GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- Home
- Algorithms
- DS
- GATE
- Interview Corner
- Q&A
- C
- C++
- Java
- Books
- Contribute
- Ask a Q
- About

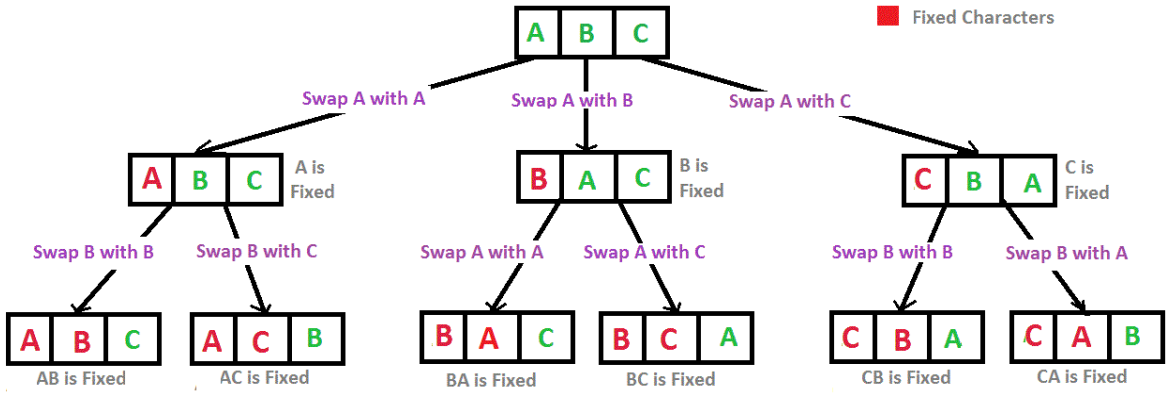
Array
Bit Magic
C/C++
Articles
GFacts
Linked List
MCQ
Misc
Output
String
Tree
Graph

Write a C program to print all permutations of a given string

A permutation, also called an “arrangement number” or “order,” is a rearrangement of the elements of an ordered list S into a one-to-one correspondence with S itself. A string of length n has n! permutation.
Source: Mathword(http://mathworld.wolfram.com/Permutation.html)

Below are the permutations of string ABC.
ABC, ACB, BAC, BCA, CAB, CBA

Here is a solution using backtracking.



Recursion Tree for Permutations of String "ABC"

```
# include <stdio.h>

/* Function to swap values at two pointers */
void swap (char *x, char *y)
{
    char temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

/* Function to print permutations of string
   This function takes three parameters:
   1. String
   2. Starting index of the string
   3. Ending index of the string. */
void permute(char *a, int i, int n)
{
   int j;
   if (i == n)
     printf("%s\n", a);
   else
   {
        for (j = i; j <= n; j++)
       {
          swap((a+i), (a+j));
          permute(a, i+1, n);
          swap((a+i), (a+j)); //backtrack
       }
   }
}

/* Driver program to test above functions */
int main()
{
   char a[] = "ABC";
   permute(a, 0, 2);
   getchar();
   return 0;
}
```

Output:

```
ABC
```

ACB
BAC
BCA
CBA
CAB

Algorithm Paradigm: Backtracking
Time Complexity: O(n*n!)

Please write comments if you find the above codes/algorithms incorrect, or find other ways to solve the same problem.

Related Topics:

- Recursively print all sentences that can be formed from list of word lists
- Check if a given sequence of moves for a robot is circular or not
- Find the longest substring with k unique characters in a given string
- Function to find Number of customers who could not get a computer
- Find maximum depth of nested parenthesis in a string
- Find all distinct palindromic sub-strings of a given string
- Find if a given string can be represented from a substring by iterating the substring “n” times
- Suffix Tree Application 6 – Longest Palindromic Substring

Tags: Backtracking, MathematicalAlgo

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

Vishal · 2 hours ago

```
class Permutaiton {

static main(def args){

def s="ABC"

println("\nString " + s + ":\nPermutations: " + getAllPermut(s));

}

static def getAllPermut(def str){
```

```
def crunchifyResult = new HashSet<string>();

if (str == null) {

return null

} else if (str.length() == 0) {

crunchifyResult.add("")
```

see more

• Reply • Share ›

Cracker • 19 days ago

http://algods-cracker.blogspot...

• Reply • Share ›

Vivek Shah • 22 days ago

Recursive solution :

http://ideone.com/qrTf2F

• Reply • Share ›

n.a.s • 25 days ago

Hi,
How we can reduce time complexity to polynomial ?

Thanks

• Reply • Share ›

rahulkadukar ➚ n.a.s • 13 days ago

You cannot, a string with n distinct characters will have n! different permutations, so the best case scenario where you enumerate all n! permutations leads to a n! solution.

• Reply • Share ›

subhadip das • a month ago

What if the array has repeated string . How will the algorithm be modified

• Reply • Share ›

wolverine ➚ subhadip das • a month ago

just add this in 1st line of for loop:

```
if(i!=j && a[i]==a[j])
continue;
```

• Reply • Share ›

nick ➚ subhadip das • a month ago

use a hashset to store what has already been printed, and check if it's already been printed before print it.

▲ | ▼ • Reply • Share ›

lorenzo • a month ago

how is the complexity determined here?

▲ | ▼ • Reply • Share ›

ajay gautam • a month ago

how to print all the permutations taking k characters at a time.?

▲ | ▼ • Reply • Share ›

rose • 2 months ago

HI can someone point the job of second swap call?

▲ | ▼ • Reply • Share ›

ninja ➚ rose • 2 months ago

to restore the string to its original form

▲ | ▼ • Reply • Share ›

rachit • 2 months ago

for the string aabc it should give only 12 anagrams but it gives 24..

▲ | ▼ • Reply • Share ›

Learner • 2 months ago

How do we do it without repitition? eg: copying chars to new array?

▲ | ▼ • Reply • Share ›

goku • 2 months ago

Why there is no explanation of any of the programs as most of us couldn't make out how does it works?

▲ | ▼ • Reply • Share ›

waterloong • 2 months ago

Does this solution assume the string has no repeated characters? It seems so to me.

▲ | ▼ • Reply • Share ›

Rishi Verma • 3 months ago

Java Code http://ideone.com/D0WUhF

1 ▲ | ▼ • Reply • Share ›

Abhijit • 3 months ago

```
void permute2(char* in, int index) {
if (index >= strlen(in)) {
printf("%s\n", in);
} else {
for (int i = index; i < strlen(in); i++) {
swap(&in[index], &in[i]);
permute2(in, index+1);
swap(&in[index], &in[i]);
}
}
}

void swap(char* a, char* b) {
char temp = *a;
*a = *b;
*b = temp;

}
```

Reply • Share ›

Anushkha Singh • 3 months ago

Refer following link:

Permutations of a string

1 Reply • Share ›

Nitin ➚ Anushkha Singh • 3 months ago

Hi ,

Could you suggest C or C++ Editor for windows 7?

Thanks
Naresh

Reply • Share ›

prassanna.mit ➚ Nitin • 3 months ago

Try Vim!!

Reply • Share ›

yo ➚ Nitin • 3 months ago

sublime text is a good editor for any language and any OS. you can add your build system to it too.

1 Reply • Share ›

yo ➚ Nitin • 3 months ago

Sublime text

1 • Reply • Share ›

Pratyush ➚ Nitin • 3 months ago

Code::Blocks

1 • Reply • Share ›

Anushkha Singh ➚ Nitin • 3 months ago

Use Visual studio/ Eclipse if u want an IDE.
If u r looking for just an editor u Notepad++ is good.

• Reply • Share ›

rithustutorials . • 3 months ago

What do you think about this video for getting to understand more about programming for a beginner?

https://www.youtube.com/watch?...

• Reply • Share ›

Aditya Goel • 3 months ago

The Time complexity is O(N!) not O(n*n!).
It can be found solving this recurrence.
T(n) = n.T(n-1) for all n>1
=1 for n=1

7 • Reply • Share ›

ashok • 3 months ago

/* package whatever; // don't place package name! */
This is the solution in Java

```
import java.util.*;

import java.lang.*;

import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */

class Ideone

{

public static void main (String[] args) throws java.lang.Exception

{
```

StringBuilder u=new StringBuilder("ashok");

see more

• Reply • Share ›

Andrey Kaygorodov ashok • 3 months ago

It is mine, similar to your, but an only difference that I used a char array instead of a StringBuilder.
https://gist.github.com/kaygor...

• Reply • Share ›

po • 4 months ago

http://ideone.com/SVP9lw

iterative version you can get it here..

• Reply • Share ›

DS+Algo=Placement po • 3 months ago

dude, it will work only for string of length 3..

• Reply • Share ›

po • 4 months ago

here is the iterative version with time complexity o(n^3)...

void permute(char str[],int n)

{

int i,j,k;

for(i=0;i<n;i++) for(j="0;j<n;j++)" for(k="0;k<n;k++)" if(i!="j" &&="" j!="k" &&="" k!="i)" printf("%c%c%c\n",str[i],str[j],str[k]);="" }="">

• Reply • Share ›

Rahul Assassin po • 7 days ago

Dude...we dont want to just have permutations of 3 character string.Wake up.:)

• Reply • Share ›

bboczeng • 4 months ago

i call this DFS. i don't think this is back tracking. no one is going back at all and there is (especially) no pruning and reject procedure. There is basically no search and this is a full combinatorial problem (all enumeration is accepted). calling it back-tracking, is misleading at its best.

• Reply • Share ›

Mayur B. Shingote • 4 months ago

Is there any dynamic way to reduce time complexity ?

Reply • Share ›

Aditya Goel ➚ Mayur B. Shingote • 3 months ago

Yes, prove P=NP

3 Reply • Share ›

DS+Algo=Placement ➚ Aditya Goel • 3 months ago

can u explain please?

Reply • Share ›

Rahul • 4 months ago

For reference on C++ Strings you may check

https://www.tutorialcup.com/cp...

http://www.cplusplus.com/refer...

Reply • Share ›

The_Geek • 4 months ago

if((i != j) && (*(a+i) == *(a+j))) continue;
using the above statement wont work in all examples like ABCA, this will give 14 permutations, while the answer is just 12.
So to solve the problem we have an idea like-
We can look in present array to see whether we have used this particular character or not?.
Go to the link, this is a working code for permutation of a sting having duplicate characters.
http://ideone.com/fW4tDb

2 Reply • Share ›

Prateek Jain ➚ The_Geek • 4 months ago

Yours code is also not working for "aaa"
because in that case it should six repetitions of "aaa".
But accd to your code it print only once "aaa".

Reply • Share ›

Geeking ➚ The_Geek • 4 months ago

What if you wanted to generate all permutations for a given subset (of length k) from a string (of length n) while still handling duplicates and providing a list of unique strings? This seems much more challenging to do efficiently. Example: ABBCCC where k=4.

2 Reply • Share ›

sandeep • 5 months ago

here is recursive function for string permutations with duplicates

here is recursive function for string permutations with duplicates

```
void permute(string s, int i, int n)

{

if (i == n)

cout << s << endl;

else

{

int w[256];

for (int k = 0; k < 256; k++)

w[k] = 0;

int j;
```

see more

Reply • Share ›

gansai • 5 months ago

@GeeksforGeeks: In the main Algorithms page, there is a section for BackTracking containing links to this page. Could you please add an introduction to what is BackTracking algorithm paradigm. Like how it is given for other algorithmic paradigms? Thanks

1 Reply • Share ›

Balaji • 5 months ago

Could anyone explain this algorithm how backtracking occurs??

Reply • Share ›

Praveen Kumar • 5 months ago

I have written an amateur code , it's working , any suggestion will be welcome.

```
import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class Permutations {

static List<string> getPerm(String str,int len){

List<string> perms = new ArrayList<string>();
```

```
if(len==1){

perms = Arrays.asList(str.split(""));

return perms.subList(1,str.length()+1);

}
```

see more

2 • Reply • Share ›

Aakash Raina • 5 months ago

we can think of each node a single character of the string and then apply dfs or bfs to find the permutations.

• Reply • Share ›

Shaurya • 5 months ago

I'd rather work it out easily.
Use `next_permutation()` from `<algorithm>` .

1 • Reply • Share ›

Aditya Goel ➚ Shaurya • 3 months ago

Don't use that in programming interview.

• Reply • Share ›

Raj • 5 months ago

Here is the implementation without repetitions for Duplicates

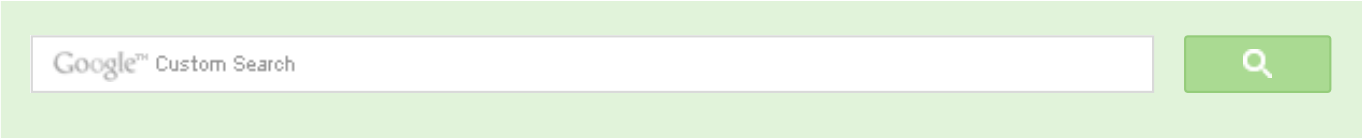
http://ideone.com/9w1N7L

3 • Reply • Share ›

sandeep ➚ Raj • 5 months ago

bro have tried implementing it.i don't think it works for the string "bada"

• Reply • Share ›



- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- 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

Follow @GeeksforGeeks

Recent Comments

- Ashish Aggarwal

  Try Data Structures and Algorithms Made Easy -...

  Algorithms · 17 minutes ago

- Vlad

  Thanks. Very interesting lectures.

Expected Number of Trials until Success · 1 hour ago

- cfh

  My implementation which prints the index of the...

  Longest Even Length Substring such that Sum of First and Second Half is same · 1 hour ago

- Gaurav pruthi

  forgot to see that part ;)

  Bloomberg Interview | Set 1 (Phone Interview) · 1 hour ago

- saeid aslami

  thanks

  Greedy Algorithms | Set 7 (Dijkstra's shortest path algorithm) · 1 hour ago

- Cracker

  Implementation:...

  Implement Stack using Queues · 2 hours ago