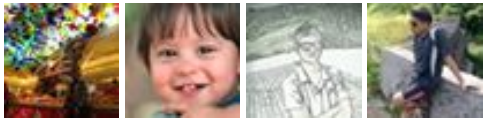
Print all permutations with repetition of characters

Given a string of length n, print all permutation of the given string. Repetition of characters is allowed. Print these permutations in lexicographically sorted order
Examples:

```
Input: AB
Ouput: All permutations of AB with repetition are:
      AA
      AB
      BA
      BB

Input: ABC
Output: All permutations of ABC with repetition are:
       AAA
       AAB
       AAC
       ABA
       ...
       ...
       CCB
       CCC
```

For an input string of size n, there will be n^n permutations with repetition allowed. The idea is to fix the first character at first index and recursively call for other subsequent indexes. Once all permutations starting with the first character are printed, fix the second character at first index. Continue these steps till last character. Thanks to PsychoCoder for providing following C



implementation.

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

/* Following function is used by the library qsort() function to sort
  array of chars */
int compare (const void * a, const void * b);

/* The main function that recursively prints all repeated permutations
  the given string. It uses data[] to store all permutations one by on
void allLexicographicRecur (char *str, char* data, int last, int index
{
    int i, len = strlen(str);

    // One by one fix all characters at the given index and recur for
    // subsequent indexes
    for ( i=0; i<len; i++ )
    {
        // Fix the ith character at index and if this is not the last
        // then recursively call for higher indexes
        data[index] = str[i] ;

        // If this is the last index then print the string stored in d
        if (index == last)
            printf("%s\n", data);
        else // Recur for higher indexes
            allLexicographicRecur (str, data, last, index+1);
    }
}

/* This function sorts input string, allocate memory for data (needed
  allLexicographicRecur()) and calls allLexicographicRecur() for print
  permutations */
void allLexicographic(char *str)
{
    int len = strlen (str) ;

    // Create a temp array that will be used by allLexicographicRecur(
    char *data = (char *) malloc (sizeof(char) * (len + 1)) ;
    data[len] = '\0';

    // Sort the input string so that we get all output strings in
    // lexicographically sorted order
    qsort(str, len, sizeof(char), compare);
```


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

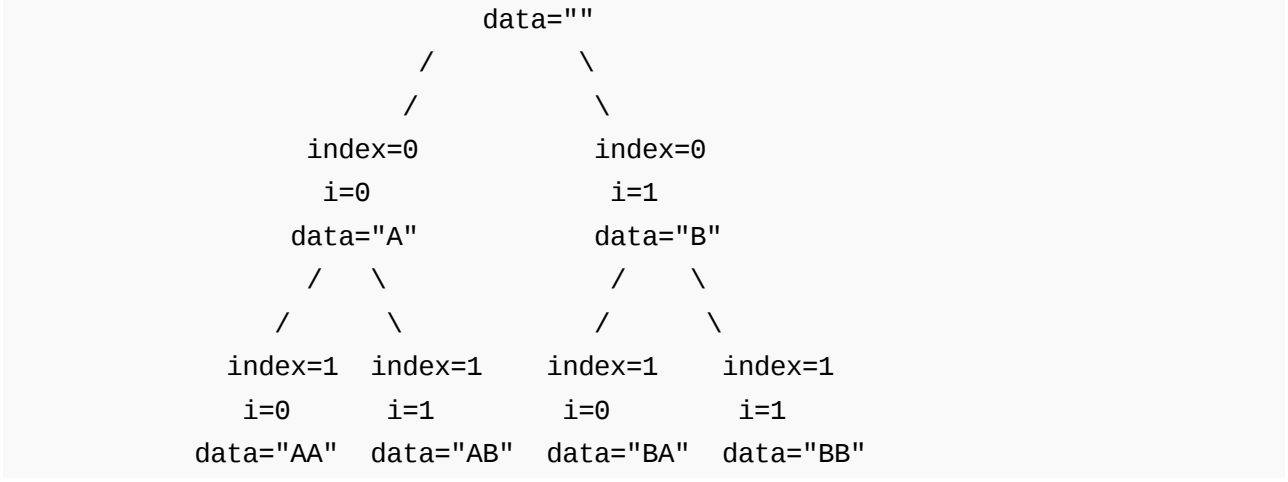
```
    // Now print all permutaions
    allLexicographicRecur (str, data, len-1, 0);

    // Free data to avoid memory leak
    free(data);
}

// Needed for library function qsort()
int compare (const void * a, const void * b)
{
    return ( *(char *)a - *(char *)b );
}

// Driver program to test above functions
int main()
{
    char str[] = "ABC";
    printf("All permutations with repetition of %s are: \n", str);
    allLexicographic(str);
    getchar();
    return 0;
}
```

Following is recursion tree for input string “AB”. The purpose of recursion tree is to help in understanding the above implementation as it shows values of different variables.



In the above implementation, it is assumed that all characters of the input string are different. The implementation can be easily modified to handle the repeated characters. We have to add a

preprocessing step to find unique characters (before calling allLexicographicRecur()).

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)

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

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

19 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

pramod • 2 months ago

It will produce repetitive results for data ="AAB"

• Reply • Share ›

nanda • 2 months ago

Java Code

```
public static void main(String[] args) {
doPermutationRepeat("", new String("ab"));
}

public static void doPermutationRepeat(String prefix, String str) {
if (prefix.length() == str.length()) {
System.out.println(prefix);
return;
}
for (int i = 0; i < str.length(); i++) {
doPermutationRepeat(prefix + str.charAt(i), str);
}
}
```

• Reply • Share ›

One can use for loops and continue statement for forming all combination of c

Omor J. Kocharee • 6 months ago

```
//Make the necessary changes
#include<iostream>
#define loi(a) scanf("%d",&a)
#define _W(t) while(t--)
using namespace std;
int prnt(int p,string s)
{
    if(p==3)
    {

  cout<<s<<endl; return="" 0;="" }="" <pre=""> for(int y=0; y<3; y++
     {

    s[p]='A'+y;
              prnt(p+1,s);
       }

  }
```


asunel • 7 months ago

@GeeksforGeeks: Instead of computing len every time in allLexicographicRec
of the function.

Hello,

I try to generate all repetitive permutation of a number array by putting bound o

For example;
I have my array {3,4,5,6}
My bound is 11.

I would like to generate all repetitive permutations reaching and just crossing 1

3 3 3 3
3 4 3 3
3 3 5 3
3 3 3 6
3 4 4 3
4 4 4
6 6
...

It is quite hard to put such bounds and cardinality of permutation set might cha
pls help

abhishek08aug • a year ago

Intelligent :D

monika • a year ago

MY IMPLEMENTATION

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

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

void repeated_perm(char *str,char *to_print,int n,int len)
{
if(n==len)
{
to_print[n] = '\0';
printf("%s\n",to_print);
return;
}
int i;
for(i = 0; i<len;i++) {="" to_print[n]="str[i];" repeated_perm(str,to_print,n+1,len)
char="" *to_print;="" scanf("%s",s);="" char="" s[20]="ABC" ;="" int="" n="strle
*)malloc(sizeof(char)*(n+1));="" repeated_perm(s,to_print,0,n);="" return="" 1;
aac="" aba="" abb="" abc="" aca="" acb="" acc="" baa="" bab="" bac="" bba=
caa="" cab="" cac="" cba="" cbb="" cbc="" cca="" ccb="" ccc="">
```

AT • a year ago

```
public class PermutationsRepetition {

        private static List<String> permute(String s) {
                List<String> list = new ArrayList<String>();
                char[] c = new char[s.length()];
                return permute(s, list, c, 0);
        }

        private static List<String> permute(String s, List<String> li
                if (count == s.length()) {
                        list.add(new String(c));
```

```
                }
                else {
                        for (int i = 0; i < s.length(); i++) {
                                c[count] = s.charAt(i);
                                permute(s, list, c, count + 1);
                        }
```

see more

Reply • Share ›

aygul AT • a year ago

A good improvment on the code would be defining sizeo f the list. We k
be in the list. So Give it the contstructor. That would gurantee O(1) inse

Reply • Share ›

Prateek Caire • 2 years ago

```
P(i, k)
        if(k == sz)
                while(j < sz)
                        print a[i]
                        if(i < k)
                                i++
                        j++
                return
        for each j from i to sz-1
                swap(i, j)
                P(i, k+1)
                P(i+1, k+1)
```

Reply • Share ›

sentinel • 2 years ago

Why is qsort required.....Will the answer be correct even without it

```
/* Paste your code here (You may delete these lines if not writing co
```

• Reply • Share ›

Sreenivas Doosa ➦ sentinel • a year ago

Please read the problem statement. We have to print the permutations
to sort the input string first in ascending order of alphabets.

Suppose input string is BA and if you dont sort then as per the given al
not in lexicographic order.
BB
BA
AB
AA

• Reply • Share ›

Ali • 2 years ago

I tried another method that uses a counter like a timer starting at 111 and endi

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

void allLexicographic (char *str){
    int len = strlen(str);
    int  counter[len];
```

```
    for(int x = 0; x < len; x++){
        counter[x] = 1;
    }

    while (!done) {
```

see more

• Reply • Share ›

kartikaditya • 2 years ago

For iterative version refer -> http://kartik-bommepally.blogs...

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

using namespace std;

void swap(char* s, int i, int j) {
    if (s[i] == s[j]) {
        return;
    }
    s[i] ^= s[j];
    s[j] ^= s[i];
    s[i] ^= s[j];
}
```

see more

candis • 2 years ago

for a string will all unique characters and with length m we can create a m digi
For eg: for ABC we can create a 3 digit array with base 2
Associating the characters with numbers viz a-0 b-1 c-2
starting from 000-222 (by adding 1 in base 3), we can create all the permutatic

```
/* Paste your code here (You may delete these lines if not writing coc
```

Reply • Share ›

Star_Trek candis • a year ago

@candis-gud code....

```
/* Paste your code here (You may delete these lines if not writ
```

Reply • Share ›

PsychoCoder • 2 years ago

The function in the allLexicographic is

allLexicographicRecur (str, ata, len-1, 0);

just update it as

allLexicographicRecur (str, data, len-1, 0);

Reply • Share ›

GeeksforGeeks PsychoCoder • 2 years ago

@PsychoCoder: Thanks for pointing out the typo. We have corrected i

Reply • Share ›