**Experiment-8**

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

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

**Eg .**

**2.Write pseudo code for Naive Algorithm for string matching and find Time and Space complexity.**

1. Enter the main string:";

gets(txt);

2.Enter the maching string:"

gets(pat);

1. Search(pat ,txt)
2. N ---- length[ pat ]
3. M --- length[txt]
4. For i to N-M:
5. Int j
6. For j to M:
7. if ( txt[i + j] != pat[j] ) :

break

1. if (j == M) :

count=count+1

Print :"Pattern found at index : " i.

1. Print : pat “ ocurrences : ”count
2. Exit

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

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

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

**Best case** : Pattern string : **ABAABBCAAD** time=o(1)

Maching patteren **:ABA**

**Worst case :** Pattern string : **ABAABBCAAD** time=o((m)n)

Maching patteren **:AAD**

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

Code:

#include <iostream>

using namespace std;

int search(char\* pat, char\* txt)

{

int count=0,index[100];

int M = strlen(pat);

int N = strlen(txt);

/\* A loop to slide pat[] one by one \*/

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

int j;

/\* For current index i, check for pattern match \*/

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

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

break;

}

if (j == M) {// if pat[0...M-1] = txt[i, i+1, ...i+M-1]

count=count+1;

cout << "Pattern found at index "

<< i << endl;

}

}

return count;

}

int main()

{

char txt[100] ;

char pat[100];

cout<<"Enter the main string:";

gets(txt);

cout<<"Enter the maching string:";

gets(pat);

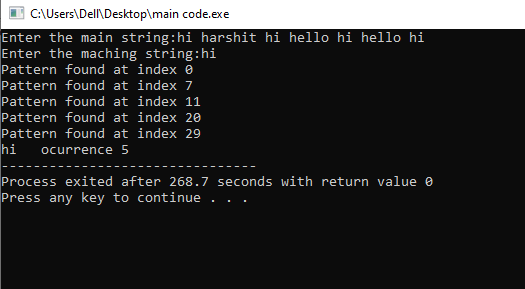
int z=search(pat, txt);

cout<<pat<<" ocurrence "<<z;

return 0;

}

**Output:**

****

5.Variables and other data structure.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variable Name** | **Datatype** | **Typical Value** | **Minimum Value** | **Maximum Value** |
| Pat[ ] | Char | ABAABBCAAD | N =null | ABAABBCAAD (N) |
| txt [ ] | Char | ABA , AAD |  |  |
| N | Int | 11 | 0 | - |
| M | Int | 3 | 0 | - |
| count | INT | - | 0 | N |
| I | INT | - | 0 | N-M |
| J | INT | \_ | 0 | M |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

1. Test Plan.

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Comments** |
| Input1=hi Harshit hi hello hi hello hi,  Input 2= hi | 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  hi ocurrence 5 | 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  hi ocurrence 5 | NA |