

A B , Max | A ⊆ B | ... ?

s1 = KwakKwokKwikKwekKwok

s2 = KwakKwikKwekKwokKwokKwekKwik

subsq1 = KwakKwikKwek~~Kwok~~

subsq2 = KwakKwokKwok

subsq3 = KwikKwekKwok

subsq4 = KwikKwek

dst

$$LCS(s_1, s_2) = 4$$

$$subsq3 = subsq4 + kwok$$

$$|subsq3| = 3$$

$$subsq2 = kwak + subsq3$$

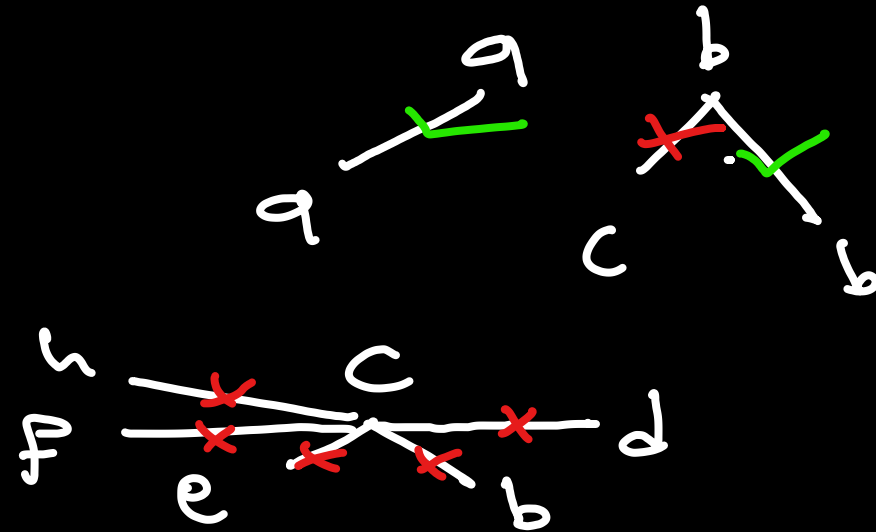
$$|subsq3| = 4$$

s1 = "abcdefgh"  
s2 = "acbdefh"

$lcs(s_1, s_2) =$

$s_1[i] == s_2[j] \rightarrow ambil$

```
for(char cs1 : s1){
  for(char cs2 : s2){
    if(cs1 == cs2) ambil
    lanjut next cs1
  }
}
```



...

$m = m - 1$        $m = 7$   
 a b c d e f g h  
 a c b d e f h  
 $n = 7$

The diagram illustrates a sequence of characters and their alignment in a dynamic programming context. The top row shows the sequence 'a b c d e f g h'. The bottom row shows 'a c b d e f h'. The character 'f' in the bottom row is highlighted with a yellow box, and the character 'h' in the bottom row is highlighted with a green box. A red 'x' is placed between the yellow and green boxes, indicating a mismatch. Above the 'g' and 'h' in the top row, the text ' $m = m - 1$ ' and ' $m = 7$ ' is written. To the right of the 'h' in the bottom row, the text ' $n = 7$ ' is written. A green checkmark is visible to the right of the 'h' in the top row.

LCS  $\rightarrow$  max  $\begin{cases} m \text{ nya kita geser} \\ n \text{ nya kita geser} \end{cases}$

Seq = { 1, 1, 9, 0, 6, 3, 1, 2, 4 }  $\rightarrow$  LIS()

IS =

1 9



LIS  $\rightarrow$  strictly increasing

$LIS[i] \neq LIS[j]$

1

1 6

1 3

Longest Non-decreasing subseq  
 $(\forall) LNDS[i] \neq LNDS[j]$

1 2 9  $\rightarrow$  Longest }  $\equiv$   
 1 3 9  $\rightarrow$  Longest }  $\equiv$

# 0 - 1 Knapsack problem



$N$  barang  
barang ke -  $i$  mempunyai berat sebesar  $W_i$ ,  
mempunyai nilai kegunaan  $C_i$

Kita diberikan kapasitas sebuah ransel  
mampu menampung maksimal sebesar  $M$   
kg,

Berapa nilai kegunaan maksimal yang bisa  
kita peroleh?

$n$	<u>Barang</u>	<u>Berat</u>	<u>NK</u>
1	Botol Minum ✓	5kg	5 1/0
2	Hp ✓	2kg	7 1/0
↑ 3	Buku	10kg	3 1/0
↑ ↑	Snack ✓	3kg	2 1/0

↑ NK Max = ... ?

14 =

\* Berat Min — NK Max  
\* knapsack problem

