**Experiment-9**

**To WAP and analyze to find all occurrences of a pattern P in a given string S**

**using Rabin Karp algorithm.**

1. **Define the problem.**

Ans . We have give a main string we have to give an another to check whether how many times the given string by us has occur inside the main string and also count the position and occurrence of it in the main string using Rabin Karp algorithm .

1. **Write pseudo code for Rabin Karp Algorithm for string matching and find Time and Space complexity.**

**RABIN-KARP-MATCHER (T, P, d, q)**

1. n ← length [T]

2. m ← length [P]

3. h ← dm-1 mod q

4. p ← 0

5. t0 ← 0

6.counter-0

7. for i ← 1 to m

8. do p ← (dp + P[i]) mod q

9. t0 ← (dt0+T [i]) mod q

10. for s ← 0 to n-m

11. do if p = ts

12. then if P [1.....m] = T [s+1.....s + m]

13. then "Pattern occurs with shift" s

14.counter =counter+1

15. If s < n-m

16. then ts+1 ← (d (ts -T [s+1]h)+T [s+m+1])mod q

17.exit

**Time complexity:** O(((n-m)+1)n)

**space complexity:** O((m)n)

1. **Discuss the best and worst case with example.**

Let hash codes for these alphabets are as follows

A-1, B-2, C-3, D-4, E -5, F-6, G-7

**Best case :**  Main String : AABAACCB

Match String : AAB

**Worst case :** Main String : CCACCAAEDBA

Match String : DBA

**Time in worst Case- o**(nm)

1. **Write Source Code in C/C++.**

Code

#include <iostream>

#include<string.h>

#define d 256

using namespace std;

// main -txt match-pat

//h -hash p-hashpat t-hashtext

void search(char match[], char main[], int q) {

int M = strlen(match);// length of second string

cout<<"match length:"<<M;

int N = strlen(main);// length of main string

cout<<"\n main length:"<<N<<endl;

int hash = 1;

int hashpat = 0; // hash value for pattern (second string)

int hashtext = 0; // hash value for txt (main string)

int i=0;

int j=0;

int counter=0;

// h=d^(m-1) h= d to the power of m-1

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

// calclating hash value

//d is macro defined above

hash = (hash \* d) % q;

//here q to decrease hashvalue

}

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

{

// hash value for both text

hashpat = (d \* hashpat + match[i]) % q;

//p=(256\* 0 + ascci value of h()))%101

hashtext = (d \*hashtext + main[i]) % q;

}

//now we going to compare hash values again and again

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

{

// check for characters one by one

if (hashpat == hashtext )

{

/\* Check for characters one by one \*/

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

{

if (main[i+j] != match[j])

break;

}

if (j == M) {

counter =counter+1;

cout<<"Pattern found at index "<< i<<endl;

}

}

//again calculating hash value for sliding after no match from previuos

if ( i < N-M )

{

hashtext = (d\*(hashtext- main[i]\*hash) + main[i+M])%q;

// We might get negative value of t, converting it

// to positive

if (hashtext < 0)

hashtext = (hashtext + q);

}

}

cout<<match<<"occurences :"<<counter;

}

// main code

int main()

{

char main[1000];

char match[1000];

cout<<"enter the main string :";

gets(main);

cout<<"enter the second string :";

gets(match);

int l= 101; // A prime number to decrese the value of hashvalue in hash function

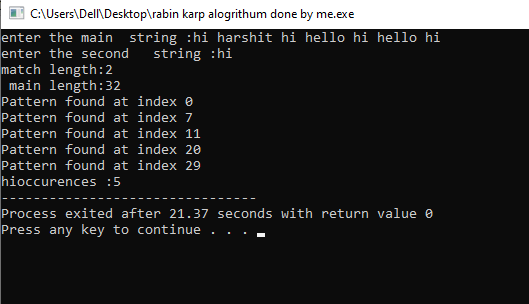
// two check them are they equal or not

search(match, main, l);

return 0;

}

**Output**

****

1. Variables and other data structure.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variable Name** | **Datatype** | **Typical Value** | **Minimum Value** | **Maximum Value** |
| main[ ] | Char | Hi Harshit hi hello hi hello hi | null | - |
| match[ ] | Char | hi | null | - |
| N | Int | 32 | 0 | - |
| M | Int | 2 | 0 | - |
| Counter | INT | 5 | 0 | N |
| I | INT | - | 0 | N-M |
| J | INT | \_ | 0 | M |
| h | INT | \_ | 1 | - |
|  |  |  |  |  |
|  |  |  |  |  |

1. Test Plan.

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Comments** |
| Input1=hi harshit hi  hello hi hello hi,  Input 2= hi | enter the main string :hi harshit hi  hello hi hello hi  enter the second  string :hi  match length:2  main length:32  Pattern found at index 0  Pattern found at index 7  Pattern found at index 11  Pattern found at index 20  Pattern found at index 29  hioccurences :5 | enter the main string :hi harshit hi  hello hi hello hi  enter the second  string :hi  match length:2  main length:32  Pattern found at index 0  Pattern found at index 7  Pattern found at index 11  Pattern found at index 20  Pattern found at index 29  hioccurences :5 | NA |