Name: Shaan Agarwal
UID: 2021300001
Title: Design and Analysis Of Algorithms Lab
Experiment No.: 9

Aim: To implement string matching algorithms using Rabin Karp.

Theory:

Rabin-Karp algorithm is an algorithm used for searching/matching patterns in the text using a hash function. Unlike Naive string matching algorithm, it does not travel through every character in the initial phase rather it filters the characters that do not match and then performs the comparison.

Algorithm:

```
n = t.length
m = p.length
h = dm-1 mod q
p = 0
t0 = 0
for i = 1 to m
    p = (dp + p[i]) mod q
    t0 = (dt0 + t[i]) mod q
for s = 0 to n - m
    if p = ts
        if p[1.....m] = t[s + 1..... s + m]
            print "pattern found at position" s
    If s < n-m
```

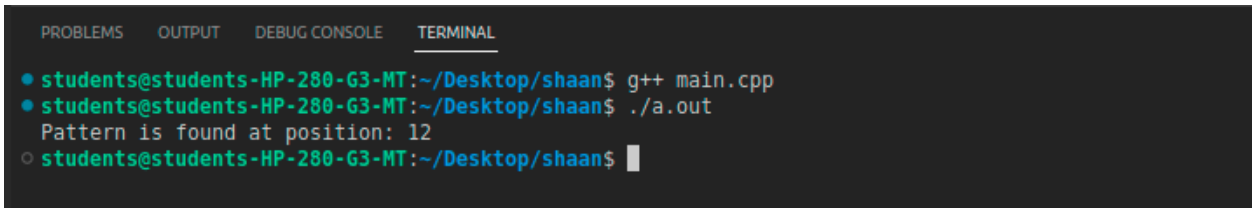
$$ts + 1 = (d (ts - t[s + 1]h) + t[s + m + 1]) \bmod q$$

Program:

```
#include <bits/stdc++.h>
using namespace std;

#define d 10

void rabinKarp(char pattern[], char text[], int q) {
 int m = strlen(pattern);
 int n = strlen(text);
 int i, j;
 int p = 0;
 int t = 0;
 int h = 1;

 for (i = 0; i < m - 1; i++) {
   h = (h * d) % q;
 }

 // Calculating hash value for pattern and text
 for (i = 0; i < m; i++) {
   p = (d * p + pattern[i]) % q;
   t = (d * t + text[i]) % q;
 }

 // Finding the match
 for (i = 0; i <= n - m; i++) {
   if (p == t) {
     for (j = 0; j < m; j++) {
       if (text[i + j] != pattern[j]) break;
     }

     if (j == m) cout << "Pattern is found at position: " << i + 1 <<
endl;
   }
```

```
    if (i < n - m) {
      t = (d * (t - text[i] * h) + text[i + m]) % q;

      if (t < 0) t = (t + q);
    }
  }
}

int main() {
  char text[] = "MY NAME IS SHAAN";
  char pattern[] = "SHAAN";
  int q = 13;
  rabinKarp(pattern, text, q);
}
```

Output:



Conclusion: After performing the above experiment, I have understood the implementation of String Matching, and its implementation using the Rabin Karp Algorithm.