104062261 Algorithm HW11 Report

We create a (n+1)\*(n+1) table to save the longest length, then we make all [0][0…n] and [0…n,0] entry equal to zero.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 1 | 5 |
|  | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 |  |  |  |  |  |
| 2 | 0 |  |  |  |  |  |
| 5 | 0 |  |  |  |  |  |
| 5 | 0 |  |  |  |  |  |
| 3 | 0 |  |  |  |  |  |

Next, we follow three rules to fill the table:

1. If S1[n-1] = S2[m-1], table[n][m] = table[n-1][m-1] + 1
2. Else if table[n-1][m] ≥ table[n][m-1], table[n][m] = table[n-1][m]
3. Else table[n][m] = table[n][m-1]

After that, we will get a table like this.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 1 | 5 |
|  | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 2 | 0 | 1 | 2 | 2 | 2 | 2 |
| 5 | 0 | 1 | 2 | 2 | 2 | 3 |
| 5 | 0 | 1 | 2 | 2 | 2 | 3 |
| 3 | 0 | 1 | 2 | 3 | 3 | 3 |

And the table last element will be our answer, that is table[n][n] = 3.

The running time of each entry in the table is , since we have comparison, so the algorithm running time is . And the worst-case running time also is .