Batch: SY-IT(B3) Experiment Number:6

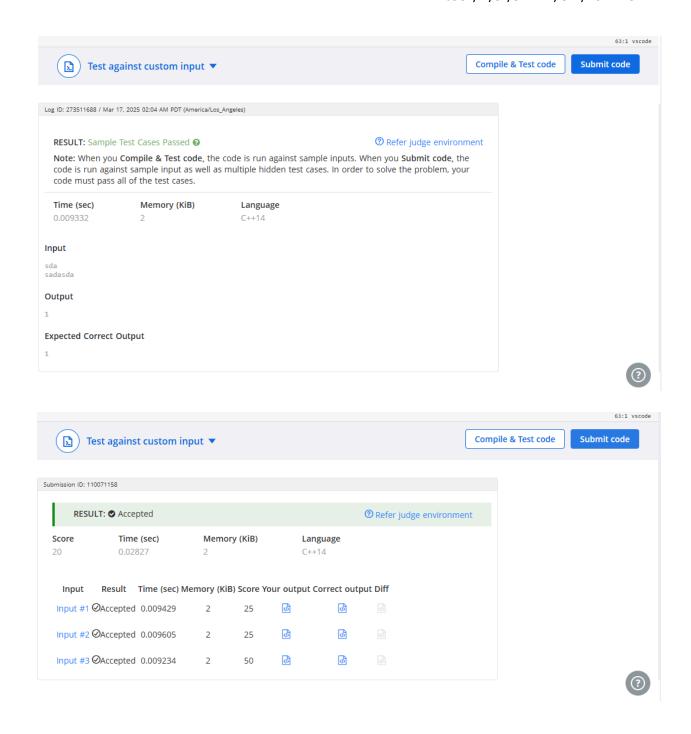
Roll Number: 16010423076 Name: Ritesh Jha

Aim of the Experiment: To study implement KMP Algorithm

```
Program/ Steps:
#include <iostream>
#include <vector>
using namespace std;
// Function to compute the LPS (Longest Prefix Suffix) array
vector<int> computeLPS(string P) {
  int m = P.length();
  vector\leqint\geq lps(m, 0);
  int j = 0;
  for (int i = 1; i < m; ) {
     if(P[i] == P[j]) \{
       lps[i] = j + 1;
       j++;
       i++;
     } else {
       if (j != 0) {
          j = lps[j - 1];
       } else {
          lps[i] = 0;
          i++;
  return lps;
// Function to count occurrences of P in T using KMP algorithm
int countOccurrences(string P, string T) {
  int n = T.length();
  int m = P.length();
```

```
if (m > n) return 0;
  vector<int> lps = computeLPS(P);
  int i = 0, j = 0, count = 0;
  while (i \le n) {
    if(T[i] == P[j]) \{
       i++;
       j++;
       if (j == m) {
          count++;
          j = lps[j - 1];
       }
     } else {
       if (j != 0) {
          j = lps[j - 1];
       } else {
          i++;
  return count;
int main() {
  string P, T;
  cin >> P >> T;
  cout << countOccurrences(P, T) << endl;</pre>
  return 0;
}
```

Output/Result:



Outcomes:

CO4. Learn effective computation and programming practices for numeric and string operations and computation geometry

Conclusion (based on the Results and outcomes achieved):

From this experiment, I learned how the Knuth-Morris-Pratt (KMP) algorithm efficiently finds occurrences of a pattern in a text using the LPS (Longest Prefix Suffix) array to avoid unnecessary comparisons. I understood how the algorithm improves string searching by reducing time complexity to O(n + m), making it faster than brute force methods. Implementing and testing the code helped me see how theoretical concepts apply in real-world programming. This experiment also improved my understanding of string operations and their role in competitive programming and algorithm design.

References:

- 1. https://www.hackerearth.com/practice/algorithms/string-algorithm/string-searching/tutorial/
- 2. T.H. Coreman ,C.E. Leiserson,R.L. Rivest, and C. Stein, "Introduction to algorithms", 3rd Edition 2009, Prentice Hall India Publication
- 3. Antti Laaksonen, "Guide to Competitive Programming", Springer, 2018
- 4. Gayle Laakmann McDowell," Cracking the Coding Interview", CareerCup LLC, 2015
- 5. Steven S. Skiena Miguel A. Revilla,"Programming challenges, The Programming Contest Training Manual", Springer, 2006
- 6. Antti Laaksonen, "Competitive Programmer's Handbook", Hand book, 2018
- 7. Steven Halim and Felix Halim, "Competitive Programming 3: The Lower Bounds of Programming Contests", Handbook for ACM ICPC