**EXPERIMENT N0-2**

AIM: Implementation of Rabin Karp algorithm.

**CODE:**

#include<stdio.h>

#include<string.h>

void search(char pat[], char txt[], int q)

{

int M = strlen(pat);

int N = strlen(txt);

int i, j;

int p = 0;

int t = 0;

int h = 1;

int d=256;

for (i = 0; i < M-1; i++)

h = (h\*d)%q;

for (i = 0; i < M; i++)

{

p = (d\*p + pat[i])%q;

t = (d\*t + txt[i])%q;

}

for (i = 0; i <= N - M; i++)

{

if ( p == t )

{

for (j = 0; j < M; j++)

{

if (txt[i+j] != pat[j])

break;

}

if (j == M)

printf("Pattern found at index %d \n", i);

}

if ( i < N-M )

{

t = (d\*(t - txt[i]\*h) + txt[i+M])%q;

if (t < 0)

t = (t + q);

}

}

}

int main()

{ char txt[80],pat[80];

int q;

printf("Enter some text \n");

gets(txt);

printf("Enter a pattern to be searched \n");

gets(pat);

printf("Enter a prime number \n");

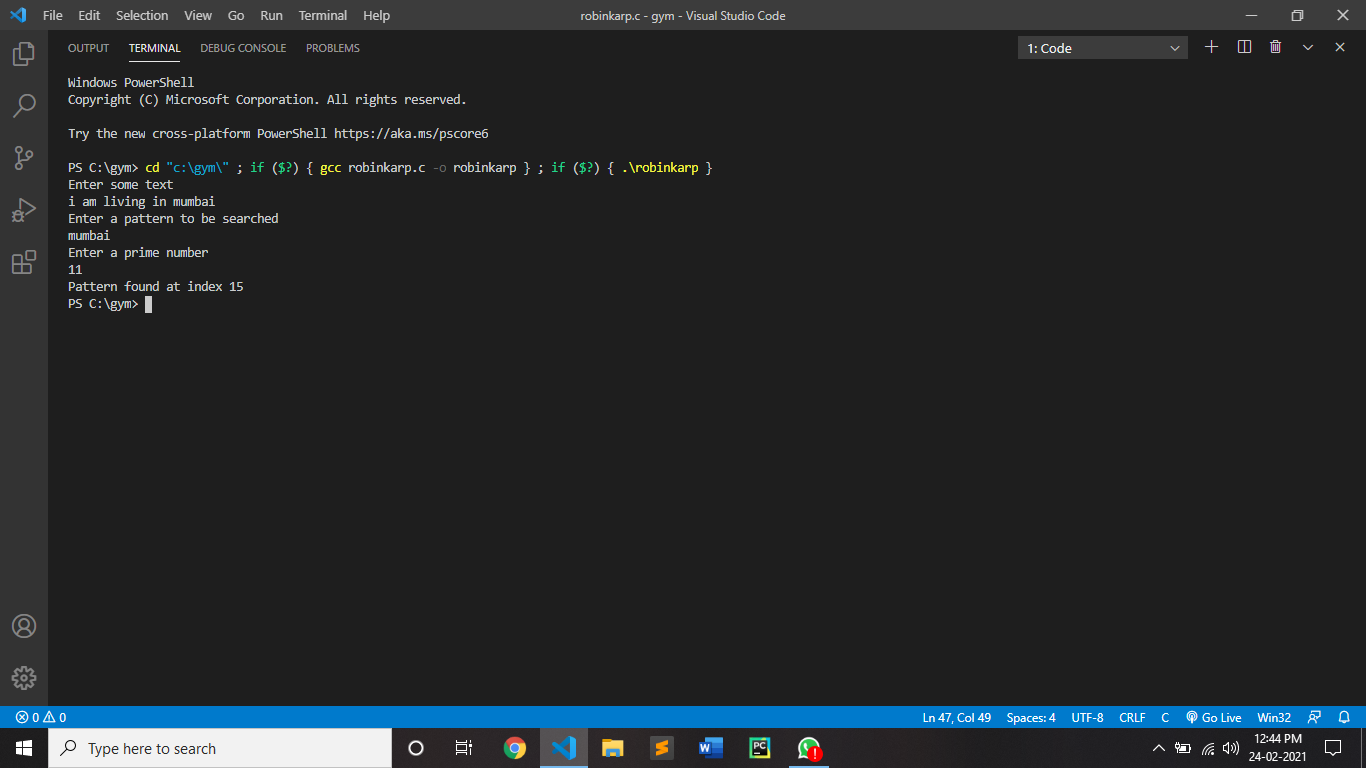
scanf("%d",&q);

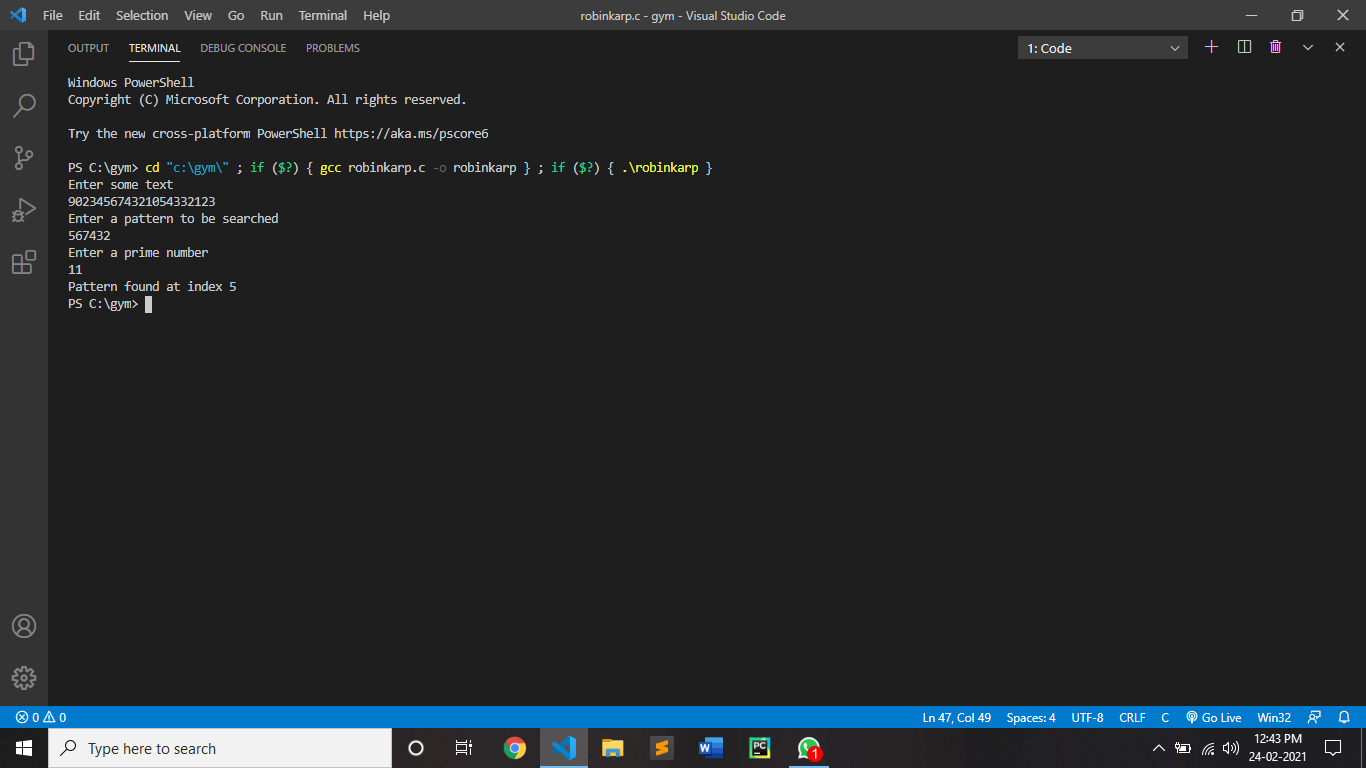
search(pat, txt, q);

return 0;

}

**OUTPUT:**





**Conclusion:**

By performing above algorithm we can concluded that,The normal and best-case running season of the Rabin-Karp calculation is O(n+m), however its most pessimistic scenario time is O(nm). Worst case of Rabin-Karp calculation happens when all characters of example and text are same as the hash estimations of the multitude of substrings of txt[] coordinate with hash estimation of pat[]. For instance pat[] = "AAA" and txt[] = "AAAAAAA".