String Alignment

(EDIT DISTANCE)

teh the

Given: two strings x, y over same alphabet Σ,

operations $op_1, \dots, op_k$
costs $cost_1, \dots, cost_k$

Goal: Find the min-cost sequence of operations transforming x into y.

For today: our operations will be

substitution (single-letter)	insertion	deletion
e.g. "go" → "so"		
cost 1	cost 1	cost 1

e.g. $x =$ THEIR $y =$ THERE

```
THEIR
|||ss      cost 2
THERE
```

```
THEIR-     ← not part of alphabet
|||d|i     cost 2
THE-RE
```

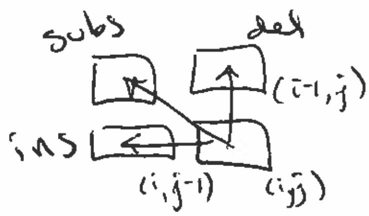
optimal substructure property ✓

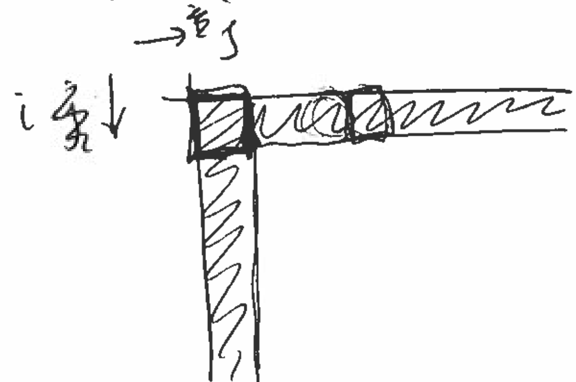
overlapping sub-problems property

proposed subproblems (i,j) align $x[1..i]$ w/ $y[1..j]$.

$\text{cost}(i,j)$ = min cost alignment of $x[1..i]$ w/ $y[1..j]$.

$$= \min \begin{cases} \text{cost}(i-1,j-1) + \text{cost}(\text{subs}) \\ \text{cost}(i,j-1) + \text{cost}(\text{ins}) \\ \text{cost}(i-1,j) + \text{cost}(\text{del}) \end{cases}$$



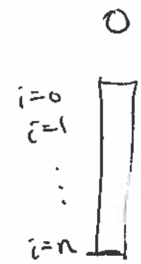


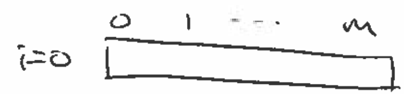
$\text{cost}(0,0) = 0$ (ok base case)

(Note: $\text{cost}(0,j)$ = cost to align "" w/ $y[1..j]$.

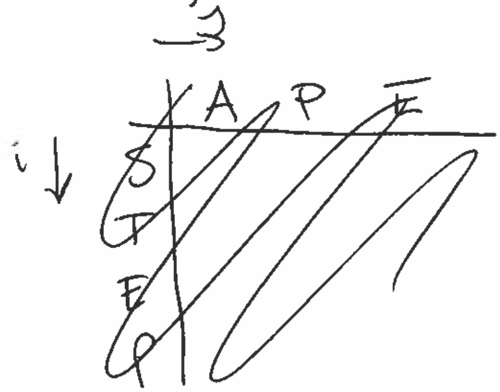
Also works to say $\text{cost}(0,j) = j$ $\text{cost}(i,0) = i$.

```
align (x,y):
    n = len(x)
    m = len(y)
                (n+1)x(m+1)
    table                    array
    aux           "     "     "

    // Base cases
    for i = 0 to n
        table (i,0) = i
        aux (i,0) = (i-1, 0)    (if i ≥ 1)
    for j=0 to m
        table (0,j) = j
        aux (0,j) = (0, j-1)    (if j ≥ 1)

    for i = 1 to n
        for j = 1 to m
            poss1 = table(i-1, j-1) + 1      // subs
            poss2 = table (i, j-1) + 1       // ins
            poss3 = table (i-1, j) + 1       // del
            table (i,j) = min (poss1, poss2, poss3)
            aux (i,j) = if pos1 was min then (i-1, j-1)
                        "   poss2 "   "   "  (i, j-1)
                        "   poss3 "   "   "  (i-1, j)

    return table(n,m) , (alignment)
              ↑
          overall cost
```





④ e.g.: $x = STEP$ $\quad y = APE$



table

	-	A	P	E
-	0	1	2	3
S	1	$\boxed{1}$	2	
T	2			
E	3			
P	4			

aux

ins

	-	A	P	E
-		(0,0)	(0,1)	(0,2)
S	(0,0)	(0,0)	(0,1)	
T	(1,0)			
E	(2,0)			
P	(3,0)			

another aux table

	-	A	P	E
-	·	i	i	i
S	d	S	S	
T	d			
E	d			
P	d			

$i=1,\ j=1$

subs $\quad poss1 = table(0,0)+1 = 0+1 = 1$

ins $\quad poss2 = table(1,0)+1 = 1+1 = 2$

del $\quad poss3 = table(0,1)+1 = 1+1 = 2$

```
- S
i
A

S
d ?
- A
```

$i=1,\ j=2$

subs $\quad table(0,1)+1 = 1+1 = 2$

ins $\quad table(1,1)+1 = 1+1 = 2$

del $\quad table(0,2)+1 = 2+1 = 3$

```
- S
? s
A P

S -
? i
A P
```

Claim: Recurrence relation correctly computes min cost.

Proof: By induction on i and j.

Base case: $(i,j) = (0,0)$. In this case, we're aligning the empty string w/ itself, which has cost 0, and our base case of the recurrence correctly says $cost(0,0) = 0$. ✓

IH: $cost(i', j')$ is correct whenever $i' < i$ or $j' < j$.

Inductive step: Goal is to show $cost(i,j)$ is correct (assuming IH).

Consider the last op in an optimal alignment of $x[1..i]$ w/ $y[1..j]$.

Case 1: Last op is a subs.

In this case, the cost of the alignment is

$$c(\text{subs}) + \underbrace{\text{cost of aligning } x[1..i-1] \ \& \ y[1..j-1]}_{\text{by IH } = cost(i-1,j-1)}.$$

(applies b/c $i-1 < i$)

$= cost(i-1, j-1) + c(\text{subs})$

~~If this is~~ Because this was an opt alignment, this must be the min in the recurrence, so the recurrence relation correctly assigns $cost(i,j) = cost(i-1,j-1) + c(\text{subs})$.