Searching for Patterns | Set 1 (Naive Pattern Searching)

Given a text *txt[0..n-1]* and a pattern *pat[0..m-1]*, write a function *search(char pat[], char txt[])* that prints all occurrences of *pat[]* in *txt[]*. You may assume that *n > m*.

Examples:
1) Input:

```
  txt[] =  "THIS IS A TEST TEXT"
  pat[] = "TEST"
```

Output:

```
Pattern found at index 10
```

2) Input:

```
  txt[] =  "AABAACAADAABAAABAA"
  pat[] = "AABA"
```

Output:

```
   Pattern found at index 0
   Pattern found at index 9
   Pattern found at index 13
```

Pattern searching is an important problem in computer science. When we do search for a string in notepad/word file or browser or database, pattern searching algorithms are used to show the search results.

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

Naive Pattern Searching:
Slide the pattern over text one by one and check for a match. If a match is found, then slides by 1 again to check for subsequent matches.

```
#include<stdio.h>
#include<string.h>
void search(char *pat, char *txt)
{
    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]
        {
           printf("Pattern found at index %d \n", i);
        }
    }
}

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

What is the best case?
The best case occurs when the first character of the pattern is not present in text at all.

```
txt[]  = "AABCCAADDEE"
pat[] = "FAA"
```

The number of comparisons in best case is O(n).

Geometric Algorithms

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

What is the worst case ?

The worst case of Naive Pattern Searching occurs in following scenarios.
1) When all characters of the text and pattern are same.

```
txt[] = "AAAAAAAAAAAAAAAAAA"
pat[] = "AAAAA".
```

2) Worst case also occurs when only the last character is different.

```
txt[] = "AAAAAAAAAAAAAAAAAB"
pat[] = "AAAAB"
```

Number of comparisons in worst case is O(m*(n-m+1)). Although strings which have repeated characters are not likely to appear in English text, they may well occur in other applications (for example, in binary texts). The KMP matching algorithm improves the worst case to O(n). We will be covering KMP in the next post. Also, we will be writing more posts to cover all pattern searching algorithms and data structures.

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

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)

1 Tweet 0 1

Writing code in comment? Please use **ideone.com** and share the link here.

14 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

rahul · 7 days ago

can anyone please tell me how can i implement this algo on cuda(Parallel proç

Reply · Share ›

nani · 3 months ago

what could be the average case complexity of this algorithm

Reply · Share ›

695 Subscribe

Recent Comments

affiszerv 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

pappu • 6 months ago

thanks for the code...

one doubt:: In the "WHAT IS THE WORST CASE?" section, how is the first ex

• Reply • Share ›

Dynamite pappu • 6 months ago

In the first example, you are doing comparison of (n-m+1) substrings ir
comparison continues for m characters in the pattern, hence it is the w
the pattern was BBBBB instead of AAAAA, in that case you would have
text in the very first comparison with the pattern, as compared to m co
the first example is worst case

• Reply • Share ›

manshi • 9 months ago

The given code has a flaw:
Try to find the pattern "ch" in string "aaaaaaach"

Output: pattern doesn't exist

Correction:
/* instead of N-M, allow the loop till end of string */
for (int i = 0; i <= N ; 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]

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

Please correct me if I am wrong

Reply • Share ›

Chirag Patel manshi • 8 months ago

it works k!!The original explained program works k with yr input!!

Reply • Share ›

Shiwakant Bharti manshi • 8 months ago

My adaptation of the given code in Java actually works. Please check y
i = N, j= 1, txt[i+j] will actually lead to ArrayIndexOutOfBoundsExceptio
C/C++.

Reply • Share ›

gautam • 10 months ago

[sourcecode language="JAVA"]

```
public HashSet<Integer> naivePatternSearch(String pattern, String string) {
HashSet<Integer> index = new HashSet<Integer>();
for (int j = 0; j < string.length(); j++) {
for (int i = 0; i < pattern.length(); i++) {
if (pattern.charAt(i) == string.charAt(j+i)) {
if(i==pattern.length()-1){
index.add(j);
}
} else {
break;
}
```

}
}
return index;
}

^ | ⌄ • Reply • Share ›

abhishek08aug • a year ago

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

void search_pattern(char * str, char * pattern) {
  int str_len=strlen(str);
  int pattern_len=strlen(pattern);
  int i, j;
  for(i=0; i<str_len-pattern_len; i++) {
    for(j=0; j<pattern_len; j++) {
      if(*(str+i+j)!=*(pattern+j)) {
        break;
      }
    }
    if(j==pattern_len) {
      printf("Pattern found at index: %d\n", i);
    }
  }
}
```

see more

^ | ⌄ • Reply • Share ›

meap4aa • a year ago

A similar Approach through Recursion:

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

int my_cmp(char* a,char* b,int i,int n);

int my_cmp(char* a,char* b,int i,int n)
{
                if(*(a+i)==*(b+(strlen(b)-n)))
                {
                        //printf("\nChecking i = %d ",i);
                        //printf("\nNow n = %d ",n);
                if(n==1)
                return 1;
                my_cmp(a,b,++i,--n);
                }
                else
```

see more

1 • Reply • Share ›

alien • a year ago

1 more approach could be as below:

bool isSubstring(char* src, char* pattern) {
int i=0,j=0, flag=0;
int lenp, lens;
for(i=0;*src+!='';i++);
lens = --i;
for(i=0;*pattern+i!='';i++);
lenp = --i;
i=0;

while((*(src+i)) ! = '')
{

if(*(src+i) == *(pattern+j))
{
j++;
flag++;
if(flag == lenp)

see more

^ | v • Reply • Share ›

Yatendra Goel • a year ago

The above algo is very similar to previous "Naive Pattern Search" algo but as t
previous one uses 'for' loop, so novice programmers might have to spend few
same algo as previous EXCEPT FEW LINES.

So I have written the above algo again using 'for' loop which is same as previo
Search" algo) except two lines (added comment on those two lines) so that it'
between two.

```
private void printPatternIndices(char[] text, char[] pattern) {

                for (int i = 0; i < text.length - pattern.length + 1;

                        int j;
                        for (j = 0; j < pattern.length; j++) {
                                if (text[i + j] != pattern[j]) {
                                        break;
                                }
                        }
```

see more

Reply • Share ›

raman • 3 years ago

@geeksforgeeks plz post KMP, rabin karp string searching algorithm

ASAP , i am looking forward ..Plese Keep in Posting Such

Reply • Share ›

Vinay raman • 2 months ago

1,1,2,3,4,5,6,2,2,2,5,3,3,2,2,1,5,5,5,5,4,4,4,1,2,2,2,2,6,6,2,2,1,1,2,2

Can anyone help me to find out how many times a sequence number (
times 2,2,2,2) are repeated in any programming language?

Reply • Share ›

Subscribe   Add Disqus to your site