Algorithm 1: Dynamic_Programming_Single_Matrix

Input: S1 , S2

```
/* Parameters :                                                    */
/*    s1,s2 :  strings.                                           */
/* Return :                                                       */
/*    ED, alignment                                               */
```

```
begin
    n ←len(s1) ; m ←len(s2)
    define dist_mat : array with first row 0 to m and first column 0 to n
    define alignment : empty array
    if (s1 == s2) then
        return {"ed" : 0 , "alignment" : "match" for all letter };
    else
        for j ←1 to m+1 do
            for i ←1 to n+1 do
                if s1[i-1] == s2[j-1] then
                    dist_mat[j,i] = dist_mat[j-1 , i-1]// "keep" this letter
                else
                    dist_mat[j,i] = 1+min{ dist_mat[j,i-1]// "remove" letter
                                           dist_mat[j-1,i]// "add" letter
                                           dist_mat[j-1,i-1]// "substitute" letter
    alignment ←retrieve alignment by backtracking dist_mat;
    return {"ed" : dist_mat[-1,-1] , "alignment" : alignment };
```