Longest Common Subsequence

- **Definition:** The ***longest common subsequence*** **or** ***LCS*** of two strings *S1* and *S2* is the longest subsequence common between two strings.

S1 :	***A***	--	***A***	***T***	--	*G*	*G*	*C*	***C***	--	*A*	***T***	***A***	***n=10***
S2:	*A*	*T*	*A*	*T*	*A*	*A*	*T*	*T*	*C*	*T*	*A*	*T*	--	***m=12***

The *LCS* is *AATCAT*. The length of the *LCS* is 6.
The solution is not unique for all pair of strings. Consider the pair (*ATTA*, *ATAT*). The solutions are *ATT*, *ATA*. In general, for arbitrary pair of strings, there may exist many solutions.

LCS Theorem

- The *LCS* can be found by dynamic programming formulation. One can easily show:
  - **Theorem:** With a score of 1 for each match and a zero for each mismatch or space , the matched characters in an alignment of maximum value for a *LCS*.
- Since it is using the general dynamic programming algorithm its complexity is *O(nm)* .
- A longest substring problem, on the other hand has a *O(n+m)* solution. Subsequences are much more complex than substrings.
- Can we do better for the LCS problem? We will see ...

S1 :	***A***	--	***A***	***T***	--	*G*	*G*	*C*	***C***	--	*A*	***T***	***A***		***n=10***
S2:	*A*	*T*	*A*	*T*	*A*	*A*	*T*	*T*	*C*	*T*	*A*	*T*	--		***m=12***

- The optimal alignment is shown above. Note the alignment shows three insert (dark), one delete green) and three substitution or replacement operations (blue), which gives an edit distance of 7.
- But, the 3 replacement operations can be realized by 3 insert and 3 delete operations because a replacement is equivalent to first delete the character and then insert a character in its place like:

G	**--**	**G**	**--**	**C**	**--**
--	**A**	**--**	**T**	**--**	**T**