# Question 1:

**The instructor describes a method of building a Suffix Trie. What time complexity is it?**

#### Select the correct choice:

O(N) - Suffix trie construction is always linear

O(N^2) - We're adding N suffixes with max length N

O(Log N) - Suffix trie is a tree. Tree insert is Log N

O(2^N) - It takes 2^N time to generate all suffixes and then insert them

# Answer 1:

O(N^2) – We are adding N suffixes with max length N.

# Question 2:

**Once a Suffix Trie is built, what is the time complexity of looking up a pattern?**

#### Select the correct choice:

O(Height of Trie)

O(Length of Pattern)

O(Total # of nodes in the Trie)

# Answer 2:

O(Length of Pattern)

# Question 3:

**How many nodes in a Suffix Trie made out of a string of length N?**

#### Select the correct choice:

O(2^N) - Suffix Trie is a Tree and # of nodes in a tree is 2^N

O(N) - each letter corresponds to one node

O(NxN) - N suffixes, with max length of N and each node is added to the tree

O(N Log N) - Suffix Trie is a tree and # of nodes in a tree is N Log N

# Answer 3:

O(NxN) – N suffixes, with max length of N and each node is added to the tree.

# Question 4:

**Which of the following are true, for solving the Longest Common Substring problem? (Given two strings of length M and N)**

#### Select the correct choices:

Brute Force takes O(M^3 x N), where you'd first find all substrings of M [O(M^2)] and check if they exist in N (MxN).

Brute Force can be optimized by way of using KMP and reducing the complexity down to O(M^2 x (M+N)),

If we are open to using space O(MxN), then we can use DP to solve it in O(MxN) time.

After we construct a Suffix Trie, then it can be solved in linear time O(M + N), because all it takes is a DFS over the combined tree.

If we use hash tables and hash all substrings of both strings, then we can reduce the time complexity to constant time, because hashtables are always constant

# Answer 4:

Brute Force takes O(M^3 \* N), where you’d first find all substring of M [O(m^2)] and check if they exist in N (M x N)

Brute Force can be optimized by way of using KMP and reducing the complexity down to O(M^2 x (M + N))

If we are open to using space O(MxN), then we can use DP to solve it in O(MxN) time.

After we construct a Suffix Trie, then it can be solved in linear time O(M + N), because all it takes is a DFS over the combined tree.

# Question 5

**Which of the problems below can be solved efficiently by first building a Suffix Trie (Given two strings X and Y of length M and N respectively)**

#### Select the correct choices:

Find the Longest Common Substring between X and Y

Find the Longest Repeating Substring between X and Y

Find the Longest Palindromic Substring between X and Y

Find Substrings that are Common to X and Y

Find Substrings that are Not Common to X and Y

Find Shortest path between first occurrence of a character in X and last occurrence of a character in Y

Find Longest path between first occurrence of a character in X and last occurrence of a character in Y

# Answer 5:

Find the longest common substring between X and Y.

Find longest repeating substring between X and Y.

Find the longest Palindromic Substring between X and Y.

Find Substrings that are common to X and Y.

Find Substrings that are not common to X and Y.