

Searching for Patterns | Set 4 (A Naive Pattern Searching Question)

Question: We have discussed Naive String matching algorithm [here](#). Consider a situation where all characters of pattern are different. Can we modify [the original Naive String Matching algorithm](#) so that it works better for these types of patterns. If we can, then what are the changes to original algorithm?

Solution: In the [original Naive String matching algorithm](#), we always slide the pattern by 1. When all characters of pattern are different, we can slide the pattern by more than 1. Let us see how can we do this. When a mismatch occurs after j matches, we know that the first character of pattern will not match the j matched characters because all characters of pattern are different. So we can always slide the pattern by j without missing any valid shifts. Following is the modified code that is optimized for the special patterns.

```
#include<stdio.h>
#include<string.h>

/* A modified Naive Pattern Searching algorithm that is optimized
   for the cases when all characters of pattern are different */
void search(char *pat, char *txt)
{
    int M = strlen(pat);
    int N = strlen(txt);
    int i = 0;

    while(i <= N - M)
    {
        int j;

        /* For current index i, check for pattern match */
        for (j = 0; j < M; j++)
```

Google™ Custom Search



GeeksforGeeks



52,731 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

```

{
    if (txt[i+j] != pat[j])
        break;
}
if (j == M) // if pat[0...M-1] = txt[i, i+1, ...i+M-1]
{
    printf("Pattern found at index %d \n", i);
    i = i + M;
}
else if (j == 0)
{
    i = i + 1;
}
else
{
    i = i + j; // slide the pattern by j
}
}

/* Driver program to test above function */
int main()
{
    char *txt = "ABCEABCDABCEABCD";
    char *pat = "ABCD";
    search(pat, txt);
    getchar();
    return 0;
}

```

Output:

Pattern found at index 4

Pattern found at index 12

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



Popular Posts

[All permutations of a given string](#)

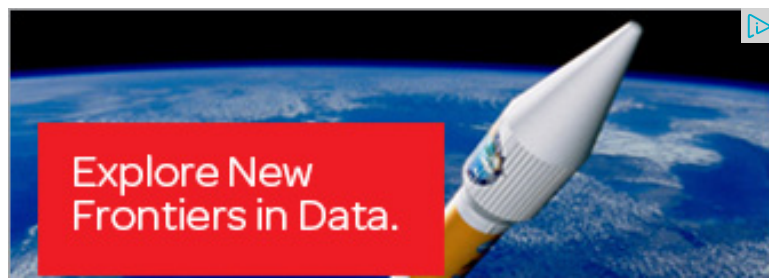
[Memory Layout of C Programs](#)

[Understanding "extern" keyword in C](#)

[Median of two sorted arrays](#)

[Tree traversal without recursion and without stack!](#)

[Structure Member Alignment, Padding and Data Packing](#)



Intersection point of two Linked Lists


Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST



HPCC Systems is Big Data Processing and Analytics
Open Source. Proven. Trusted.

 LexisNexis® [Learn More](#) 

Related Tpoics:

- [Printing Longest Common Subsequence](#)
- [Suffix Array | Set 2 \(nLogn Algorithm\)](#)
- [Rearrange a string so that all same characters become d distance away](#)
- [Recursively remove all adjacent duplicates](#)
- [Find the first non-repeating character from a stream of characters](#)
- [Dynamic Programming | Set 33 \(Find if a string is interleaved of two other strings\)](#)
- [Remove "b" and "ac" from a given string](#)
- [Dynamic Programming | Set 29 \(Longest Common Substring\)](#)



0

 Tweet

0



1

Writing code in comment? Please use [ideone.com](#) and share the link here.

18 Comments

GeeksforGeeks

Sort by Newest ▼



Join the discussion...



sijayaraman · 3 days ago

`#include<iostream>`

Custom market
research at scale.

Get \$75 off

 Google consumer surveys

```
#include<string>
using namespace std;
int main()
{
char str[]="ABCEABCDABCEABCD";
char pat[]="ABCD";
for(int i=0;i<strlen(str);i++) {="" int="" j,k;="" for(j="0,k=i;j<strlen(pat);j++)" {="
}="" if(j=="strlen(pat))" {="" cout<<"startIndex="&lt;&lt;i&lt;&lt;endl;" and="" endIndex
```

^ | v • Reply • Share ›



zyzz • 10 months ago

```
/#include<stdio.h>
#include<string.h>
```

```
void pattern(char *s,char *p){
    int n=strlen(p);
    int i,flag=0;
    while(*p!=*s)
    {s++;}

    for(i=0;i<n;i++){

        if(*p==*s){
```

695



Subscribe

Recent Comments

affizerv Your example has two 4s on row 3, that's why it...

[Backtracking | Set 7 \(Sudoku\)](#) · 18 minutes ago

RVM Can someone please elaborate this Qs from above...

[Flipkart Interview | Set 6](#) · 38 minutes ago

Vishal Gupta I talked about as an Interviewer in general,...

[Software Engineering Lab, Samsung Interview | Set 2](#) · 38 minutes ago

@meya Working solution for question 2 of 4f2f round....

[Amazon Interview | Set 53 \(For SDE-1\)](#) · 1 hour ago

sandeep void rearrange(struct node *head) {...

Given a linked list, reverse alternate nodes and append at the end · 2 hours ago

Neha I think that is what it should return as, in...

[Find depth of the deepest odd level leaf node](#) · 2 hours ago

```
p++;  
s++;  
  
flag++;
```

[see more](#)

^ | v • Reply • Share ›



seeker • 2 years ago

it can easily be done using following simple loop

```
void search(char *pat, char *txt)  
{  
    int M = strlen(pat);  
    int N = strlen(txt);  
    int i = 0; //points to location in text  
    int j = 0; //points to location for pattern  
    for( int i = 0 ; i <= N-M ;i++)  
    {  
        if( txt[i] == pat[j] ) {  
            j++;  
        }else {  
            j=0;  
        }  
        if( j == M ) {  
            printf("Pattern found at index %d \n", i-M);  
            j=0;  
        }  
    }  
}
```

AdChoices

► [Algorithm Design](#)

► [Pattern Matching](#)

► [Java Pattern](#)

AdChoices

► [Design Pattern](#)

► [Algorithm Java](#)

► [C Algorithm](#)

AdChoices

► [Pattern Code](#)

► [Repeated Pattern](#)

► [Pattern Search](#)



saurabh · 2 years ago

Above algorithm not work for

```
char *txt = "AgneepathAgneepathkanchaAgneepathKancha";
char *pat = "Kancha";
```

^ | v · Reply · Share ›



GeeksforGeeks → saurabh · 2 years ago

@saurabh: Please take a closer look at the problem statement. The pattern as 'a' occurs two times. You can apply the algorithm discussed [here](#).

^ | v · Reply · Share ›



Naveen Makwana · 3 years ago

I think , i got a better solution to this ...accepting all types of patterns and yes works so here is the code.....

[sourcecode]

```
int main()
{
char *s="ARAARAJAAARAJAAJARRAARAJRARAJAA";
char *p="ARAJAA";
int i=0,j=0,ls,lp;
lp=strlen(p); // length of pattern
ls=strlen(s); // length of text
```

```
while(ls--){
if((s[i]==p[j])){
j++;
}
else{
j=0;
```

[see more](#)

^ | v • Reply • Share ›



Agniswar → Naveen Makwana • 3 years ago

@Naveen:Hi,your code gives wrong output in case of inputs like char *
*p="aaba".Accd to your code the output is "Pattern found at 0" and "Pa
missed out position 1..I guess it's because you have incremented i one
j.So,i guess you will need two loops in order to print all the positions !

^ | v • Reply • Share ›



Naveen Makwana → Naveen Makwana • 3 years ago

u can remove use of strlen here....like

[sourcecode]

```
int main()
{
char *s="ARAARAJAAARAJAAJARRAARAJRARAJAA";
char *p="ARAJAA";
int i=0,j=0;
while(s[i]){
if((s[i]==p[j])){
j++;
}
else{
j=0;
if((s[i]==p[j])){
j++;
}
}
if(p[j+1]!='&#092&#048'){
printf("Pattern found at %d\n",i-j+1);
```



```
}  
}  
i++;  
}  
return 0;  
}
```

^ | v • Reply • Share ›



student • 3 years ago

still the solution will not work for cases like this

```
char *txt = "AABCD";  
char *pat = "ABCD";
```

^ | v • Reply • Share ›



GeeksforGeeks → student • 3 years ago

@student: Thanks for pointing this out. We have changed the code to I

^ | v • Reply • Share ›



student → GeeksforGeeks • 3 years ago

thanks for correcting the error.

^ | v • Reply • Share ›



Raja • 3 years ago

Will it work for "RARAJA" if the pattern is "RAJA"

^ | v • Reply • Share ›



Sandeep → Raja • 3 years ago

@Raja: Please take a closer look at the question. The pattern "RAJA" i
question as all characters of the pattern must be different.

^ | v • Reply • Share ›



Venki · 3 years ago

The above function misses few corner cases. For example see the following i

```
char *txt = "AABAACAADAABAAAABA"
char *pat = "AABA"
```

There are three matching patterns. But the code prints only two.

Here is the correct version of program (or increment i by (M + 1) in original pro

```
#include <stdio.h>
#include <string.h>

#define TEXT "AABAACAADAABAAAABA"
#define PATT "AABA"

// Improved pattern matching
```

[see more](#)

^ | v · [Reply](#) · [Share](#) ›



Sandeep → Venki · 3 years ago

@venki: Please take a closer look at the question. The pattern "AABA" question as all characters of the pattern must be different.

^ | v · [Reply](#) · [Share](#) ›



shanker · 3 years ago

@geeksfoegeesk..can you post Boyce Moorrie string matching algo with expla

^ | v · [Reply](#) · [Share](#) ›



Simran · 3 years ago

This code is not right.. I feel you have modified the algorithm from Cormen a lil function, you cannot increment 'i' by 'j+1'.. It has to be incremented by 1 only..
Test Case where your code fails..

```
char *txt = "ABAABABACBCABCABABA";  
char *pat = "ABABAC";
```

^ | v · Reply · Share ›



Simran → Simran · 3 years ago

Sorry, didn't see your statement about pattern string having different ch

^ | v · Reply · Share ›



Subscribe



Add Disqus to your site