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

Count number of binary strings without consecutive 1's

Given a positive integer N, count all possible distinct binary strings of length N such that there are no consecutive 1's.

Examples:

```
Input:  N = 2
Output: 3
// The 3 strings are 00, 01, 10

Input: N = 3
Output: 5
// The 5 strings are 000, 001, 010, 100, 101
```

This problem can be solved using Dynamic Programming. Let a[i] be the number of binary strings of length i which do not contain any two consecutive 1's and which end in 0. Similarly, let b[i] be the number of such strings which end in 1. We can append either 0 or 1 to a string ending in 0, but we can only append 0 to a string ending in 1. This yields the recurrence relation:

```
a[i] = a[i - 1] + b[i - 1]
b[i] = a[i - 1]
```

The base cases of above recurrence are a[1] = b[1] = 1. The total number of strings of length i is just a[i] + b[i].

Following is C++ implementation of above solution. In the following implementation, indexes start from 0. So a[i] represents the number of binary strings for input length i+1. Similarly, b[i] represents binary strings for input length i+1.

```
// C++ program to count all distinct binary strings
// without two consecutive 1's
#include <iostream>
using namespace std;

int countStrings(int n)
{
    int a[n], b[n];
    a[0] = b[0] = 1;
    for (int i = 1; i < n; i++)
    {
        a[i] = a[i-1] + b[i-1];
        b[i] = a[i-1];
    }
    return a[n-1] + b[n-1];
}


// Driver program to test above functions
int main()
{
    cout << countStrings(3) << endl;
    return 0;
}
```

Output:

```
5
```

Source:
courses.csail.mit.edu/6.006/oldquizzes/solutions/q2-f2009-sol.pdf

This article is contributed by **Rahul Jain**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tags: Dynamic Programming

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

Vaibhav Singh • 25 days ago

```
#include <iostream>
#include <cmath>
using namespace std;

int CountString(int Length);

int CountString(int Length){
int Sum = pow(2, Length);
for(int i = 2; i <= Length; i++)
int Sum = Sum + pow(-1, i+1) * (Length - i + 1) * pow(2, Length-i);
return Sum;
}

int main(){
cout << "Total Possible Strings: " << CountString(3) << endl;
return 0;
}
```


Mission Peace • a month ago

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

Check my video on explanation.

Ayan Banerjee • a month ago

This question can be solved in log(n) time. The solution to this problem is the Fibonacci number recursion f(n) = f(n-1)+f(n-2) and base case f(0) =1, f(1) =2 , So we can use the log(n) time solution of Fibonacci to solve this problem. (3 2; 2 1)^n

• Reply • Share ›

Vito • 2 months ago

Just speaking mathematically,

Total Number of strings possible with 0 and 1, let's say X -> 2^n

Total Number of strings with consecutive 1s, let's say Y -> For i=2 to n, Sum (n-1)C(i-1)

And take difference of X-Y, to get number of strings without consecutive ones.

• Reply • Share ›

Aditya Goel • 3 months ago

I didn't get the solution given here, it is concise but so confusing.
Refer mine - Simple & elegant
http://ideone.com/eS4zn4

• Reply • Share ›

Sanket Patel ➚ Aditya Goel • 2 months ago

Overlapping subproblems. Not elegant.

• Reply • Share ›

Aditya Goel ➚ Sanket Patel • 2 months ago

see my comment on the code
"can optimize further using DP"

• Reply • Share ›

anyonymous • 3 months ago

C code to display all the string using backtracking:
http://ideone.com/jMsffL

• Reply • Share ›

beginner555 • 3 months ago

```
#include<stdio.h>
int main()
{int i;
int n=8;
for(i=0;i<4;i++)
{binary(i);
printf("\n");}
```

return 0;}

int getbit(int i,int bit)
{int x;
x=i&(1<<bit)?1:0; return="" x;="" }="" void="" binary(int="" l)="" {int="" x,i,flag="1;"
for(i="0;i<2;i++)" {x="getbit(l,i)+getbit(l,i+1);" if(x%2="=0" &&="" x!="0)" flag="0;" }if(flag="=1)"
printf("%d",l);="" }="">

• Reply • Share ›

Ankit • 4 months ago

Why is it return a[n-1] + b[n-1]; and not return a[n] + b[n]; ??

• Reply • Share ›

Nirdosh Pal ↗ Ankit • 3 months ago

bcos we r starting from 0,then we must end the array on (n-1)th element of the given array...

• Reply • Share ›

Amit ↗ Ankit • 3 months ago

Array is represented from 0 to n-1. And since the last element holds the answer we are returning that. Hope I am clear.

• Reply • Share ›

Vishal Johri • 4 months ago

Answer: pow(2,n)- (n*(n-1))/2

• Reply • Share ›

Kenneth • 5 months ago

The result is actually Fibonacci number, which can be got by O(n) and O(1) space complexity, where n is the bit count.
http://ideone.com/5rlq2Y

Results:
Result for 1 bits is 2
Result for 2 bits is 3
Result for 3 bits is 5
Result for 4 bits is 8
Result for 5 bits is 13
Result for 6 bits is 21
Result for 7 bits is 34
Result for 8 bits is 55
Result for 9 bits is 89
Result for 10 bits is 144

3 • Reply • Share ›

Anuj Garg ➚ Kenneth • 4 months ago
Its nothing but Fibonacci series.
• Reply • Share ›

Siya • 5 months ago
How to think in such problem??I don't think that i will come up with this solution in interview.
1 • Reply • Share ›

guest ➚ Siya • 3 months ago
see how the solution of the smaller subproblem(s)(here there are two *types* of subproblems, those ending with 0 and with 1), can be extended to form larger solutions. if n=2 and n=3 is big take even smaller examples n=1 and n=2. See how solutions for n=1 has been extended to form solution for n=2
• Reply • Share ›

Sai Gudigundla ➚ Siya • 5 months ago
It's definitely is not obvious that this is Fibonacci series. The way I tackled it when I was asked this question is by starting with n = 1, n=2, n=3, n=4 and started jotting down the answers. Once I looked at the answers I was quickly able to recognize that it's Fibonacci series.
1 • Reply • Share ›

Ratheesh Mohan • 6 months ago

```
private static int countStrings(int n)
{
if (n == 0) return 0;
int endsZero = 1;
int endsOne = 1;
for (int i = 2; i <= n; ++i)
{
var temp = endsZero;
endsZero += endsOne;
endsOne = temp;
}
return endsZero + endsOne;
}
```

• Reply • Share ›

Atanu Sikder • 6 months ago
#Similar to series 3,5,8,13,21..... for n=2,3,4,5.....
respectively....

#include<bits stdc++.h="">

```
using namespace std;

int count(int n)

{

int a=3,b=5,count=0;

if(n==2)

return 3;

if(n==3)

return 5;
```

see more

• Reply • Share ›

Pranjal Goswami • 6 months ago

a[0] = b[0] = 1; is not correct. It has to be a[1] = b[1] =1;

• Reply • Share ›

m_asif • 6 months ago

http://ideone.com/tV6olb

Not as efficient as the original solution but this was the best I could think of. Just in case you are required to print all the combinations.

• Reply • Share ›

neelabhsingh • 6 months ago

In above problem, suppose size of binary string is n,
now concentrate the last index. last index has two options 0, or 1.
I. If last is 0, then previous has two options like
.....1,0
.....0,0

II. and if last is 1 then previous has only one option
.....0,1

Now a[], show number of the string which has 0 as last binary. and b[] show number of the string which has 1 as last binary, So
answer is =a[n-1]+b[n-1].

a[n-1] counts the string which ends "0" of size "n-1" and b[n-1] counts the string which ends "1" of size "n-1"

I think best way to demonstrate, suppose the n=3,
0 1 <= end of string has 2 options
/ \ |
0 1 0
/ \ / / \
0 1 0 1 0
1 2 3 4 5=> total number of such string is 5.

• Reply • Share ›

rohit_90 • 7 months ago

We can optimize the space. Here is my java code that uses array of only two integers.

Link: http://ideone.com/P0AFmK

• Reply • Share ›

aa1992 • 7 months ago

Here is my java code for printing the strings without consecutive ones.

please report if any problem.

http://ideone.com/Gw2H8e

• Reply • Share ›

Sachin • 8 months ago

This is similar to (n+1)th Fibonacci Number

4 • Reply • Share ›

Chakrapani • 8 months ago

void CountNumber(int a)

{

int p=1;

for(int i=0;i<a;i++) {="" p="p*2;" }="" if(p!="1)" {="" for(int="" i="2;i<=a;i++)" {="" p="p-((a-i)+1);" }="" }="" cout<<p;="" }="">

• Reply • Share ›

PJ • 8 months ago

How about this?

unsigned int mask1 = 0x55; // Generate the mask dynamically based on the input size. currently its static.
unsigned int mask2 = 0xAA;

```
unsigned int num = 255;
unsigned int count = 0;

while(num > 0)
{
if ( (num & 0x55) == num)
count++;
if ( (num & 0xAA) == num)
count++;
num--;
}

Printf: count
Space: O(1)
```

• Reply • Share ›

Aditya Sharma • 8 months ago

Note that total for n is a[n] + b[n] (which are as defined above). since b[n] is a[n-1] the total is a[n] + a[n-1]. The base cases can be defined as needed.Hence the solution is fibonacci n+1 .

2 • Reply • Share ›

Jiggs • 8 months ago

No need for arays. Following code is without extra space:

```
#include <iostream> <iostream>using namespace std; namespace std;typedef unsigned int
u32;
u32 NumStrsWithoutConsec1s(u32 n)
{
switch(n)
{
case 0:
return 0;
case 1:
return 2;
}
int a = 1;
int b = 1;
for(u32 i = 2; i <= n; ++i)
{
u32 temp = b;
b = a;
```

see more

• Reply • Share ›

Rashid khan • 9 months ago

what if we want to print those bnary strings also??

• Reply • Share ›

aa1992 ➚ Rashid khan • 7 months ago

Here is my java code for printing the strings without consecutive ones.

please report if any problem.

http://ideone.com/Gw2H8e

1 • Reply • Share ›

dark_warrior ➚ aa1992 • 6 months ago

can u provide the c code also?

1 • Reply • Share ›

www ➚ Rashid khan • 9 months ago

http://sdream4peace.blogspot.i...

• Reply • Share ›

krishna • 9 months ago

we can do this one in O(1) time

just by applying permutation concepts formula will be pow(2,n)-(sum of n-1 to 1)

```
int binary_with_out_cons_1(int n){
return std::pow(2,n)-(n*(n-1))/2;
}
```

correct me if am wrong

• Reply • Share ›

jatara ➚ krishna • 8 months ago

check it for n >=4.

• Reply • Share ›

shreygupta_dtu ➚ krishna • 9 months ago

no
the ans is
return std::pow(2,n-1)+1;

• Reply • Share ›

Arnab ➚ shreygupta_dtu • 9 months ago

@shreygupta_dtu ,@krishna
check it for n=5

• Reply • Share ›

Mohaan Raja • 9 months ago

what is the output for N=1?
Above program gives 2 as output.... Is it right?? Base condition is not verfied, Isn't it?

• Reply • Share ›

Jun ➚ Mohaan Raja • 9 months ago

its right, o/p is 0 and 1

1 • Reply • Share ›

devakar verma • 9 months ago

we can think it as fibonacci series with base case n=1 and n=2

```
#include <iostream>
using namespace std;

int count(int n)
{

if (n==1)
return 3;
else if (n==2)
return 5;
else
return count(n-1)+count(n-2);

}

int main() {
int n;
cin>>n;
cout<<count(n); return="" 0;="" }="">
```

5 • Reply • Share ›

thevagabond85 ➚ devakar verma • a month ago

```
if (n==1)
return 2;
else if (n==2)
return 3;
```

• Reply • Share ›

devakar verma ➚ thevagabond85 • a month ago

ya its typo

• Reply • Share ›

Subham • 9 months ago

dynamic programming with memoization... complexity of o(2*n)..
http://ideone.com/htgJX7#view_...

2 ^ | v • Reply • Share ›

Saurabh Suman • 10 months ago

A simple solution :

```
#include <iostream>

using namespace std;

int countStrings(int n)

{

int a=1,b=1,c=0;

for (int i = 1; i <= n; i++)

{

c=a+b;

a=b;

b=c;
```

see more

3 ^ | v • Reply • Share ›

Sharika Pongubala • 10 months ago

Doesn't use extra space

```
import java.util.*;

public class countOnes {

public static int countConsec(int n, int prev, int ind) {

int count=0;

if(ind==n) return 1;

if (prev==-1) {

count += countConsec(n, 1, ind+1);

count += countConsec(n, 0, ind+1);
```

```
}

else {
```

see more

• Reply • Share ›

Gitanshu behal • 10 months ago

Improved Version:

```
int countStrings(int n)
{
int a[n+2]; // Extra space if n<2....
a[0] = 2; a[1] = 3;
for (int i = 2; i < n; i++)
{
a[i] = a[i-1] + a[i-2];
}

return a[n-1];
}
```

• Reply • Share ›

Hugo • 10 months ago

Note that the result is Fibonacci (N + 2). It may be calculated in O (log N).

• Reply • Share ›

np • 10 months ago

O(1) solution
short and simple

```
#include<stdio.h>

int cal(int n)
{
int a,b,c,d,e,i,f,ans=0;
if(n==0)
return 0;
else if(n==1)
return 2;
else if(n==2)
{
return 3;
}
else if(n==3)
```

```
{
```

see more

• Reply • Share ›

Abhishek • 10 months ago

Thanks Rahul for the explanation.
I would like to add that it is not a direct fibonacci series.The recurrence relation should be:
result[n]= (2*result[n-1]) + result[n-2] with base case as
result[0]=1 and result[1]=2

I checked it with n=4,the output should be 8
0000,0100,0001,1000,0010,0101,1010,0101

4 • Reply • Share ›

Load more comments

Subscribe Add Disqus to your site Privacy

- 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

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 3 hours ago

-