Dynamic Programming | Set 7 (Coin Change)

Given a value N, if we want to make change for N cents, and we have infinite supply of each of S = { S1, S2, .. , Sm} valued coins, how many ways can we make the change? The order of coins doesn't matter.

For example, for N = 4 and S = {1,2,3}, there are four solutions: {1,1,1,1},{1,1,2},{2,2},{1,3}. So output should be 4. For N = 10 and S = {2, 5, 3, 6}, there are five solutions: {2,2,2,2,2}, {2,2,3,3}, {2,2,6}, {2,3,5} and {5,5}. So the output should be 5.

1) Optimal Substructure

To count total number solutions, we can divide all set solutions in two sets.

1) Solutions that do not contain mth coin (or Sm).

2) Solutions that contain at least one Sm.

Let count(S[], m, n) be the function to count the number of solutions, then it can be written as sum of count(S[], m-1, n) and count(S[], m, n-Sm).

Therefore, the problem has optimal substructure property as the problem can be solved using solutions to subproblems.

2) Overlapping Subproblems

Following is a simple recursive implementation of the Coin Change problem. The implementation simply follows the recursive structure mentioned above.

```
#include<stdio.h>

// Returns the count of ways we can sum  S[0...m-1] coins to g
int count( int S[], int m, int n )
{
    // If n is 0 then there is 1 solution (do not include any
    if (n == 0)
        return 1;

    // If n is less than 0 then no solution exists
    if (n < 0)
        return 0;

    // If there are no coins and n is greater than 0, then no
    if (m <=0 && n >= 1)
        return 0;

    // count is sum of solutions (i) including S[m-1] (ii) exc
    return count( S, m - 1, n ) + count( S, m, n-S[m-1] );
}

// Driver program to test above function
int main()
{
```

Popular Posts

- Top 10 Algorithms and Data Structures for Competitive Programming
- Top 10 algorithms in Interview

```
    int i, j;
    int arr[] = {1, 2, 3};
    int m = sizeof(arr)/sizeof(arr[0]);
    printf("%d ", count(arr, m, 4));
    getchar();
    return 0;
}
```

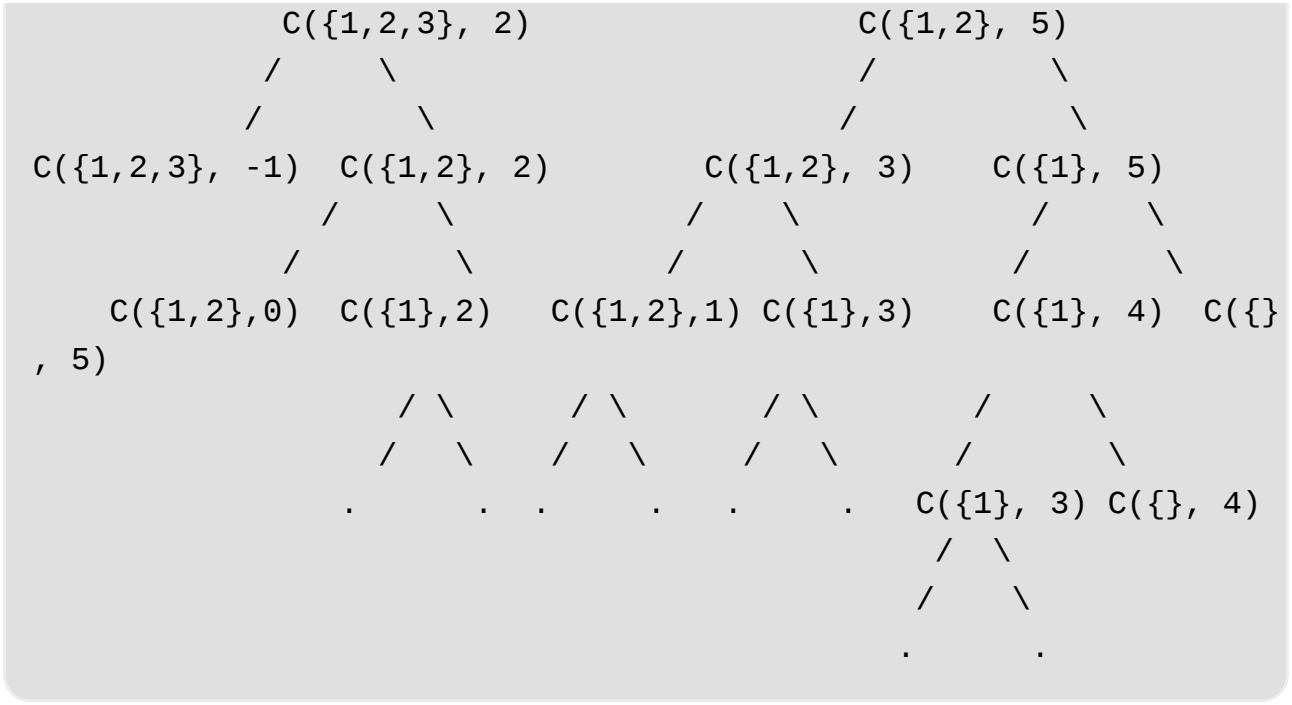
Run on IDE

It should be noted that the above function computes the same subproblems again and again. See the following recursion tree for S = {1, 2, 3} and n = 5.
The function C({1}, 3) is called two times. If we draw the complete tree, then we can see that there are many subproblems being called more than once.

```
C() --> count()
                              C({1,2,3}, 5)
                            /               \
                          /                   \
```

Questions

- How to begin with Competitive Programming?
- Reflection in Java
- Memory Layout of C Programs
- Push Relabel Algorithm
- Heavy Light Decomposition
- Sorted Linked List to Balanced BST
- Generics in Java
- Aho-Corasick Algorithm for Pattern Searching
- Binary Search , QuickSort , MergeSort , HeapSort



property. So the Coin Change problem has both properties (see this and this) of a dynamic programming problem. Like other typical Dynamic Programming(DP) problems, recomputations of same subproblems can be avoided by constructing a temporary array table[][] in bottom up manner.

Dynamic Programming Solution

C Python

```
#include<stdio.h>

int count( int S[], int m, int n )
```

```
{
    int i, j, x, y;

    // We need n+1 rows as the table is consturcted in bo
    // the base case 0 value case (n = 0)
    int table[n+1][m];

    // Fill the enteries for 0 value case (n = 0)
    for (i=0; i<m; i++)
        table[0][i] = 1;

    // Fill rest of the table enteries in bottom up manne
    for (i = 1; i < n+1; i++)
    {
        for (j = 0; j < m; j++)
        {
            // Count of solutions including S[j]
            x = (i-S[j] >= 0)? table[i - S[j]][j]: 0;

            // Count of solutions excluding S[j]
            y = (j >= 1)? table[i][j-1]: 0;

            // total count
            table[i][j] = x + y;
        }
    }
    return table[n][m-1];
}

// Driver program to test above function
int main()
{
    int arr[] = {1, 2, 3};
    int m = sizeof(arr)/sizeof(arr[0]);
    int n = 4;
    printf(" %d ", count(arr, m, n));
    return 0;
}
```

Run on IDE

- Sorting
- Hashing
- Analysis of Algorithms
- Mathematical Algorithms
- Randomized Algorithms
- Recursion

Tags

Adobe Advance Data Structures Advanced Data Structures Amazon array puzzle D-E-Shaw DBMS design Divide and Conquer Dynamic

Output:

```
4
```

Time Complexity: O(mn)

Following is a simplified version of method 2. The auxiliary space required here is O(n) only.

```
int count( int S[], int m, int n )
{
    // table[i] will be storing the number of solutions for
    // value i. We need n+1 rows as the table is consturcted
    // in bottom up manner using the base case (n = 0)
    int table[n+1];

    // Initialize all table values as 0
    memset(table, 0, sizeof(table));

    // Base case (If given value is 0)
    table[0] = 1;

    // Pick all coins one by one and update the table[] values
    // after the index greater than or equal to the value of t
    // picked coin
    for(int i=0; i<m; i++)
        for(int j=S[i]; j<=n; j++)
            table[j] += table[j-S[i]];

    return table[n];
}
```

Run on IDE

Thanks to Rohan Laishram for suggesting this space optimized version.

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

References:
http://www.algorithmist.com/index.php/Coin_Change

214 Comments Category: Dynamic Programming Tags: Dynamic Programming

Related Posts:

- Find if string is K-Palindrome or not

minutes ago

Anmol Varshney Easier to understand:...

Split a Circular Linked List into two halves · 27 minutes ago

Bala Murali Krishna Java implementation which works for multiple...

Find the maximum repeating number in O(n) time and O(1) extra space · 33 minutes ago

Maneesh What company ?

Find the first circular tour that visits all petrol pumps · 49 minutes ago

Deepak which data structure you used??

D E Shaw Interview Experience | Set 16 (On-Campus for Internship) · 1 hour ago

Banbari Sharma what about online and offline algorithms

Bin Packing Problem (Minimize number of used Bins) · 1 hour ago

- Find maximum length Snake sequence
- Minimum time to finish tasks without skipping two consecutive
- Maximum profit by buying and selling a share at most k times
- A Space Optimized Solution of LCS
- Ways to arrange Balls such that adjacent balls are of different types
- Count number of ways to fill a “n x 4″ grid using “1 x 4″ tiles
- Weighted Job Scheduling in O(n Log n) time

Previous post in category

Next post in category

(Login to Rate and Mark)

3.2 Average Difficulty : **3.2/5.0**
Based on **47** vote(s)

Add to TODO List

Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.

214 Comments GeeksforGeeks

Amit Maheshwari • 5 days ago

Can we find the exact number of each coins used?

^ | v • Reply • Share ›

Shivam Maharshi • 18 days ago

Python Code: https://github.com/shivam-maha...

^ | v • Reply • Share ›

Sahil Dhawan • 22 days ago

Not able to understand a single word here...I watched Tushar Roy problem. How do you guys figured out the DP solution using the 2 here? Did you guys consulted some other source like me?
I still cannot figure out how on excluding coins, we're subtracting t coin, shouldn't the change (i.e n) remain the same?

2 ^ | v • Reply • Share ›

santhosh ➦ Sahil Dhawan • 9 days ago

For example:I will try to elaborate the tree diagram given.
If you want to make a sum of 5 using 1,2,3...split it as belo
(no. of ways you get 5 with 3)+(no. of ways you get 5 witho
-->(no. of ways you get 5 with 3) = C({1,2,3},2) because yo
require is only 2.(5-3=2) => count(S, m, n-S[m-1]).
-->(no. of ways you get 5 without 3)=C({1,2},5) because 3
element less in the set. => count(S, m - 1, n)
Now he is extending the procedure as count(S, m - 1, n)

Shashank Shekhar • 24 days ago

Is this same as the subset sum problem where we want to see nu

Aman Aggarwal ➦ Shashank Shekhar • 19 days ago

It's not because you have "an infinite supply of each kind o

Vashu Garg • a month ago

sir can you help me to understand simplified version of method 2 I

baymax28 ➦ Vashu Garg • 16 days ago

It is not exactly the method's 2's implementation.

It is an approach entirely different from the recursively rela
arithmetic involved in the recursive relation.

Somya kumar Sodani • a month ago

is there any solution if the number of coins which can be used are
say k=4 means only 4 coins can be used.

hisopogae • a month ago

Can someone help me to type a somethink like:
value | amount
amount mean a max use in find
10, 2; 5, 5; 1, 4
from example number 18, should return a 2 combinations:

10 5 1 1 1
5 5 5 1 1 1

• Reply • Share ›

dronz3r • a month ago

I'm solving these dynamic programming questions for the first tim
above mentioned solution after hours of trying :(Maybe, it get easi

• Reply • Share ›

Rahul Vaidya • 2 months ago

I believe this problem can be used to build up 0-N knapsack as we

• Reply • Share ›

NewBee • 2 months ago

how to print the sets of solutions ?
like {1,1,1,1},{1,1,2},{2,2},{1,3}.
Using backtracking seem will get duplicate set.

1 • Reply • Share ›

abhi • 2 months ago

// Initialize all table values as 0

memset(table, 0, sizeof(table));
this is wrong

^ | v • Reply • Share ›

Manthan Barot • 3 months ago

what are the real time applications for coin change problem

1 ^ | v • Reply • Share ›

mohit • 3 months ago

int totalWays(vector<int> coins, int sum, int partial, int index, int n)

if(index >= n || partial > sum){return 0;}

if(partial == sum){return 1;}

int ways = 0;

for(int i = index+1;i<n;i++){ for(int="" j="1;j<=sum/coins[i];j++){" i
+="totalWays(coins," sum,="" sub="" ,="" i,="" n);="" }="" }="" retu

^ | v • Reply • Share ›

Siva • 3 months ago

If the given coins are not sorted, Below code will correct.
http://code.geeksforgeeks.org/...

^ | v • Reply • Share ›

Binary • 4 months ago

Can someone help me understand what happens when we suppl

For eg. { 1, 1, 2} to get sum 4?

Reply • Share ›

abhishek Binary • 4 months ago

note we have infinite supply of each type of coin so your qu
considered as 2 units supply of coin 1

Reply • Share ›

Axanamaxa • 4 months ago

Does anyone know how to edit one of the above approaches so th
each coin ?

For example:

All possible combinations with coins 1 = 30 // numbers purely ficti
All possible combinations with coins 2 = 20
All possible combinations with coins 5 = 10

Reply • Share ›

abhishek Axanamaxa • 4 months ago

you can print desired values using table[][] .for ex no. of wa

Reply • Share ›

Axanamaxa abhishek • 4 months ago

Hey, thanks for the reply. Unfortunately I'm not that
version. Is there any chance you could paste an ex

Reply • Share ›

abhishek ➚ Axanamaxa • 4 months ago

https://ideone.com/WJcTOI

^ | v • Reply • Share ›

Axanamaxa ➚ abhishek • 4 months ago

I can't open the link ?

^ | v • Reply • Share ›

abhishek ➚ Axanamaxa • 4 months ago

https://ideone.com/JXciaJ

1 ^ | v • Reply • Share ›

Axanamaxa ➚ abhishek • 4 months ago

Thanks a lot !

^ | v • Reply • Share ›

Lannister007 • 4 months ago

In DP method , S[] array has to be the sorted one ?

^ | v • Reply • Share ›

AlienOnEarth • 5 months ago

GeeksforGeeks

I just want to add a minor comment here. In the brute force approa

if (m <=0 && n >= 1) // --> should not include n>=1 because other
// are already covered in previous 2 conditions

1 ^ | v • Reply • Share ›

himanshuk ➦ AlienOnEarth • 3 months ago

yes, check n >= 1 is redundant

^ | v • Reply • Share ›

dheeru487 • 5 months ago

How about following code :

```
int count (int S[], int m, int n)
{
if (n == 0)
return 1;
if (n < 0)
return 0;

int ret = 0;
for (i=1; i<= m; i++)
ret += count(S, m, n-S[i]);
}
```

^ | v • Reply • Share ›

AlienOnEarth ➦ dheeru487 • 5 months ago

@dheeru487: here the assumption is all the elements are
which may not be the case. Also, it will generate same se
generate 1,1,2 and 1,2,1. But these 2 sequences are sam
permutations of size m (in worst case).

^ | v • Reply • Share ›

dheeru487 ➚ AlienOnEarth • 5 months ago

@AlienOnEarth Yeah. I get the problem. Thanks a

• Reply • Share ›

翁林君 • 6 months ago

Actually this one is the same as “Find number of solutions of a lin variables”, the solution here is more clear.

1 • Reply • Share ›

Pranay ➚ 翁林君 • 5 months ago

Wow... _/_ Unique observation..

• Reply • Share ›

shivam shukla ➚ Pranay • a day ago

Sorry couldn't get ... Can you please elaborate it ?

• Reply • Share ›

abhishek • 6 months ago

very easy and simple way to solve this problem.....
http://algorithms.tutorialhori...

• Reply • Share ›

Ankit • 7 months ago

Here is a working code which prints the optimal coin change requi

http://www.edufyme.com/code?id...

1 ^ | v • Reply • Share ›

gaurav singh • 7 months ago

my code is counting cases like (1,2,3) and (3,2,1) as different con
help me out...thanks in advance

```
public static int totalways(int n,int[] change){

if(n==0)

return 1;

if(n<0)

return 0;

int i;
int result=0;
for(i=0;i<=change.length-1;i++){
result=result+totalways(n-change[i],change);
}

return result;
}
```

^ | v • Reply • Share ›

Rahul • 7 months ago

Please help me to understand the recursive solution of this proble

^ | v • Reply • Share ›

Vivek Garg ➚ Rahul • 7 months ago

It is simple. Either you take a coin to make change or not t
So if take last coin then recursion will be count(S,m-1,n-s[
and if we discard last coin then count(S,m-1,n);

^ | v • Reply • Share ›

raj007 ➚ Vivek Garg • a month ago

It is not count(S,m-1,n-s[m]) , it is count(S,m,n-s[m

^ | v • Reply • Share ›

Jason • 8 months ago

Final solution fails if the coins are not in non-decreasing order. for solution provided says 2.

^ | v • Reply • Share ›

Manoj ➚ Jason • 7 months ago

The answer is 4 and the solution provided says 4
{1,1,1,1,1,1},{1,1,1,3},{1,5},{3,3}
Observe carefully you will get it..

^ | v • Reply • Share ›

Sakshi Singhal • 8 months ago

Can someone please explain the simplified version of method 2? Browsed through the comments but couldn't find any satisfactory

^ | v • Reply • Share ›

Vivek Garg ➦ Sakshi Singhal • 7 months ago

Observe that all values of table set to 0. So if a coin can't b
it will added to 0 and has assigned itself of as zero.
Just take an example, write it on paper and you can see th

▲ | ▼ • Reply • Share ›

Tushar Dwivedi • 9 months ago

I solved this problem with all the three ways. ie. Recursion, DP-m

C++ implementation : https://ideone.com/14bBIv

.

Below were the results :
(Input taken via stdin)

For n = 10
recursion : 0 seconds and 14 microseconds
memoization : 0 seconds and 17 microseconds
tabulation : 0 seconds and 17 microseconds

For n =100
recursion : 0 seconds and 1300 microseconds
memoization : 0 seconds and 1270 microseconds
tabulation : 0 seconds and 35 microseconds

What I am unable to understand , is that why for slightly larger inp
time as simple recursive solution ? Shouldn't memoization take e
skips many useless calculations from the matrix, that have to be
calls ?

2 ^ | v • Reply • Share ›

raj007 ➚ Tushar Dwivedi • a month ago

Probably it is due to recursive calls. :)

^ | v • Reply • Share ›

Arpit Quickgun Arora • 9 months ago

explanation for bottom-up method -
just convert the logic count(S, n, m) = count(S, n - Sm, m) + cour
table[n][m] = table[n][m-1] + table[n - s[m]][m].
since we have already initialised the values for n = 0 and m = 0(if
loop from n = 1onwards and fill the table.
Also, if n < 0, return 0;
Now while filling the table, we can notice that to find any cell value
same row for lesser values of n(which have already been filled). U
derived.

^ | v • Reply • Share ›

dividebyzero • 9 months ago

In the last case, shouldn't it be `for(int j=S[i]; j<=n; j+=S[i])`?

^ | v • Reply • Share ›

Mandeep ➚ dividebyzero • 9 months ago

Even I had the same doubt, but later I figured this out.
Suppose u have coins {1,3} with n = 5.
Suppose that you have already processed the j loop for co
all values from 0 to 5 can be done in only 1 way(i.e. using

Now think that you are processing the j loop for S[i] = 3.
That means you have a new way to make value 3 (111, 3)
so u update table[3] to 2. But according to your logic, it sho
according to you, j += S[i]). But think about the case of 4. I
one was 1111, now the new one is 1,3.
Because of this reason, you do j++ and NOT j+=S[i].