Unofficial Editorials December Lunchtime

22

Hello Guys, I'm back with another set of editorials in hope you would like them. :)

I hadn't attended December Cook Off due to certain unavoidable circumstances, that's the reason of no editorials for Cook Off.

Without wasting much time, Here are the editorials.

2

Days in a Month

Difficulty: Cakewalk

Problem:

Given number of days in month and Day as on 1st of given month, Find out frequency of each day.

Solution:

Simplest solution use if else statements (Carefully though).

For W = 28, print 4 for each day, because a month with 28 days will always have 4 Sunday, 4 Monday ans so on.

For W = 29, print 4 for all days except the day of first of month. Example, for input 29 mon, ans is 5 4 4 4 4 4 4. (frequency of monday increased by 1).

For W = 30, print 4 for all days except the day of first of month and its next day. Example, for input 29 mon, ans is 5 5 4 4 4 4 4. (frequency of monday increased by 1).

For W = 31, print 4 for all days except the day of first of month and next two days. Example, for input 29 mon, ans is 5 5 5 4 4 4 4. (frequency of monday increased by 1).

For proof of above, You may consult Calendar. :D

Link to my Code

Strange Function

Difficulty:Easy

Problem:

Given A and N, compute $f(A^N)$, where $f(X)$ is defined as:

- Compute the sum of digits of A; let's denote this sum by S.
- If S is a single-digit integer, then F(A) = S.
- Otherwise, F(A) = F(S).

Solution:

FIrst thing, make a function $f(X)$ as described above. Now, Notice that we need to calculate $f(ans)$ where ans = A^N, we can directly compute $f((f(A))^N)$. Doing this avoids the issue of overflow.

Typical Modular exponentation Code (Refer geeksforgeeks) int power(int x, unsigned int y, int p) { int res = 1; // Initialize result

```
x = x % p;  // Update x if it is more than or
            // equal to p

while (y > 0)
{
    // If y is odd, multiply x with result
    if (y & 1){
        res = (res*x);
        if(res>=p)res = res%p;
    }
    // y must be even now
    y = y>>1; // y = y/2
    x = (x*x) % p;
}
return res;

}
```

Now, our solution is similar to modular exponentation, with one change. In Modular exponentation, we take res = (res*x); if res >= p, res = res%p;

But in our solution, we take res = res*x; $if(res >= 10) res = f(res)$. This way, we are calculating $f(A^N)$ in the same manner as Modular exponentation.

That's all.

Question tags:

editorials ×280
ltime55 ×139
abx01 ×32
abx02 ×28
nw1 ×25

question asked: **30 Dec '17, 23:13**

question was seen: **2,649 times**

last updated: **yesterday**

Related questions

- NW1 - EDITORIAL
- ABX01 - EDITORIAL
- A problem from December Lunchtime 2 matches with a problem from ACM-IC Dhaka Site Regional...
- editorial for ALGOTHIKA 2012
- Detailed comments in editorial solutions tester and/or author
- When will the editorials for june 17 long out??
- Suggestion : Official Video editorials
- Editorials of Alkhwarizm 2012
- doubt in COUNTREL
- Need help with Solution,of chefuba from l contest

Reason this works, is that suppose we have A = 15, N = 2.

$f((9+6)^2) = f(6^2 + 9*(9+12)) = f(36 + 9*21)$

Now, notice that the term 9*21 will never affect the value of our function.

For further proof, Calculate $f(9*X+Y)$ for some values of X and Y. The answer will always be Y.

Link to my code

Too Much Sweetness

Difficulty: Easy-medium

Prerequisites: DP

Problem

Given two bags of sweets with p and q sweets respectively, Find out number of ways of eating **Exactly N** sweets, subject to conditions that

1. ith bag cannot be opened more than Bi time.
2. We cannot eat more than Si sweets in one step from ith bag.
3. We need to choose bags alternatively.
4. First bag can be either one.

Solution;

Usually for counting problems where we need to make choices and count the number of ways of some sequence, we use DP, taking all the variables affecting answer in dp state.

DP state for this problem has been defined as

1. N = Number of sweets to be eaten.
2. p = Number of sweets in Bag 1
3. q = Number of sweets in Bag 2
4. B1 = Number of times we can open Bag 1
5. B2 = Number of times we can open Bag 2

Base Case if(N == 0) return 1; if (N < 0,)return 0.

DP recurrence

if(last == 1)dp[N][p][q][B1][B2][last] = Summation from i = 1 to i = min(q,S2)solve(N-i, p,q-i,B1,B2-1, 2);

if(last == 2)dp[N][p][q][B1][B2][last] = Summation from i = 1 to i = min(p,S1)solve(N-i, p-i,q,B1-1,B2, 1);

Now, see what;s going on.

In each recursive call, we eat i sweets from bag other than the last one, Reducing sweets to be eaten, also reducing Number of sweets in that bag, and reducing the max number of times of opening respective bag by one, Thus, meeting all conditions.

Also, we run loop from 1 to min(p, S1) and 1 to min(q,S2), to comply with the condition of max sweets at a single step.

Also, use map in place of arrays which may give TLE. we do not visit all states.

Further We can also reduce state variables from 6 to 5 by noticing that N = P - p + Q - q where P and Q are input values while p and q is the current state values. But is didn't use it ad still it works. :)

Here's the link to my code.

Hope you all like my editorials.

On an important note, For further contests, One of my friend (@soham1234) has agreed to write unofficial editorials similar to mine. I hope you would like them too as much as you all do like mine.

ltime55 editorials nw1 abx01

asked **30 Dec '17, 23:13**

6★ taran_1407
[2.7k]●8●27
accept rate: 23%

17 Answers: oldest answers newest answers **popular answers**

10

I have explained by Combinatorics approach briefly for Too Much Sweetness below:

Let us say our sequence formed is x1y1x2y2.....xkyk, where xi is the number of sweets taken out from first box in the ith turn, and similarly for yi. Notice here there are three other cases which can be similarly dealt with:

1) y1x1y2x2....ykxk 2) x1y1x2y2.....yk-1xk 3) y1x1y2x2....xk-1yk

Now we need to find the number of solutions to the equation for each k (max limit depending upon B1 and B2).

x1 + y1 + x2 + y2 + ... + xk + yk = n

=> (x1 + x2 +.... +xk) = n1 and (y1 + y2 +... + yk) = n - n1 such that xi ranges from 1 to S1 and yi ranges from 1 to S2.

This can be done using Multinomial and Inclusion and Exclusion. How?

Let us consider x1 + x2 + ... + xk = n1, x1 can range from 1 to S1. Now, take a substitution zi = xi - 1

So, z1 + z2 + ... + zk = n1 - k, zi can range from 0 to S1 - 1(both inclusive)

Let us say there was no upper bound then by Multinomial we have number of solutions as NcR(n1 - k + k - 1, k - 1). But there are some other solutions in which zi >= S1. So that can be removed from the answer by taking similar substitutions and Inclusion - Exclusion.

Code: https://www.codechef.com/viewsolution/16718580

link | award points edited **31 Dec '17, 01:54** answered **31 Dec '17, 01:45**

5★ shubhiks
[401]●1●6
accept rate: 0%

1 Can you explain how you're excluding the solutions with $z_i \geq S_i$? Is it possible to do that by fixing some i elements such that $S_i \leq z_i \leq p$ and excluding those solutions?

2★ wittyceaser (31 Dec '17, 09:56)

2 Too much Sweetness can be solved using basic Combinatorics as well! Just consider the cases of length of sequence and find out the number of the solutions of the equation formed using Multinomial and Inclusion - Exclusion!

link | award points answered **31 Dec '17, 00:25**

5★ shubhiks
[401]●1●6
accept rate: 0%

Didn't expect a Combinatorics solution for this problem.

6★ taran_1407 (31 Dec '17, 00:32)

Yeah,I too didn't expect a combinatorics solution. Infact I think DP solution is quite simple.It was an overkill I suppose :p .

4★ abx_2109 (31 Dec '17, 00:42)

Can you explain the math behind your solution? @shubhiks

6★ taran_1407 (31 Dec '17, 00:45)

I explained it as a reply as it wasn't fitting in a comment :v

5★ shubhiks (31 Dec '17, 01:46)

2 Actually we can reduce it to 50^3 with 50^3 states using arrray (instead of map). The state must be like 1. p left
2. q left
3. block number
4. flag to indicate either first or second.

And now naively calculating each state takes O(50) time. But in a clever way computing prefix sums over previous level it can changed to O(1) since the dependency states are contiguous.I hope its clear. Caveats in your states: B2 is redundant if B1 is there since it was required for them to be alternate.

link | award points answered **31 Dec '17, 01:28**

6★ teja349
[20]●2
accept rate: 0%

I know there can be many optimizations, but i happened to keep my solution simple.

And by the way, your statement proves that only those certain states will be visited where for example N = P-p+Q-q. My solution, technically won't visit such states.

Yeah, your optimization is really helpful for reduction of dp table size, but i happened to use map, which solved my problems.

6★ taran_1407 (31 Dec '17, 01:33)

Very nice. But this means that the time limit is very very lenient, I wonder why.

5★ meooow ♦ (31 Dec '17, 01:34)

@meooow, my solution would be costly had i used dp table. But by using map, i didn't visit those states. Therefore, there might be only slight difference in runtime i guess.

6★ taran_1407 (31 Dec '17, 01:40)

@taran_1407 it's not about that, my comment was for @teja349. By his optimizations, the complexity comes to $\mathcal{O}(p^3)$. Actually it can be reduced further to $\mathcal{O}(p^2 + N^2)$ by considering box 1 and 2 separately as I have done in my solution (not optimal). So in the worst case iterations required is in the order of 10^4 only. Whereas there were 10 test cases with a time limit of 3 sec!

5★ meooow ♦ (31 Dec '17, 01:49)

Oh...

I thought you was saying that about my solution.

And Yeah, i too wondered, because when i read problem statement, i expected p+q upto 1000.

6★ taran_1407 (31 Dec '17, 01:51)

showing 5 of 10 show all

1 Thanx!! Very nice editorial and congracts for 6*.

I think we could solve the 4th problem using persistent segment tree.Lets say we have 7 of them for each of the combination (001-111) where 0 represents not divisible ,and 1 represents divisible by 2,3,5 respectively.Now lets say for a node v i have a segment tree for a combination(lets say 001 - divisible by 5),so if we store the sum of valid edge cost at a node ,lets say there are edges of cost 5,5,3 from this node to root so segment[5:5]=10,segment[3:3]=0. This segment tree could be constructed from parent.So ansFromRootToNodeV can be obtained by simply searching for the range x:y in all combination of segment tree and adding it,where x,y can easily be calculated(as it is intersection ie for 011 it is intersection of x2,y2 and x3,y3). So ans for u to v is ansFromRootToNodeu+ansFromRootToNodev-2*ansFromRootToNodeLCA(u,v)

link | award points answered **31 Dec '17, 00:12**

5★ vivek_1998299
[224]●5
accept rate: 17%

I have never implemented Persistent Segment Tree. So, it would be better if you could either share your approach in detail, possibly accompanied by implementation if possible.

PS: Thanks

6★ taran_1407 (31 Dec '17, 00:17)

Sorry i too couldn't implement it due to lack of time,if u wanna read it refer this: https://blog.anudeep2011.com/persistent-segment-trees-explained-with-spoj-problems/

5★ vivek_1998299 (31 Dec '17, 00:20)

To understand it more clearly,u could assume that there are N segment trees for each combination (1 for each node) where N is the number of nodes in tree which store the valid edges along the path from root to that node.Persistent just helps to use previous things (as u could see for a node v the segment tree would be the same as its parent except that only edge[v][parent] is not included in parents segment)

5★ vivek_1998299 (31 Dec '17, 00:36)

1

4th problem can be solved using centroid decomposition. Let L be the lca of U and V in the centroid tree. So the path between U and V can be broken down into two parts, U to L and L to V. Also we know that the centroid tree has at most log(n) depth, so each node has atmost log(n) ancestors. We store information about the path between U and ancestors of U(say A). We store the cost of each edge between U and A in a vector v[U][A] and sort them. Then using binary search and prefix sums we can process each query in log(n). Expected Time Complexity: *log(n) per query with nlog(n) preprocessing.*

link | award points

answered **31 Dec '17, 02:24**

5★ aviroop123
[11]●1
accept rate: 0%

Well, i too tried centroid decomposition but couldn't formulate the idea properly.

Would be a lot better if you could share your implementation with us.

6★ taran_1407 (31 Dec '17, 10:18)

1

Thanks for the editorial. I have a couple comments:

Strange Function

We can simplify a bit further if we need to. We observe that $f(A)$ simply maps A to its equivalence class modulo 9 (just like $A \mod 9$, except multiples of 9 get mapped to 9 instead of 0). This is the key to the grade-school "divisibility by 9" check, and it is the reason why $f(A^N) = f(f(A)^N)$. However, we can simplify even further.

- If $f(A) = 9$, then $f(A^N) = 9$ for all values of N.
- If $f(A) \in \{3, 6\}$ and $N > 1$, then $f(A^N) = 9$, since A^N will be divisible by 9.
- The remaining remainders $\{1, 2, 4, 5, 7, 8\}$ form a cyclic-group under multiplication of order 6. Thus, if we are not in the first two categories above, we can instead calculate $f(f(A)^{(N\%6)})$.

TL;DR,

- If A is divisible by 3 and $N > 1$, print 9
- Otherwise, just calculate $((A \mod 9)^{(N \mod 6)}) \mod 9$. If the answer is 0, print 9, otherwise, just print that result.

Too Much Sweetness

(Full disclosure: my programming is rusty, do I didn't get AC during the contest and only later in the afternoon (after debugging silly mistakes))

I used what I thought was perhaps a slightly more efficient approach than the one described. I also like it because the first half is a pretty generic piece of code that can possibly be used elsewhere. I broke the problem up into two parts:

- I used DP and precomputed the ways to split up n objects into k non-empty partitions where each partition was no bigger than s. I stored these in a 3-D array (i.e. ways[n][k][s]).
- I then used these to solve each query with a nested for loop per query (looping over the number of candies of type 1 and then the number of trips to the first box). After fixing these two terms, the number of candies of type 2 is determined, and there are only 3 possibilities for the number of trips to the 2nd box.

The thing I liked about this was that we could reuse the same DP result for both types of candies which kept the size of the DP down. The code seemed to execute several orders of magnitude faster than the time limit.

link | award points

edited **31 Dec '17, 07:00**

answered **31 Dec '17, 06:56**

4★ m1sterzer0
[101]●5
accept rate: 0%

Yes, ur solution for strange function is very neat and simple. But i believe its and overkill here. My approach can also be applied where F(X) is "strangely" defined, say any unique property.

I liked ur solution of too much sweetness, but it requires good knowledge of combinatorics which even i dont possess. Ur solution has better time complexity, but mine is widely applicable. :)

6★ taran_1407 (31 Dec '17, 10:24)

1

@taran_1407 'Strange Function' can be solved with O(1) with a bit of precomputation like @bhpra said. Basically, we need to find cycle for each value from 1 to 9 inclusive.

Code

EDIT - Yes it's complexity O(log(log(...(n))..)

link | award points

edited **31 Dec '17, 20:28**

answered **31 Dec '17, 11:09**

4★ dushsingh1995
[742]●8
accept rate: 11%

1

@taran_1407 Only two observations were to be made to solve the second question in O(1) time for each query.

1. F(A^N) = F(F(A)^N)
2. If the sum of digits of A is 1 or 9, the answer will always be the number itself. For 3 & 6, if N = 1, it will be the number itself. Else if N > 1, it will always be 9.
3. For the remaining numbers 2, 4, 5, 7 & 8 we just needed to find the cycle after how many multiplication the sum of digits will start repeating.

Here is the link to my solution which runs in constant time for all the subtasks.

link | award points

answered **31 Dec '17, 12:53**

3★ vatsalsura
[153]●7
accept rate: 9%

Nice one.. Though an overkill..

6★ taran_1407 (31 Dec '17, 18:40)

0

i have done second question in O(1) link:-https://www.codechef.com/viewsolution/16717619

link | award points

edited **31 Dec '17, 00:38**

answered **31 Dec '17, 00:35**

3★ bhpra
[88]●6
accept rate: 6%

Can you explain the logic behind this?

6★ taran_1407 (31 Dec '17, 00:41)

@bhpra its not O(1) I guess it is O(logn) .

beginner_1111 (31 Dec '17, 01:08)

@beginner_1111 can you explain how

3★ bhpra (31 Dec '17, 01:17)

$F(A^n) = F((F(A))^n)$ so i calculated F(a) now only possible values of a are from 0 to 9 if it is 0 or 1 the $f(a)^n$ will be only 0 and 1 respectively. if it is 3 ,6 or 9 the if n==1 then it will be 3,6,9 respectively else it will be 9. for 2 and 5 it repeat the patter of sum. the pattern of sum for the power's of 2 is {2,4,8,7,5,1} ans for and sum pattern 5 is {5,7,8,4,2,1} ans similar for 7 ,8,4

3★ bhpra (31 Dec '17, 01:44)

What is the complexity for F(n) ?

beginner_1111 (31 Dec '17, 02:18)

showing 5 of 6 show all

0

For the second question, i used the exact same code of mod exp from geeksforgeeks and got an AC :D

Link to my solution.

link | award points

answered **31 Dec '17, 00:36**

3★ chant_coder
[61]●4
accept rate: 0%

1 I too tried this, but forgot to consider ans == 9 case and got many WAs.

6★ taran_1407 (31 Dec '17, 00:40)

Great editorials btw! :D

3★ chant_coder (31 Dec '17, 00:46)

Thanks @chant_coder

6★ taran_1407 (31 Dec '17, 00:55)

0

what is the time-complexity of your dp solution?

link | award points

answered **31 Dec '17, 01:00**

4★ jonsnow7
[1]
accept rate: 0%

To be frank, i didn't think about time complexity, but implemented it because i had an intuition that this will be accepted.

Theoretically, worst case complexity should be around 50^4, but i don't think this limit is actually achieved in any test case.

6★ taran_1407 (31 Dec '17, 01:13)

0

Thank you for the lovely editorials! I too tried to memoize my recursive solution for the 3rd question, but got AC verdict only for the 1st subtask. The others were TLE and WA. Looking at your solution, I don't find much difference in my approach. Maybe I missed a trick somewhere.Could anyone help me find the error in my code?

Here's a link to my submission https://www.codechef.com/viewsolution/16719858

link | award points

answered **31 Dec '17, 02:31**

5★ sherewillpower
[30]●3
accept rate: 0%

1 I cannot surely say, but i suspect ur hash function to be faulty. Consider ur has function for 2 pair of values (1,111) and(11, 11) both will have hash value "1111".

Conclusion: your hash function is poor.

6★ taran_1407 (31 Dec '17, 10:42)

Thanks man! Understood where I went wrong. Changed the hash function to b1+201*(b2+201*(p+201*(q)))+last; Using an unordered hash map and a hash of numbers instead of strings removes the TLE. The new hash function removes the WA. Got AC! Only if this has dawned on me during the contest time :\

Could you give further insights as to how you chose the hash function like this? A hash function of p+201*(q+201*(b1+201*(b2)))+last; fails even for the sample test case. What's the difference?

5★ sherewillpower (31 Dec '17, 12:54)

You didn't multiply p+201(q+201(b1+201*(b2))) by 2 before adding last. This causes collisions in your hash values. There must be atleast two different tuple (p,q,b1,b2,last) which give same hash value.

6★ taran_1407 (31 Dec '17, 18:42)

0

Thank you for the "Too Much Sweet" Editorial.

key = (B2 + 201*(B1 + 201*(q + 201*(p+201*(N)))))*2 + prev%2)

This line of your code got me into thinking if there would be any pair of tuples (tuple1 = (B2,B1,q,p,N,prev) and

tuple2 = (B2',B1',q',p',N',prev')) will collide i.e. if tuple1 and tuple2 under any condition will yield the same key?

Also How did you came up with very hash formula ? Is there any specific norm ?

link | award points

answered **31 Dec '17, 04:11**

5★ rohan_bose95
[543]●1●4
accept rate: 9%

1

I checked the constraints of p, q, B1 and B2 and choose 201. To get a better idea, consider a grid with n rows and M columns. Each cell can be uniquely represented as i*M+j where i and j are row number and column number respectively.(0<= i <N, 0<=j<M).

I used same hash for multiple variables.

6★ taran_1407 (31 Dec '17, 10:21)

Wow. the concept is really is good. I used to struggle a lot with these DP problems.

It would be really helpful if you can link few more problem of the similar type.

5★ rohan_bose95 (31 Dec '17, 15:14)

I remember reading an editorial written by @likecs where he had hashed a bitmask, and grid coordinates this way.

6★ taran_1407 (31 Dec '17, 18:40)

0

I need some help in question "Too much sweetness". I have implemented it in similar way as explained in editorial, but getting WA even for subtask#1.

My code is: https://www.codechef.com/viewsolution/16725382

Thanks in advance!!

link | award points

answered **31 Dec '17, 14:49**

4★ sukhbir947
[1]
accept rate: 0%

0

@taran_1407: can u plzz explain this briefly

we can directly compute f((f(A))N). Doing this avoids the issue of overflow.

link | award points

answered **2 days ago**

3★ coder_ishmeet
[21]●4
accept rate: 0%

0

@sukhbir947 your logic and approach to the problem is wrong... it will give wrong answers in some test cases like these...

1 2 3 5 2 2 1 1

please dry run it and then check your logic.. :D

link | award points

answered **2 days ago**

4★ shreybatra
[1]●1
accept rate: 0%

plz format the test case accordingly...

4★ shreybatra (2 days ago)

0

can someone please explain me the O(50^3) solution of 3rd problem. I had read the comments above but didn't get it! please help.

link | award points

answered **2 days ago**

2★ pk301
[295]●7
accept rate: 15%

You may refer this solution. https://www.codechef.com/viewsolution/16730334

I know this is not the best implementation and probably complex than other ones mentioned above, but i had implemented it this way, so i shared. This one use dp table instead of map, thus doesn't use hash.

6★ taran_1407 (2 days ago)

thanks but currently i don't know java. So, if possible please explain the dp states.

2★ pk301 (2 days ago)

P and Q are number of sweets left in bag 1 and 2 respectively, B mean number of times box can be opened, and cur is the current box to be opened.

count0 and count1 has only one difference, see the recursive call, count0 calls with B-1 when cur is 1 while count1 calls with B-1 when cur is 0.

6★ taran_1407 (2 days ago)

isn't it O(50^4) because you are looping for building each dp states!

2★ pk301 (2 days ago)

Your answer

[hide preview] community wiki:

Preview

Type the text

Privacy & Terms

Answer the question