Proof of correctness:-

In DP then it is implicit inductive proof.

In inductive proof we try to proof something for n and then for n+1 by using base case.

In 0/1 knapsack problem, we started with base case (No item).

Inductive step uses equations. The value of i, j is based on the value of smaller i & j.

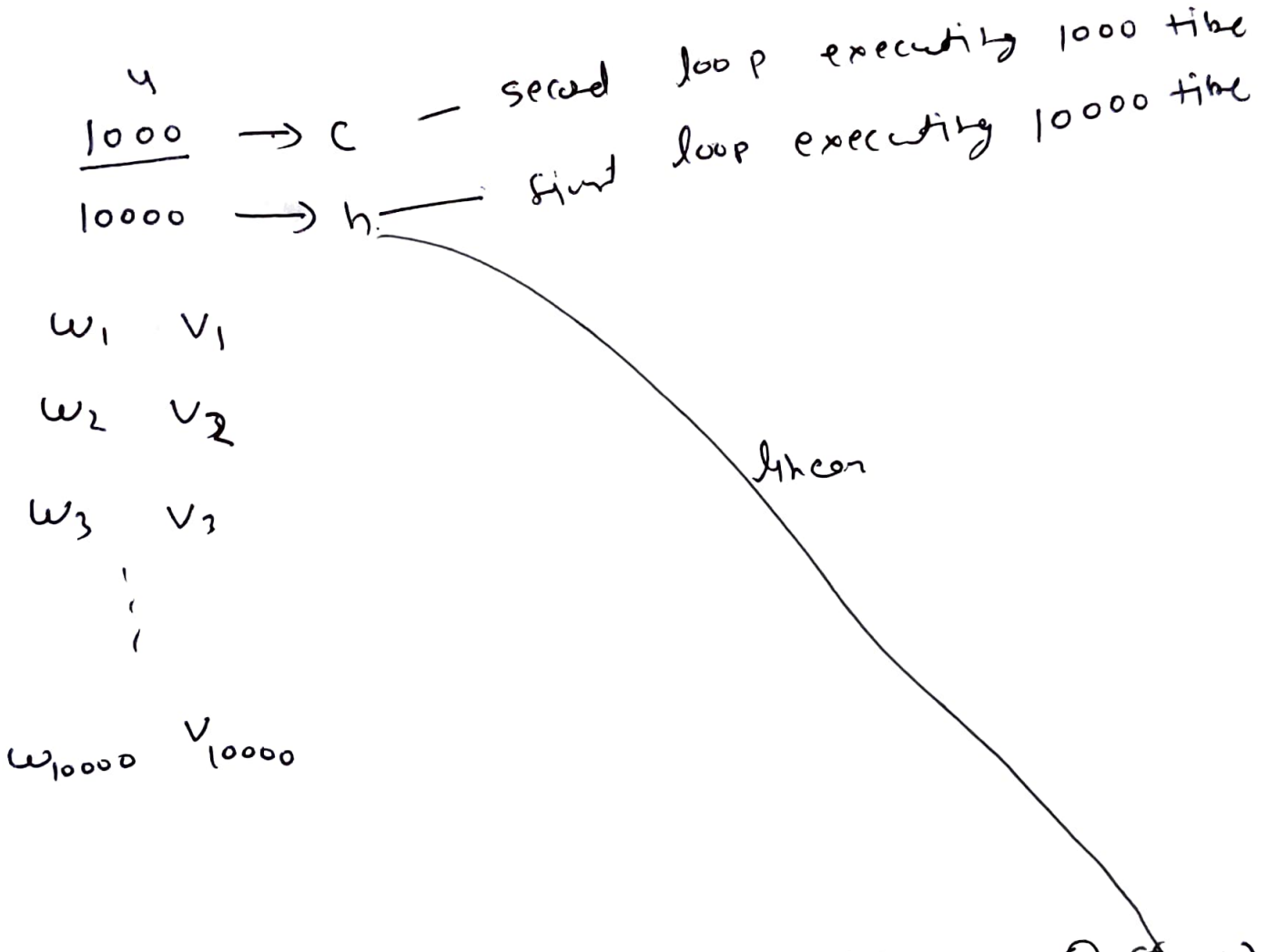
If item does not fit, take the previous solution

else if it fits (we take or not take) takes the larger value.

So 0/1 knapsack it uses implicit inductive proof.

Greedy	Vs	D.P
Big to small		small to big

Is the 0/1 knapsack algorithm executed in polynomial time? NO



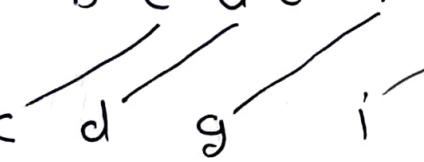
value of C	input size	loop
100	3	100
1000	4	1000
10000	5	10000

So based on input size and number of operation performed, it is not polynomial time operation.

longest common subsequence

String a b c d e f g h i j

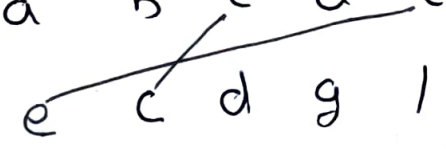
S2 c d g i



No need to find exact match. we have to write set of characters on coming in sequence.. so c d g i is L.C.S (Not required in continious).

a b c d e f g h i j

e c d g i



so (e c d g i) is set L.C.S

e g i (Yes)

c d g i (Yes)

Recursive

```
int LCS (i, j)
{ if ( A[i] == '\0' || B[j] == '\0')
      return 0;
  else if ( A[i] == B[j]
      return 1 + LCS (i+1, j+1)
  else
      return ( LCS (i+1, j), LCS (i, j+1));
```

A = | b | d | \0 |

B = | a | b | c | d | \0 |