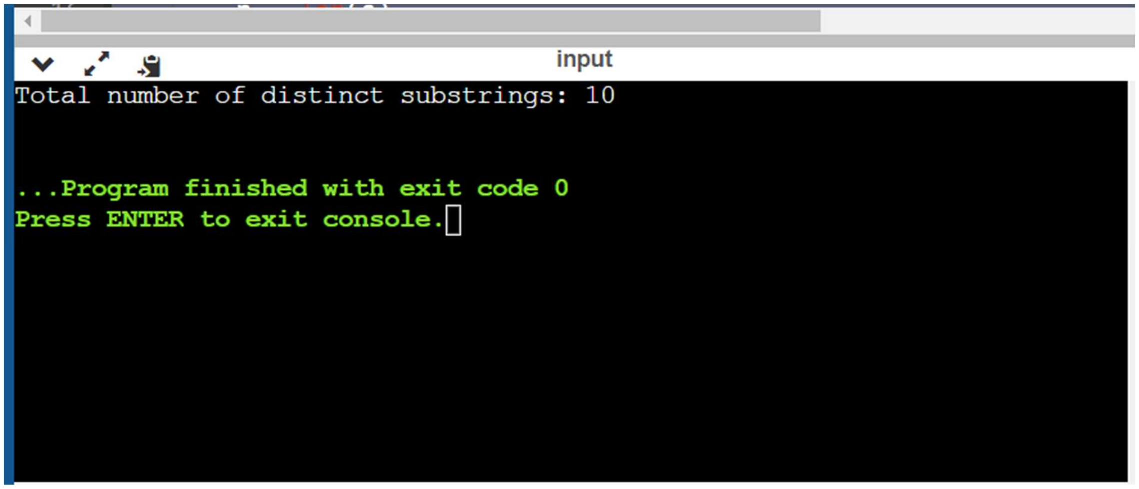
Problem 3 –Given a string of alphabetic characters. Return the count of distinct substrings of the string(including the empty string) using the Trie data structure.

```
class TrieNode:
    def __init__(self):
        self.children = {}
        self.is_end_of_word = False


def insert(root, word):
    node = root
    for char in word:
        if char not in node.children:
            node.children[char] = TrieNode()
        node = node.children[char]
    node.is_end_of_word = True


def count_distinct_substrings(s):
    def get_all_substrings(s):
        n = len(s)
        return [s[i:j+1] for i in range(n) for j in range(i, n)]


    root = TrieNode()
    substrings = get_all_substrings(s)


    for substring in substrings:
        insert(root, substring)


    def count_nodes(node):
        count = 1
        for child in node.children.values():
            count += count_nodes(child)
        return count

    return count_nodes(root)

S1 = "ababa"

result = count_distinct_substrings(S1)

print("Total number of distinct substrings:", result)
```



```
Total number of distinct substrings: 10


...Program finished with exit code 0
Press ENTER to exit console.
```