



My TopCoder

Competitions

Overview

Copilot Opportunities

Design

Development

UI Development

QA and Maintenance

CloudSpokes

Algorithm

High School

The Digital Run

Submit & Review

TopCoder Networks

Events

Statistics

Tutorials

Forums

Surveys

My TopCoder

Help Center

About TopCoder



Member Search:

 Handle:
[Advanced Search](#)
[Dashboard](#) > [TopCoder Competitions](#) > ... > [Algorithm Problem Set Analysis](#) > [SRM 605](#)



TopCoder Competitions

SRM 605

View

Attachments (0)

Info

[Browse Space](#)
Added by [[rng_58]], last edited by vexorian on Jan 27, 2014 ([view change](#))Labels: (None) [EDIT](#)

Single Round Match 605

Tuesday, January 21st, 2014

Archive

[Match Overview](#)
[Discuss this match](#)

Match summary

More than 1800 coders participated in SRM 605, a problem set by **Witaliy**. A match with interesting ad hoc and dynamic programming problems. In division 1, the coders were greeted by a problem that required coders to be crafty in implementation or spend too much time coding. **semiepx** found a way to code some not-so simple code in just few minutes and got the fastest score. The second problem was the interesting dynamic programming one that required care and analysis to be implemented correctly. **yeputons** got the best score in mere 13 minutes. The hard problem was mostly about decomposing a problem into simple properties and was solved by only 6 coders of which, **semiepx**, got the best score (that's two speed records in a single match). Not content with dominating the problem scores, **semiepx** also got 50 challenge points to magnify the lead over second place: **Nerevar**. Also breaking the 1000 points barrier: **VARtem** got the third place.

The Problems

[AlienAndPassword](#) | [AlienAndGame](#) | [AlienAndSetDiv2](#) | [AlienAndHamburgers](#) | [AlienAndSetDiv1](#) | [AlienAndPermutation](#)

AlienAndSetDiv2

Used as: Division Two - Level Three:

Value	1000
Submission Rate	7 / 1015 (0.69%)
Success Rate	0 / 7 (0.00%)
High Score	null for null points (NONE)
Average Score	No correct submissions

Adding the numbers in some order

Count the number of ways to assign each number from 1 to $2N$, inclusive to one of two sets. The condition requires us to compare the i -th largest integer of each set. Let's consider the two sets as sorted arrays.

The trick consists in deciding the sets of the integers in some order. This explanation uses decreasing, even though increasing order should work too. We add $2N$, then $2N - 1$, $2N - 2$, ... 1.

Consider $N = 4$, $K = 2$. Then we have the following setup:

A=????
B=????

We can now decide where should 8 go. Since the sets will be sorted, there are only two positions for it:

A=???8
B=????

or

A=????
B=???8

Assume we picked the first option. Now add number 7. We have two options:

A=???8
B=???7

or:

A=???8
B=????

Let's pick the second option. We now have to add number 6. Note that the following option is invalid:

A=?678
B=????

Because $K = 2$, if we add 6 to that position, then there will be no number that can share position 3 with number 8. So we must pick:

A=??78
B=???6

Now 5 has two available positions and so and so.

Eventually, we should reach a state in which we added all numbers and all of them are matched. This would be a single correct way to add the numbers. In this counting problem, however, we cannot iterate through them all, we have 100 choices and at most 2 options in each of them, so the complexity is likely too large. We should aim towards a faster solution.

Dynamic programming

If you are new to dynamic programming, try [dumitru's tutorial: Dynamic programming: from novice to advanced](#) from the educational section.

The plan is to turn the previous logic into a dynamic programming solution. We can describe the above strategy as a function $f(n, A, B)$ that counts the number of ways to add the first n integers where:

- n is the number of integers we haven't added yet. (So the next integer we choose a position for is n).
- A represents the *contents* of set A so far.
- B represents the *contents* of set B so far.

A recurrence relation needs a base case. When $n = 0$, there are no more numbers to add. A and B must contain the same number of elements, in that case, the result is 1 as there is one valid way to do it (do nothing). Else, there is no valid way, so the result is 0.

Otherwise, we can solve $f(n, A, B)$ using the logic above. $f(n, A, B)$ is equal to the sum of the results of adding n to A or B depending on whether or not the step is valid.

This recurrence relation solves the problem but we need to optimize it. Currently it is still a brute-force. The trick lies in simplifying the state. Currently there are too many valid pairs of sets A, B for $f(n, A, B)$. There are a couple of things we can do:

Ignore matched integers

Consider the following two states of the problem:

A=?478
B=???56

And:

A=?456
B=??78

Note that their solutions are given by $f(3, A = \{4, 7, 8\}, B = \{5, 6\})$ and $f(3, A = \{4, 5, 6\}, B = \{7, 8\})$. These two cases should actually have the same result. Really, all that matters is how we fill this part:

A=?4
B=??

We can freely ignore the contents of the sets that were already matched.

The symmetry

Consider the two following states of the problem:

A=??78
B=???6

And:

A=???6
B=??78

Note that their solutions are given by $f(5, A = \{7, 8\}, B = \{6\})$ and $f(5, A = \{6\}, B = \{7, 8\})$. These two cases should actually have the same result.

Combined with the previous idea, we can represent the state just by a subset \hat{A} that contains the integers that are still *unmatched*. So the two previous states are represented as:

A=??7
B=???

K won't be large

Remember the case when $N = 4, K = 2$ and $n = 6$, the following assignment was invalid:

A=?678
B=????

Simply because 8 is unmatched, and $K = 2$, so if we don't match it with 6, it will be impossible to match it with any smaller number. So in this case, we should always match 8 with 6. To generalize, if $n + K$ is unmatched, it must be matched with n in this turn. This means that $n + K + 1$ will never be unmatched. So the set of unmatched elements \hat{A} will only contain numbers from $n + 1$ to $n + K$,

inclusive. For each n , there are 2^K different subsets \hat{A} that could be paired with it. This means there are $O(2^K \cdot N)$ different pairs of (n, \hat{A}) . If we use dynamic programming so that each state of the function is evaluated exactly once, the total complexity is $O(2^K \cdot N)$ which is fine, because $K \leq 10, N \leq 50$

The implementation

In this problem, the code part is a challenge of its own. We are supposed to do dynamic programming to a function that has two arguments, one is an integer while the other is a set. One option is to use memoization. We can represent the set as a vector / array / list or even set, and use it as a key to an associative array. E.g: In c++, we can use a [std::map](#), Java has [HashMap](#), [TreeMap](#) or some others; Python has [dictionaries](#). Note that the use of those data structures will add complexity to the algorithm, but the original $O(2^K \cdot N)$ complexity is so low it allows it in C++:

Problems that ask you to return the result modulo 1,000,000,007 (or a similar number) usually do so to allow us to solve problems that have large results without actually using those large numbers in calculations. We can handle these numbers with modular arithmetic. Read [this recipe](#) for more info.

```
const int MOD = 1000000007;
int K;

// A type set so that we don't need to type set<int, greater<int> > all the time
// We use greater instead of the default so that it is easy to grab the maximum element by calling .begin()
using gset = set<int, greater<int> >;

map< gset, long > dp[101];

long rec(int n, gset unmatched)
{
    auto q = dp[n].find(unmatched);
    long res = 0;
    if (q == dp[n].end()) {
        if (n == 0) {
            if (unmatched.size() == 0) {
                res = 1;
            }
        } else {
            if (unmatched.size() == 0) {
                // we can put n in any of the two sets.
                res += ( 2 * rec(n - 1, {n}) ) % MOD;
            } else {
                int mx = *unmatched.begin(); //get the largest number in set
                // match n with mx:
                gset newset = unmatched;
                newset.erase( newset.begin() );
                res += rec(n - 1, newset);

                // add to the other set.
                if ( mx != n + K ) {
                    newset = unmatched;
                    newset.insert(n);
                    res += rec(n - 1, newset);
                }
            }
        }
        dp[n][unmatched] = res;
    }
    return res;
}
```

Or even in python. The following simple code needs 0.2 seconds in the worst case:

```
class AlienAndSetDiv2:
    def getNumber(self, N, K):
        mem = dict()
        def f(n, A):
            if (n,A) not in mem:
                res = 0
                if n == 0:
                    res = 1 if ( A == () ) else 0
                elif A == ():
                    res = 2 * f(n - 1, (n,) )
                else:
                    # match n with maximum (A[0]):
                    res += f(n - 1, A[1:])
                    # add n to A:
                    if A[0] != n + K:
                        res += f(n - 1, A + (n,) )
                mem[(n,A)] = res
            return mem[(n,A)]
        return f( 2*N, () ) % 1000000007
```

Bit masks

Representing sets using bit masks is a trick used commonly during contests. If you are unfamiliar with this, you should give [bmerry's](#) essential tutorial: [A bit of fun: fun with bits](#) a try.

If we want to avoid the extra overhead of using data structures or to arguably simplify the code in some languages, we can use a bit mask to represent the set \hat{A} . This brings some details that we need to solve.

- The set for n contains numbers from $n + 1$ to $n + K$, inclusive. The set of $n - 1$ contains numbers from n to $n + K - 1$, inclusive. If for n we use bit position 0 for element $n + 1$, for $n - 1$, the same element is represented by position 1. This means we need to do a right-shift on the bitmask whenever we move from n to $n - 1$.
- We might need a quick find to find the largest integer currently in the set. We can use a single $O(K)$ loop to test each bit, or we can

use a precalculated table, or the [clz instruction](#)...

```
static const int MOD = 1000000007, MAX_N = 50, MAX_K = 10;
int K;

int dp[2*MAX_N + 1][1<<MAX_K];

long rec(int n, int mask)
{
    long res = dp[n][mask];
    if (res == -1) {
        res = 0;
        if (n == 0) {
            res = (mask == 0) ? 1 : 0;
        } else if (mask == 0) {
            // can put n in either of the sets, it will become element 0
            // in the set for n-1.
            res = 2 * rec(n - 1, 1);
        } else {
            // 31 - clz(mask) is a shorthand to get position of the most
            // significant 1-bit:
            int mx = 31 - __builtin_clz(mask); // could just use a loop..
            // match with largest:
            res = rec(n - 1, (mask ^ (1 << mx)) << 1);
            // * we remove the mx-th bit of mask.
            // * we shift the mask right, because element 0 of set for n
            //   is element 1 in set for n-1, and so and so...

            if (mx != K-1) {
                // add to the above set
                res += rec(n - 1, (mask << 1) | 1);
                // * again, shift the mask right.
                // * add element 0 (for n-1, n is element 0).
            }
            res %= MOD;
        }
        dp[n][mask] = res;
    }
}
```

Alternative solutions and additional comments.

<Place your comments here>

Next problem: [AlienAndHamburgers](#)

Author



By **vexorian**

TopCoder Member

Editorial feedback	Choose
I liked it.	<input checked="" type="checkbox"/>
I didn't like it.	<input checked="" type="checkbox"/>

Comments ([Hide Comments](#))

While the editorial is preliminary, you are welcome to suggest changes and corrections. The editorial will be fully editable once the final problem explanation is ready.

When posting a comment thread, make sure to specify the problem you are talking about.

Posted by vexorian at [Jan 23, 2014 00:29](#) | [Reply To This](#)

AlienAndSetDiv1: "Once a integer..." should be "Once an integer..."

Posted by johnathan79717 at [Jan 23, 2014 01:20](#) | [Reply To This](#)

int the problem AlienAndSetDiv2
int the c++ solution(use map)
is "indent" means nothing(it seems you didnt use it)?
thx~

Posted by ray007great at [Jan 23, 2014 12:59](#) | [Reply To This](#)

It was used for debugging and forgot to remove it.

Posted by vexorian at [Jan 23, 2014 14:29](#) | [Reply To This](#)

For Division Two - Level One (AlienAndPassword), how can qzpm5n, #1 on this level, complete everything in (0 mins 8 secs)? 8

seconds is not enough for me even to complete reading the question statement. Any magic?

 Posted by bjiang78 at [Jan 23, 2014 13:42](#) | [Reply To This](#)

One possibility is the use of multiple accounts. Using one account read the problem and get the solution. Using another account compile and submit it.

 Posted by sushilpandey at [Jan 29, 2014 02:01](#) | [Reply To This](#)

nice solution explaintion!

 Posted by ray007great at [Jan 23, 2014 23:03](#) | [Reply To This](#)

One liner in c++

```
int getNumber(string S) {
return std::unique(S.begin(), S.end()) - S.begin();
}
```

 Posted by jitendra_theta at [Jan 24, 2014 01:57](#) | [Reply To This](#)

please explain it also + the complexity ...

 Posted by kavish_mnnit at [Jan 26, 2014 10:01](#) | [Reply To This](#)

AlienAndSetDiv1

can be solved as $C(2N, N) - \text{AlienAndSetDiv2}(K-1)$



 Posted by vlad_d at [Jan 24, 2014 13:35](#) | [Reply To This](#)

AlienAndSetDiv2

You say that "The set for n contains numbers from n+1 to n+K , inclusive." , That's not true right? Because in your own example, 6 is not present in the set which contains 7 and 8?!

 Posted by vishwasrao at [Jan 24, 2014 15:25](#) | [Reply To This](#)

Interpret as: May contain only numbers from n+1 to n+K.

 Posted by vexorian at [Jan 24, 2014 15:31](#) | [Reply To This](#)

AlienAndGame - $O(W*H)$ Solution

```
int getNumber(vector <string> board)
{
int res = 0, dp[2][64][64];
for (int i = 0; i < board.size(); i++)
for (int j = 0; j < board[i].size(); j++)
dp[0][i][j] = dp[1][i][j] = 0;

for (int i = 0; i < board.size(); i++)
if (i == 0)
{
for (int j = 0; j < board[i].size(); j++)

Unknown macro: { if (board[i][j] == 'W') dp[0][i][j] = 1; else dp[0][i][j] = 0; dp[1][i][j] = !dp[0][i][j]; }

}
else
{
for (int j = 0; j < board[i].size(); j++)
{
for (int k1 = 0; k1 < 2; k1++)
{
int a = j==0?0:dp[k1][i-1][j-1],
b = dp[k1][i-1][j],
c = j==0?0:dp[0][i][j-1],
c2 = j==0?0:dp[1][i][j-1];
if (board[i][j] == 'W')
```

Unknown macro: { dp[0][i][j] = 1 + min(a, min(b, c)); dp[1][i][j] = 0; }

else

Unknown macro: { dp[0][i][j] = 0; dp[1][i][j] = 1 + min(a, min(b, c2)); }

```
}
}
}
```

```
for (int i = 0; i < board.size(); i++)
for (int j = 0; j < board[i].size(); j++)
res = max(res, max(dp[0][i][j], dp[1][i][j]));
return res*res;
}
```

 Posted by thebvog at [Jan 27, 2014 04:08](#) | [Reply To This](#)

Hi , I was trying to code AlienAndGame which passed on sample test cases but failed on system test cases. please let me know if i have a done a logical mistake.

I will call GetMaxSquare(r,c) in the main function

```
int GetMaxSquare(int i,int j,vector<string> &board)
{
if(i == 0 || j == 0)
return 1;

if((board[i][j] == board[i][j-1]) && (board[i-1][j-1] == board[i-1][j]))
```

Unknown macro: { int size = GetMaxSquare(i,j-1,board); size= min(min(size,GetMaxSquare(i-1,j,board)),GetMaxSquare(i-1,j-1,board)); maximum =max(maximum,size+1); return size+1; }

```
else
return 1;
}
finally return maximum * maximum ;
```

 Posted by fusionreborn at [Jan 28, 2014 14:59](#) | [Reply To This](#)

awesome editorial..first time, I read editorial on topcoder. I learned a lot of things.

thanks man 😊

 Posted by anup1pma at [Jan 29, 2014 12:20](#) | [Reply To This](#)

o(wh) solution

```
public : int getNumber(vector<string>b)
{
int m=b.size();
int n=b[0].size();
int a[m][n];
for(int i=0;i<m;i++) a[i][0]=1;
for(int i=0;i<n;i++) a[0][i]=1;
int ans=1;
for(int i=1;i<m;i++)
{
for(int j=1;j<n;j++)
{
if(b[i][j]==b[i][j-1] && b[i-1][j]==b[i-1][j-1])

Unknown macro: { int temp; if(a[i][j]-1==a[i-1][j]&& a[i][j]-1==a[i-1][j-1])temp=sqrt(a[i][j]-1)+1; else temp=sqrt(min(a[i][j]-1,min(a[i-1][j]-1,a[i-1][j])))+1; a[i][j]=temp*temp; }

else a[i][j]=1;
if(a[i][j]>=ans) ans=a[i][j];
// cout<<i<<"\t"<<j<<"\t"<<a[i][j]<<endl;
}
}
return ans;
}
```

 Posted by taree.earth at [Jan 30, 2014 02:38](#) | [Reply To This](#)

I got a $O(K^2 * N)$ solution: in C++

```
typedef long long ll;

using namespace std;
```

```

template<ll mod = 1000000007, ll maxn = 100>
struct NCM
{
private:
ll fac[maxn + 1];
ll inv[maxn + 1];
ll nCm[(maxn + 1) * (maxn + 1)];
public:
NCM()
{
fac[0] = inv[0] = 1LL;
for (int i = 1; i < maxn; ++i)

Unknown macro: { fac[i] = fac[i - 1] * i % mod; inv[i] = powMod(fac[i], mod - 2); }

for(int i = 1; i <= maxn; ++i)
{
for(int j = 0; j <= maxn; ++j)

Unknown macro: { nCm[i * (maxn + 1) + j] = fac[i] * inv[j] % mod * inv[i - j] % mod; }

}
}

ll operator()(int n, int m)

Unknown macro: { return nCm[n * (maxn + 1) + m]; }

private:
ll powMod(ll a, ll b)
{
ll ret = 1LL;
while (b > 0)

Unknown macro: { if ((b & 1) > 0) ret = ret * a % mod; a = a * a % mod; b >>= 1; }

return ret;
}
};

template<ll mode = 1000000007>
struct Multi
{
public:
ll operator()(ll a, ll b)
{
int ret=0;
for(a%=mode,b%=mode; b > 0; a =(a<<1)%mode,b>>=1)
{
if (b&1)

Unknown macro: { ret = (ret + a)%mode; }

}
return ret;
}
};

class AlienAndSetDiv2
{
public:
int getNumber(int N, int K)
{
static Multi<> multi;
static NCM<> nCm;

int C[51][51] =

Unknown macro: {0}

; // C[N][K], Counting permutation with |Ai - Bi| equals K (== only, < excluded).
int S[51][51] =

; // S[N][K] = C[N][1] + C[N][2] + ... + C[N][K]
memset(C, 0, 51 * 51 * sizeof(int));
memset(S, 0, 51 * 51 * sizeof(int));

// Think about S[N][K] (N = 1, 2, ... ) with N == K, the permutations with |Ai - Bi| <= K will be either of the below two forms:
// 1 a2 a3 ... N , 1 < a2 < a3 < ... < N
// b1 b2 b3 ... bn , b1 < b2 < ... < bN
// or
// 1 a2 a3 ... an, 1 < a2 < a3 < an
// b1 b2 b3 ... N, b1 < b2 < b3 < bn.
// and please note the set A and B could be switched to each other,

```

```
// so there's a formula to calculate the total number of permutations with |Ai - Bi| <= K
// S[N][N] = 2 * (nCm(2 * N - 2, N - 1) + nCm(2 * N - 2, N)).

for(int n = 1; n <= N; ++n)
{
    if(n == 1)

        Unknown macro: { S[n][n] = 2; }

    else

        Unknown macro: { S[n][n] = 2 * nCm(2 * n - 2, n - 1) + 2 * nCm(2 * n - 2, n); }

}

for(int n = 1; n <= N; ++n)
{
    for(int k = 1; k <= n && k <= K; ++k)
    {
        // if k == n, calculate C[n][n] using S[n][n] computed above.
        // C[n][n] = S[n][n] - S[n][n - 1]

        if(k == n)


            Unknown macro: { C[n][k] = S[n][n] - S[n][n - 1]; }

        else // C[n][k] = C[1][1] * C[n - 1][k] + C[2][2] * C[n - 2][k] + ... + C[k][k] * sum(C[n - k][1], C[n - k][2], ..., C[n - k][k])
        {
            for(int i = 1; i <= k; ++i)

                Unknown macro: { C[n][k] += multi(C[i][i], (i < k ? C[n - i][k] ) }

            S[n][k] = S[n][k - 1] + C[n][k];
        }
    }
}

return S[N][K];
};
```

 Posted by wadewu at [Feb 04, 2014 15:39](#) | [Reply To This](#)

 [Add Comment](#)

