to storing the complete optimization array with dimensions m\*n. The efficient algorithm takes a linear amount of memory space since it only stores 2 columns of the optimization array at any given time. Looking at the graph, it is clear that the efficient algorithm takes up much less memory than the basic algorithm at larger input sizes, though there is an early inflection point where the basic algorithm takes up less space. This is due to the base case of the efficient algorithm running the basic version, resulting in extra memory in the efficient algorithm for small input sizes.

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

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

Basic: polynomial

Efficient: polynomial

#### Explanation:

Both algorithms have polynomial runtimes with respect to input size. The basic algorithm runtime grows slower than the efficient algorithm as input sizes increase. This demonstrates the trade-off between space and time complexity, as the efficient algorithm is more space efficient while the basic algorithm is more time efficient for larger inputs.

At least twice as much time for efficient

## 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>

2119838999 (Claire) equal contribution

XXXXXXXXXX (Robert) equal contribution