

Shakil Ahmed's Blog

Wednesday, June 24, 2015

Light OJ (DP part - 2)

এইটা এই সিরিজের দ্বিতীয় লিখা। এই পর্বে কিছু interesting problem এর solution process নিয়ে লিখার চেষ্টা করতছি।

Lighted Panels :::

এই প্রবলেম এ আমাদের একটা লাইট প্যানেল এর অবস্থা দেওয়া হয়েছে, কোন অবস্থার '*' এর মানে হচ্ছে লাইট জ্বলে আছে, '.' মানে হচ্ছে লাইট নিভে আছে। আমাদেরকে মিনিমাম মুভে প্যানেল এর সবগুলো লাইট জ্বালাতে হবে। এইখানে যখন কোন পয়েন্ট toggle করা হয় (মানে এইটা যে অবস্থায় আছে জ্বলা থাকলে নিভা, নিভা থাকলে জ্বলে উঠবে) ঐ পয়েন্ট এর সাথে adjacent যে পয়েন্টগুলো থাকে (diagonal সহ) তারাও toggle করবে।

- যে কোন প্রবলেম solution process বের করার একটা way হচ্ছে constrain গুলো চেক করা। শুধু মাত্র ইনপুট এর লিমিট দেখেই আমরা কোন সল্যুশন প্রসেস চিন্তা করা শুরু করব আবার কোন প্রসেস চিন্তা করা বাদ দিব। এইখানে Row ≤ 8 and Column ≤ 8 দেওয়া আছে। স্বাভাবিক ভাবেই লিমিট কম হলে আমাদের জন্য ভাল। আমরা যদি dp solution develop করতে চাই তাহলে মনের খুশী মত state ও বাড়াইয়া ফেলতে পারব যেটা লিমিট অনেক বড় হইলে সম্ভব হইত না।
- এই প্রবলেমটা যদি দেখি আমরা যখন কোন পয়েন্ট toggle করি তাহলে কি হবে আমরা যে row তে আছি তার আগের row এবং পরের row এর আমরা যে column এ আছি তার আগের column এবং পরের column নিয়ে কাজ করব। অর্থাৎ আমরা যদি row wise করে লাইটগুলো জ্বালাতে থাকি তাহলে আমাদের main concern হইল আগের column, এখন যে column এ আসি এ column এবং পরের column নিয়ে। একই জিনিস column wise করে এগানোর জন্য প্রয়োজ্য। তবে আমরা row wise solution process টা দেখব।
- আমরা আমাদের solution process develop করব row wise, একটা জিনিস দেখি আমরা যখন যে row নিয়ে কাজ করছি এই row এর information তার আগের row এবং পরের row এর উপর নির্ভর করে। তাই কোনভাবে আমি এখন যে row তে আসি তার কিঅবস্থা (মানে কোন কোন পয়েন্ট জ্বলে আসে কি নিভে আসে) যে row থেকে আসলাম তার কি অবস্থা ছিল এবং যে row তে যাব তার কি অবস্থা আছে আমাদের তা জানা থাকা দরকার।
- এই প্রবলেম এর একটা ভাল দিক ছিল এইখানে লিমিট অনেক কম। লিমিট কম থাকলে আমরা অনেক state নিয়ে ভাবতে পারি খুব একটা difference হয় না। যেহেতু আমরা row wise কাজ করতছি অবশ্যই আমরা কোন row তে আসি তা একটা state, আমরা যে row থেকে আসলাম তার কি অবস্থা (কোন কিছু যার মাধ্যমে আমরা বুঝতে পারি কোন কোন পজিশনের লাইট কি অবস্থায় আছে), এবং এখন যে অবস্থায় আসি তার কি অবস্থা আমাদের জানা থাকতে হবে। মানে এই ইনফরমেশনগুলো আমাদের state এ থাকা লাগবে। এখন কোন লাইট জ্বলে আসে কি নিভে আসে তা আমরা খুব সহজেই বিট দিয়ে বুঝতে পারি, যেহেতু column highest হতে পারে ৮টা তাই আমাদের শুধুমাত্র $(1 \ll 8)$ সাইজের কোন state থাকলেই আমরা বুঝে যাব কোন কোন পজিশনে কি কি আছে। তাই আমাদের dp table হবে। $dp[\text{number_of_row}][\text{bit_position_of_previous_row}][\text{bit_position_of_current_row}]$ অর্থাৎ $dp[8][(1 \ll 8)][(1 \ll 8)]$ । আসলে তাহলে আমাদের লাগতেছে $8 * (1 \ll 8) * (1 \ll 8) == 524288$ size এর dp array।
- আমাদের ক্যালকুলেশন এর জন্য আমাদের পরের row ও দরকার হবে এইটা আমরা কোথায় পাব। আমরা প্রথমেই ইনপুট নেওয়ার সময় একটা array তে রেখে দিতে পারি কোন কোন লাইট এখন জ্বলে আসে।
- যে row তে আসি এর প্রতিটা combination করে আমরা আমাদের store value গুলো change করে দেখব। এর জন্য আমরা subset mask use করতে পারি। যেহেতু column এর highest limit ৮ তাই $(1 \ll 8) == 256$ খুব সহজেই আমরা আমাদের dp এর ভিতর তা চালাতে পারি।
- যখন আমরা নেক্সট row তে যাব যদি না আমরা প্রথম কলামে থাকি তাহলে অবশ্যই আমাদের sure করতে হবে আমাদের আগের row এর সবগুলো লাইট জ্বালানো আছে (

Followers

অনুসরণকারীরা (152) পরবর্তী



অনুসরণ

Blog Archive

- ▶ 2013 (2)
- ▶ 2014 (6)
- ▼ 2015 (13)
 - ▶ May (1)
 - ▼ June (8)
 - two pointer

Light OJ DP (part - 1)

যদি না থাকে তাহলে আর কোন ভাবেই ঐ লাইটকে আর জ্বালানো সম্ভব হবে না কিন্তু প্রথম রো থেকে এই জন্যই যাব কারণ আমি দ্বিতীয় রো থেকেও প্রথম রো এর সব লাইট নিভাতে পারি ।) ।

কোড :

```

1  const int INF = 1 << 29 ;
2  int vis[9][ ( 1 << 8 ) + 10 ][ ( 1 << 8 ) + 10 ] , cs ;
3  int dp[9][ ( 1 << 8 ) + 10 ][ ( 1 << 8 ) + 10 ];
4  int sv[9] , r , c ;
5  char str[10];
6  int DP( int idx , int curmask , int prvmask )
7  {
8      if( idx >= r )
9      {
10         if( prvmask == ( 1 << c ) - 1 ) return 0 ;
11         else return INF ;
12     }
13     int &v = vis[idx][curmask][prvmask];
14     int &ret = dp[idx][curmask][prvmask];
15     if( v == cs ) return ret ;
16     // printf(" here \n");
17     ret = INF ;
18     for( int i = 0 ; i < ( 1 << c ) ; i++ )
19     {
20         int cnt = 0 ;
21         int row[3] = {prvmask , curmask , sv[idx+1] }; // আগে রো , এখন যে
22         for ( int j = 0 ; j < c ; j++ ) // সাবসেট মাস্ক দিয়ে all combination
23         {
24             if( !(i & ( 1 << j )) ) continue ; // no need for toggle
25             // need to toggle
26             cnt++;
27             rep( k , 3 ) row[k] ^= ( 1 << j );
28             if( j + 1 < c ) rep ( k , 3 ) row[k] ^= ( 1 << ( j + 1 ) );
29             if( j - 1 >= 0 ) rep ( k , 3 ) row[k] ^= ( 1 << ( j - 1 ) );
30         }
31         if( idx == 0 ) ret = min( ret , cnt + DP( idx + 1 , row[2] , row[1]
32         else if( row[0] == ( 1 << c ) - 1 ) // only if there is all light op
33         ret = min( ret , cnt + DP( idx + 1 , row[2] , row[1] ) );
34     }
35     return ret ;
36 }
37
38 int main()
39 {
40     int t = II ;
41     for ( cs = 1 ; cs <= t ; cs++ )
42     {
43         r = II , c = II ;
44         ms( sv , 0 );
45         rep( i , r )
46         {
47             scanf("%s",str);
48             int mask = 0 ;
49             rep( j , c )
50             {
51                 if( str[j] == '*' ) mask |= ( 1 << j );
52             }
53             sv[i] = mask ; // save the current condition

```

[BFS/DFS part - 1](#)

[Minimum Expression](#)

[Segment tree/ BIT part - 1](#)

[Light OJ \(DP part - 2 \)](#)

[Z Algorithm](#)

[Greedy Part - 2](#)

► [August \(3\)](#)

► [September \(1\)](#)

► [2016 \(12\)](#)

► [2017 \(1\)](#)

► [2019 \(1\)](#)

Popular Posts

Search This Blog

[Light OJ DP \(part - 1 \)](#)

এই জিনিসটা নিয়ে লিখার ইচ্ছা অনেক দিনের । কিছু তেমন জানি না বলে সাহস করে লিখা হয় নাই । লাইট ওজিতে অনেক ইউনিক আইডি এর অনেক ভাল প্রবলেম আছে ।...

[প্রোগ্রামিং কনটেস্ট , হতাশা এবং আমি](#)
কয়দিন আগে ফেসবুক এ ফান থেকে সারাহ তে একাউন্ট খুলেছিলাম । সেখানে যতগুলো প্রশ্ন বা মন্তব্য পাইছি তার ২০% এর মত ছিল প্রোগ্রামিং কনটেস্ট নিয়ে স...

[Digit Dp](#)

Digit Dp এইখানে নামের সাথে এর কাজে এর মিল আছে , যখন কোন র‍্যাঞ্জের নাম্বারের মধ্যে পার ডিজিট নিয়ে কাজ করতে হয় , যেমন আমাদের একটা নাম্বারের ...

[two pointer](#)

বিশেষ করে codeforces এর অনেক প্রবলেম এর ট্যাগে দেখা যায় " two pointer" ট্যাগ করা আছে । স্বাভাবিক ভাবেই যেহেতু পয়েন্টার কথাটা আ...

[Binary Search part - 1](#)

বাইনারি সার্চ কি ? বাইনারি সার্চ হচ্ছে একটা sorted array তে কোন Key value (যেটা আমি খুঁজে বের করতে চাচ্ছি) এর position বের করা । অধ...

[DP on Tree](#)

```

54     }
55     int ans = DP( 0 , sv[0] , 0 );
56     if( ans == INF ) printf("Case %d: impossible\n",cs);
57     else printf("Case %d: %d\n",cs,ans);
58 }
59
60 return 0;
61 }
62

```

dp1.cpp hosted with ❤ by GitHub [view raw](#)

এই প্রবলেমটা আমরা iterative ভাবেও করতে পারি , subset mask এর মাধ্যমে ।

Tiles (II) :

এইটা একটু বিরক্তি কর bitmask প্রবলেম । আমাদের ছয়টা বিভিন্ন সাইজের টাইলস দেওয়া আছে । আমাদের বলতে হবে এই বিভিন্ন টাইলজ দিয়ে (n x m) সাইজের একটা গ্রীড কতভাবে পূরণ করা যাবে । দুইটা বোর্ড different হবে যদি এদের মধ্যে কোন cell এর কালার different হয় ।

- ইনপুট লিমিট এ দেওয়া দেওয়া আছে ($1 \leq n, m \leq 100$ but $\min(n, m) \leq 8$) । এর মানে হইল row , column এর মধ্য যে কোন একটা 8 এর চেয়ে কম হবে । এইটার একটা ব্যাপার হইল আমরা চাইলে এই কম পার্টটাকে বিট করে ফেলতে পারি । এইখানে Row/Column দুইটার যে কোনটাই হইতে পারে । যেহেতু আমরা জানি না কোনটা হবে আমরা ধরে নেই column এর পার্টটাতে আমরা বিটমাস্ক করব কিন্তু Row ও হইতে পারে । যদি Row ≤ 8 হয় তাহলে আমরা given array এর Row এবং Column part টা swap করে দিব । মানে যা input এ ছিল আমাদের column এইটা কে আমরা Row ভ্যালু করে ফেলব । মানে যদি Row = 8 , Column = 100 হয় তাহলে আমাদের Row হয়ে যাবে ১০০ এবং column হয়ে যাবে ৮ ।
- এই প্রবলেমটার সাথে আগের প্রবলেম এর মিল আছে । আমরা যদি given tiles গুলো দেখি প্রতিটাই দুইটা row করে নিয়ে হচ্ছে । অর্থাৎ কোন tiles বসবে কি বসবে না এই জন্য আমাদের কাছে কমপক্ষে দুইটা row এর ইনফরমেশন থাকতে যাবে ।
- আমরা যখন tiles বসাতে যাব আমাদের দেখতে হবে আমরা এখন যে Row তে আসি তার পরের Row এর যে সব column নিতে টাইলসটা হবে তা খালি আছে কিনা ।
- আমরা column ফিল করতে করতে next column এ আগাব এবং যেহেতু দুইটা curmask ও nextmask নিয়ে কাজ করছি অবশ্যই আমরা যখন সব Row ফিল করে ফেলব তখন nextmask অবশ্যই ০ থাকবে ।
- এই কাজটা একটু বিরক্তিকর কারণ আমাদের অনেক চেক রাখতে হবে । সবগুলো tiles নিয়ে ফিল করার জিনিসটা আমরা একটা backtrack function এর মাধ্যমে করতে পারি । এইখানে যেহেতু আমরা column এর basis এ ফিল করব (backtrack এ) আর লিমিট যেহেতু কম আমাদের খুব একটা বেশী কল হবে না ।
- এইখানে রেজাল্ট কে $2^{*64} \bmod$ করে দিতে বলা হইছে তাই আমরা unsigned long long int use করব ।

কোড :

```

1 //BISMILLAHIRRAHMANIRRAHIM
2 /*
3 manus tar shopner soman boro
4 all my suceesses are dedicated to my parents
5 Author :: Shakil Ahmed
6 .....AUST_CSE27.....
7 prob   ::
8 Type   ::
9 verdict::
10 */
11 #include <bits/stdc++.h>
12 #define pb push_back
13 #define mp make_pair
14 #define pi acos(-1.0)
15 #define ff first
16 #define ss second
17 #define re return
18 #define QI queue<int>

```

ফেসবুক এ ব্লগ
লিখা বা কনটেন্ট
করার সুবিধার্থে
অনেকের সাথে
কথা হয় ।
অনেকবারই
অনুরোধ ছিল DP
on tree নিয়ে যেন
লিখি । Quora তে
অনেক ভাল একটা
পো...

Greedy Method

Greedy কি ?
প্রথমেই আসা
যাক , greedy কি ?
greedy হল
ভবিষ্যতের এর
কথা চিন্তা না করে
বর্তমান অবস্থা
গুলা বিবেচনা করে
বেস্ট একশনটা
নেও...

Probability & Expected value (part - 1)

probability and
expected value
এর উপর top
coder এবং
codechef এ ভাল
কিছু লিখা হইছে ।
যে কোন কিছু ভাল
করে শিখার একটা
ভাল উপায় হইল
প্রথমে থ...

Light OJ (DP part - 2)

এইটা এই
সিরিজের দ্বিতীয়
লিখা । এই পর্বে
কিছু interesting
problem এর
solution process
নিয়ে লিখার চেষ্টা
করতেছি ।
Lighted Panels
... ..

Programming Interview - Rajon Bardhan

প্রোগ্রামিং ইন্টারভিউ
সিরিজের
আজকের অতিথি
আমাদের
আহসানউল্লাহ সব
থেকে কপাল
থারাপ
কনটেন্টেইন
আবার একই সাথে
সবচেয়ে
ইস্পাহরিং
ক্যারেক্টার
রাজন...

Labels

- [backtrack](#)
- [Bfs](#)
- [Binary Search](#)
- [Bipartite Matching](#)
- [Bit](#)
- [Bitmask](#)
- [coding test](#)
- [Data Structure](#)
- [Dfs](#)
- [Digit Dp](#)
- [DP](#)

```

19 #define SI stack<int>
20 #define SZ(x) ((int) (x).size())
21 #define all(x) (x).begin(), (x).end()
22 #define sqr(x) ((x) * (x))
23 #define ms(a,b) memset((a),(b),sizeof(a))
24 #define G() getchar()
25 #define MAX3(a,b,c) max(a,max(b,c))
26 #define II ( { int a ; read(a) ; a ; } )
27 #define LL ( { Long a ; read(a) ; a ; } )
28 #define DD ({double a; scanf("%lf", &a); a;})
29
30 double const EPS=3e-8;
31 using namespace std;
32
33 #define FI freopen ("input.txt", "r", stdin)
34 #define FO freopen ("output.txt", "w", stdout)
35
36 typedef long long Long;
37 typedef long long int64;
38 typedef unsigned long long ull;
39 typedef vector<int> vi ;
40 typedef set<int> si;
41 typedef vector<Long>vl;
42 typedef pair<int,int>pii;
43 typedef pair<string,int>psi;
44 typedef pair<Long,Long>pll;
45 typedef pair<double,double>pdd;
46 typedef vector<pii> vpil;
47
48 // For loop
49
50 #define forab(i, a, b) for (__typeof (b) i = (a) ; i <= b ; ++i)
51 #define rep(i, n) forab (i, 0, (n) - 1)
52 #define For(i, n) forab (i, 1, n)
53 #define rofba(i, a, b) for (__typeof (b)i = (b) ; i >= a ; --i)
54 #define per(i, n) rofba (i, 0, (n) - 1)
55 #define rof(i, n) rofba (i, 1, n)
56 #define forstl(i, s) for (__typeof ((s).end ()) i = (s).begin (); i != (
57
58 template< class T > T gcd(T a, T b) { return (b != 0 ? gcd<T>(b, a%b) : a);
59 template< class T > T lcm(T a, T b) { return (a / gcd<T>(a, b) * b); }
60
61 //Fast Reader
62 template<class T>inline bool read(T &x){int c=getchar();int sgn=1;while(~c&
63
64 //int dx[]={1,0,-1,0};int dy[]={0,1,0,-1}; //4 Direction
65 //int dx[]={1,1,0,-1,-1,-1,0,1};int dy[]={0,1,1,1,0,-1,-1,-1};//8 direction
66 //int dx[]={2,1,-1,-2,-2,-1,1,2};int dy[]={1,2,2,1,-1,-2,-2,-1};//Knight Di
67 //int dx[]={2,1,-1,-2,-1,1};int dy[]={0,1,1,0,-1,-1}; //Hexagonal Direction
68
69 /* ***** My code start here *****
70 const int NX = ( 1 << 8 ) + 10 ;
71 const int MX = 105 ;
72 ull dp[MX][NX] ;
73 int vis[MX][NX] , cs , row , col ;
74 char inp[MX][MX] , grid[MX][MX] ;
75 ull DP(int , int) ;

```

- [Editorial](#)
- [Expected value](#)
- [Game theory](#)
- [Graph](#)
- [greedy](#)
- [interview](#)
- [Light Oj](#)
- [Minimum Expression](#)
- [Probability](#)
- [Semgent tree](#)
- [String](#)
- [Tree Dp](#)
- [two pointer](#)
- [Union Find](#)
- [Z Algorithm](#)

Translate

Powered by [Google Translate](#)

About Me

[Shakil Ahmed](#)

[View my complete profile](#)

```

76 ull gen( int r , int c , int curmask , int nxtmask )
77 {
78     if( c == col ) return DP( r + 1 , nxtmask );
79     int cc = ( curmask >> c ) & 1 ;
80     int cnc = ( curmask >> ( c + 1 ) ) & 1 ;
81     int nc = ( nxtmask >> ( c ) ) & 1 ;
82     int nnc = ( nxtmask >> ( c + 1 ) ) & 1 ;
83     int npc = ( nxtmask >> max( 0 , c - 1 ) ) & 1 ;
84     if( cc ) return gen( r , c + 1 , curmask , nxtmask );
85     if( grid[r][c] == '#' ) return gen( r , c + 1 , curmask , nxtmask );
86     ull ret = 0 ;
87     // tile one
88     if( r + 1 < row )
89     {
90         if( grid[r][c] == '.' && grid[r+1][c] == '.' && !cc && !nc )
91         {
92             ret += gen( r , c + 1 , curmask | ( 1 << c ) , nxtmask | ( 1 <<
93         }
94     }
95     // tile two
96     if( c + 1 < col )
97     {
98         if( grid[r][c] == '.' && grid[r][c+1] == '.' && !cc && !cnc )
99         {
100             ret += gen( r , c + 2 , curmask | ( 1 << c ) | ( 1 << ( c + 1 )
101         }
102     }
103     // tile five
104     if( c - 1 >= 0 && r + 1 < row )
105     {
106         if( grid[r][c] == '.' && grid[r+1][c] == '.' && grid[r+1][c-1] == '
107         {
108             ret += gen( r , c + 1 , curmask | ( 1 << c ) , nxtmask | ( 1 <<
109         }
110     }
111     if( r + 1 < row && c + 1 < col )
112     {
113         // tile 4
114         if( grid[r][c] == '.' && grid[r][c+1] == '.' && grid[r+1][c] == '.'
115         {
116             ret += gen( r , c + 1 , curmask | ( 1 << c ) | ( 1 << ( c + 1 )
117         }
118         // tile 3
119         if( grid[r][c] == '.' && grid[r+1][c] == '.' && grid[r+1][c+1] ==
120         {
121             ret += gen( r , c + 1 , curmask | ( 1 << c ) , nxtmask | ( 1 <<
122         }
123         // tile 6
124         if( grid[r][c] == '.' && grid[r][c+1] == '.' && grid[r+1][c+1] ==
125         {
126             ret += gen( r , c + 1 , curmask | ( 1 << c ) | ( 1 << ( c + 1 )
127         }
128     }
129 }
130 return ret ;
131
132 }

```

```

133
134 ull DP(int r , int mask)
135 {
136     if( r == row ) return mask == 0 ;
137     ull &ret = dp[r][mask];
138     int &v = vis[r][mask];
139     if( v == cs ) return ret ;
140     v = cs ;
141     ret = gen(r , 0 , mask , 0 );
142     return ret ;
143 }
144
145 int main()
146 {
147     // I will always use scanf and printf
148     // May be i won't be a good programmer but i will be a good human being
149     // ms( vis , -1 );
150
151     int t = II ;
152     for ( cs = 1 ; cs <= t ; cs++ )
153     {
154         row = II , col = II ;
155         rep( i , row ) scanf("%s",inp[i]);
156         if( row < col )
157         {
158             rep( i , row ) rep ( j , col ) grid[j][i] = inp[i][j];
159             swap( row , col );
160         }
161         else
162         {
163             rep( i , row ) rep ( j , col ) grid[i][j] = inp[i][j];
164         }
165         printf("Case %d: %llu\n",cs,DP(0,0));
166     }
167
168
169
170
171
172     return 0;
173 }
174

```

DP2.cpp hosted with ❤ by GitHub

[view raw](#)**Software Company :**

এই প্রবলেমটা binary search + dp এর। এইখানে একটা software company এর কথা বলা হইছে। তারা দুইটা প্রোজেক্ট শেষ করবে। n জন employee এর ইনফর্মেশনে দেওয়া আছে তাদের কত মিনিট করে নেয় দুইটা প্রোজেক্ট এর এক unit কাজ করতে। কোন প্রোজেক্ট সম্পূর্ণ শেষ করতে m unit কাজ কমপ্লিট করতে হবে।

- এই প্রবলেমটা খুব সম্ভব বুঝাটা এইটা binary search এ হবে এইটাই সব থেকে কঠিন পার্ট। তারপরের কাজ খুব সহজ। আমরা যেকোন একটা employee নিয়ে ও যদি একা দুইটা কাজ করত তাহলে কত সময় লাগতকে high ধরে lower_bound binary search করব। দেখব lowest কত value এর জন্য আমাদের কাজ দুইটি সম্পূর্ণ করা যাবে।
- 3 state এর normal dp , current_employee_number , work_left_for_first_one , work_left_for_second_one . এইখানে লিমিট দেওয়া আছে (1 <= n , m <= 100) তাই আমাদের dp[101][101][101] size এর dp দিয়েই হয়ে যাবে।

কোড

```

1  const int NX = 102 ;
2
3  int loop , vis[NX][NX][NX] ;
4  bool dp[NX][NX][NX] ;
5
6  pii inp[NX];
7
8  int low , high , mid , ans , n , m ;
9
10
11 bool DP( int idx , int need1 , int need2 )
12 {
13     if( need1 + need2 == 0 ) return true ;
14     if( idx == n ) return false ;
15     int &v = vis[idx][need1][need2];
16     bool &ret = dp[idx][need1][need2];
17     if( v == loop ) return ret ;
18     v = loop ;
19     ret = false ;
20     int i ;
21     for ( i = 0 ; i <= need1 && i * inp[idx].ff <= mid ; i++ )
22     {
23         int k = ( mid - ( i * inp[idx].ff ) ) / inp[idx].ss ;
24         if( k > need2 ) k = need2 ;
25         if( DP( idx + 1 , need1 - i , need2 - k ) ) return ret = true ;
26     }
27     return ret ;
28 }
29
30
31 int main()
32 {
33     // I will always use scanf and printf
34     // May be i won't be a good programmer but i will be a good human being
35     int cs , t = II ;
36     for ( cs = 1 ; cs <= t ; cs++ )
37     {
38         n = II , m = II ;
39         rep( i , n )
40         {
41             int x = II , y = II ;
42             inp[i] = mp( x , y );
43         }
44         low = 0 , high = inp[0].ff * m + inp[0].ss * m ;
45         while( low <= high )
46         {
47             mid = ( low + high ) / 2 ;
48             loop++;
49             if( DP( 0 , m , m ) )
50             {
51                 ans = mid ;
52                 high = mid - 1 ;
53             }
54             else low = mid + 1 ;
55         }
56         printf("Case %d: %d\n",cs,ans);

```

```
57     }
58
```

dp3.cpp hosted with ❤ by GitHub

[view raw](#)

এই সিরিজের আরো কয়েকটা লিখা ইচ্ছা আছে , যদি লিখার মোটিভেশন বজায় থাকে । ধন্যবাদ পড়ার জন্য ।

Posted by [Shakil Ahmed](#) at 7:00 AMLabels: [DP](#), [Light Oj](#)

8 comments:

[cerealguy](#) September 2, 2015 at 3:00 AM

This is a good stuff. Keep it up man! :)

[Reply](#)[code_blocks](#) September 2, 2015 at 9:45 AM

please discuss how can I solve TUG OF WAR . I've no idea about it

[Reply](#)[Replies](#)[shakil ahmed](#) September 3, 2015 at 1:06 AM

no problem i will in my next DP post . But it may take some times . If you want to solve it now i give you some hints how to solve it . at first we need to concern about two things . as we need to divide them into two teams and difference between two team must not differ by one (thats the little thing that different it from normal 0/1 knapsack part , if this constrain isn't given we can easily do it with limit (total_weight/2) part , hope you understand it .) and difference weight between two team is as little as possible . Total weight won't exceed 100000 some how you need to use this information . Say 1 person weight is 3 kg and another personal weight is 4 kg . We need to use this information that 3kg of weight can be one person weight and 4 kg weight is one person weight and 7kg weight is two person weight in an array . think about it .

[code_blocks](#) September 12, 2015 at 7:52 AM

thanks vaiya :) it was really helpful :) i've solved this one :)

[Reply](#)[code_blocks](#) September 20, 2015 at 7:22 AM

how can i solve Light oj 1415 (save the trees) ? :\ please help in details :\

[Reply](#)[shakil ahmed](#) October 21, 2015 at 5:04 PM

sorry , i haven't seen this comment before . ok i will try to write it down in my next dp blog , It may take time . Right now i am very busy with my stuff . Whenever i get some free times i will write it down .

[Reply](#)[cerealguy](#) November 6, 2015 at 1:26 AM

I see a mistake in line number 24 of the first problem. The condition should be

if(i & (1 << j)), shouldn't it?

[Reply](#)

**shakil ahmed**

November 6, 2015 at 1:36 AM

no , it is right . if the j(th) bit in i is 1 then we gonna toggle this otherwise no . actullay its totally upto you what you wanna do , as only 0/1 is here and all combination is fulfill by 1's combination or o's combination nothing is big deal :D you can chose one of the combination
Say 2bit combination

00

01

10

11

what ever you choose 0's combination to toggle or 1's combination both will give you right answer .

[Reply](#)

Comment as:

Google Accour

[Newer Post](#)[Home](#)[Older Post](#)Subscribe to: [Post Comments \(Atom\)](#)

Shakil ahmed. Simple theme. Powered by [Blogger](#).