COP 4516 2/10/26 DP2

1) Edit D Quick

2) MCM Show Splits

3) Increasing Subsequence

~~s1~~ s ~~s2~~ t adding letter
deleting letter
Δ letter

coat → acorn
↓1
acoat →(1) acort →(1) acorn

$f(i,j)$ = min cost to transform 1st i letters of s into 1st j letters of t.

relevant state

if $s[i-1] == t[j-1]$, $f(i,j) = f(i-1, j-1)$

else 3 options
→ delete last letter s (1)
→ delete last letter t (1)
→ Δ last letter s to Δ L t (1)

$$f(i,j) = \min \begin{cases} 1 + f(i-1,j-1) \\ f(i-1,j) \\ f(i,j-1) \end{cases}$$

Matrix Chain Multiplication

$((2\times5)(5\times1))((1\times4)(4\times6))$

$M_1 \quad M_2 \quad M_3 \quad M_4$

$r_1 \times c_1$ times $c_1 \times c_2$

$COST = r_1 c_1 c_2$

result has dimension $r_1 \times c_2$

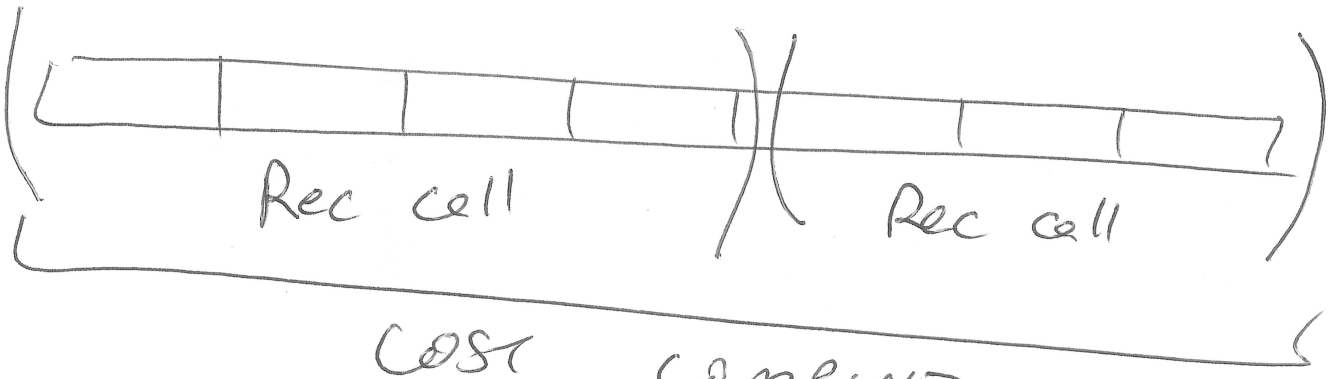
COST 10

$(2\times1) \times (1\times6)$

COST 24

COST 12

TOTAL COST = 10 + 24 + 12 = 46



$$f(i,j) = \min\Big(f(i,i) + f(i+1,j) + cost(i,i+1,j),$$
$$f(i,i+1) + f(i+2,j) + cost(i,i+2,j), \ldots$$
$$f(i,i+2) + f(i+3,j) + cost(i,i+3,j), \ldots$$
$$f(i,j-1) + f(j,j) + cost(i,j,j) \Big)$$

```
res = ∞
for(int split = i; split < j; split++)
    res = min(res, f(i, split) + f(split+1, j) +
              cost(i, split+1, j));
return memo[i][j] = res;
```

Regular LIS

✓ x x ✓ ✓ # ✓

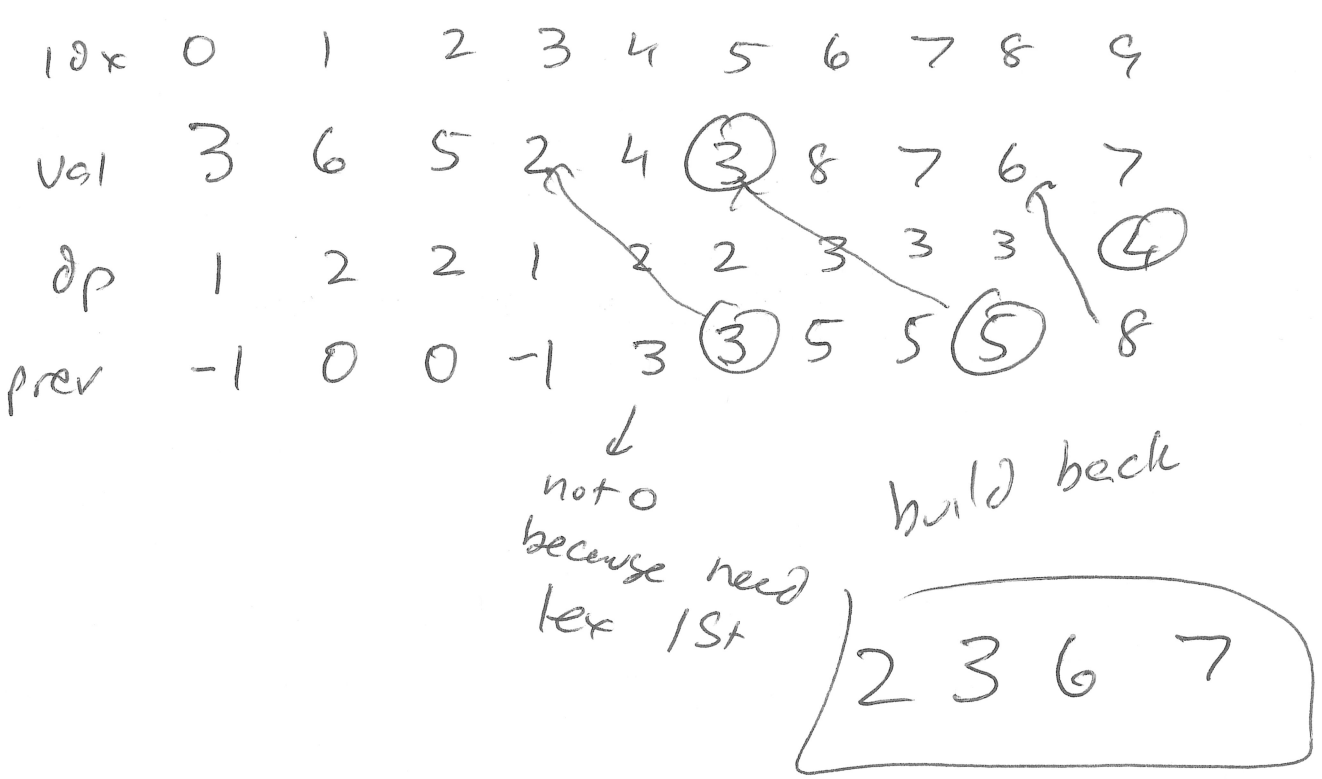
3, 6, 5, 2, 4, 3, 8, 7, 6, 7

1 2 2 1 2 2 3 3 3 4

```
int LIS(int[] array, ~~int n~~) {
    int n = array.length;
    int[] dp = new int[array.length];
    // dp[i] will be longest increasing
    // seq ending at index i
    dp[0] = 1;
    for (int i = 1; i < n; i++) {
        dp[i] = 1; // only #
        for (int j = 0; j < i; j++) {
            if (array[j] >= array[i]) continue;
            * dp[i] = Math.max(dp[i], dp[j]+1);
        }
    }
    int res = dp[0];
    for (int i = 1; i < ~~array~~ n; i++)
        res = Math.max(res, dp[i]);
    return res;
}
```

How to STORE SEQ

// prev[i] = j;



not 0 because need lex 1st

build back

2 3 6 7