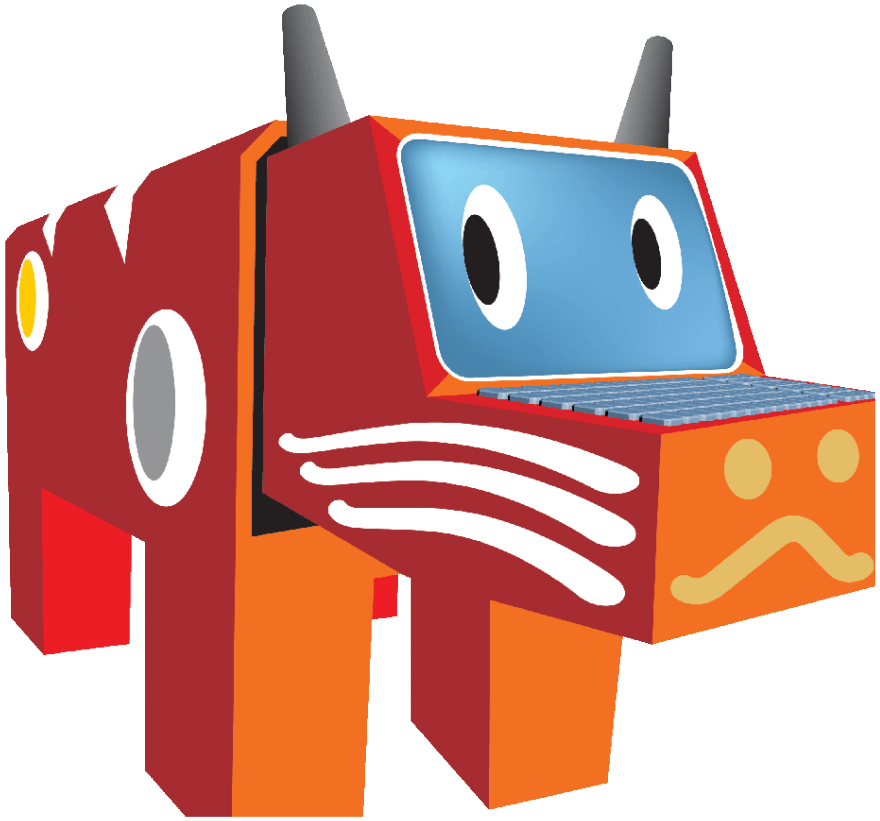
パソコン甲子園２０１７本選

プログラミング部門　解説



会津大学

2017年11月4日

概要

AIZU
THE UNIVERSITY OF AIZU

#	タイトル	分野	実装	思考	得点	正答数
1	旗を作ろう	基礎	☆	☆	3	29
2	Bange Hills Tower	基礎	☆	☆	4	28
3	年の差は。	基礎	☆	★☆	5	25
4	電子メトロノーム	整数	★☆	★☆	6	27
5	朝昼晩ごはん	全探索	★☆	★☆	7	13
6	市松模様	アドホック	★★	★★	8	15
7	おみくじ	場合分け	★★☆	★★★	9	10
8	オノガワ湖調査	計算幾何学	★★★	★★★	10	1
9	抽選ボックス	動的計画法	★★☆	★★★★	12	3
10	パーティ	グラフ・動的計画法	★★★★	★★★☆	12	2
11	航空写真	文字列処理	★★☆	★★★★	12	0
12	鉄の棒	場合分け・数え上げ	★★★★☆	★★★★☆	12	0

問１　旗を作ろう

AIZU
THE UNIVERSITY OF AIZU

問題概要

- 旗の横の大きさWと縦の大きさH、旗の真ん中に書く文字cが与えられる.
- 与えられた大きさで、文字cが真ん中に描かれた旗を描く.
- ただし、旗の四隅には「＋」、横の辺には「-」、縦の辺には「|」、旗の内部（真ん中の文字以外）には「.」を使う
- 3≦W,H≦21でW,Hは奇数. cは大文字のアルファベット.

W＝9, H＝5, c=‘A’のときの例：

```
+-------+
|.......|
|...A...|
|.......|
+-------+
```

総評

- 正答数 29.　FA 5:47　FiveAma
- プログラムの文法を理解していれば解ける問題設定.
- すべてのチームが正解した.

問１　旗を作ろう

AIZU
THE UNIVERSITY OF AIZU

解法

- ２重ループをまわす.
- ループカウンタに対しる条件分岐で、角、縦、横、真ん中、それ以外で、出力する文字を変える.
- 色々な書き方があります.

問１　旗を作ろう

AIZU
THE UNIVERSITY OF AIZU

解答例（C/C++）　２重ループと条件分岐

```
#include <stdio.h>

main() {
    int w, h, i, j;
    char c;
    scanf("%d %d %c", &w, &h, &c);

    for ( int i=1; i<=h; ++i ) {
        for ( int j=1; j<=w; ++j ) {

            if      ( (i== 1 && j == 1) || (i== 1 && j == w) ||
                      (i== h && j == 1) || (i== h && j == w) )
                printf("+");
            else if ( i == 1 || i == h ) printf("-");
            else if ( j == 1 || j == w ) printf("|");
            else if ( i == (h+1)/2 && j == (w+1)/2 ) printf("%c", c);
            else printf(".");
        }
        printf("¥n");

    }
}
```

問１　旗を作ろう

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）　２重ループと関数

```
#include <cstdio>

int w, h;
char c;

void Draw( char ends, char mid, bool isCenter=false ) {
    printf("%c", ends);
    for ( int i=1; i+1<w; ++i ) {
        if ( isCenter && i == w/2 ) printf("%c", c);
        else printf("%c", mid);
    }
    printf("%c¥n", ends);
}

main() {
    scanf("%d %d %c", &w, &h, &c);

    Draw( '+', '-' );
    for ( int i=1; i+1<h; ++i ) Draw( '|', '.', i == h/2 );
    Draw( '+', '-' );
}
```

問１　旗を作ろう

AIZU
THE UNIVERSITY OF AIZU

解答例（Java）　２重ループと条件分岐

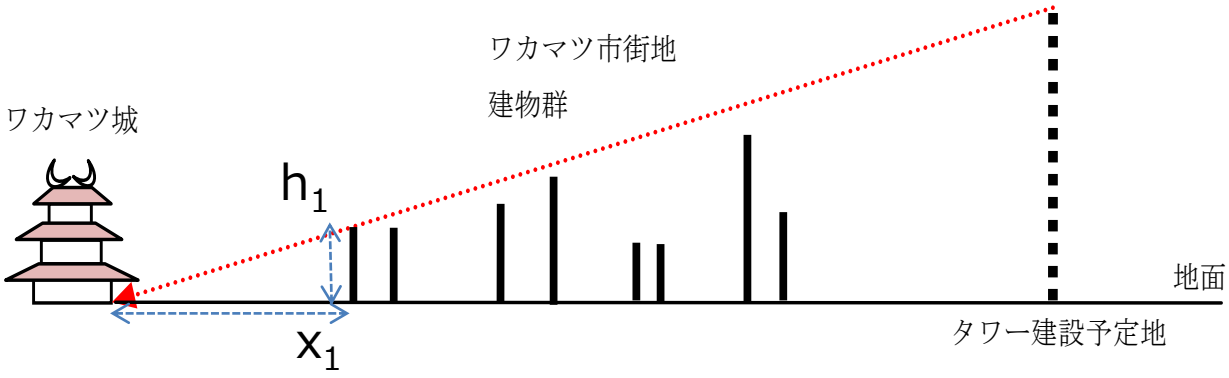
```
import java.io.*;
import java.util.*;
class F01 {
    void solve(){
        Scanner sc = new Scanner( System.in );
        int w = sc.nextInt(), h = sc.nextInt();
        String c = sc.next();

        for ( int i=1; i<=h; ++i ) {
            for ( int j=1; j<=w; ++j ) {
                if      ( (i== 1 && j == 1) || (i== 1 && j == w) ||
                          (i== h && j == 1) || (i== h && j == w) )
                    System.out.print("+");
                else if ( i == 1 || i == h ) System.out.print("-");
                else if ( j == 1 || j == w ) System.out.print("|");
                else if ( i == (h+1)/2 && j == (w+1)/2 )
                    System.out.print(c);
                else System.out.print(".");
            }
            System.out.println("");
        }
    }
    public static void main( String[] a ) {new F01.solve(); }
}
```

問２　Bange Hills Tower

AIZU
THE UNIVERSITY OF AIZU

問題概要

- タワー建設予定地tと、市街地の建物がN個、市街地のそれぞれの建物の位置x_i, 高さh_iが与えられる.
- タワー屋上からワカマツ城の天守閣が上から下まで見えるために、最低限必要な高さを計算する.
- ただし、ワカマツ城やタワーも含めたすべての建物を、地面と垂直な線分とみなす.
- $1 \leqq N \leqq 1000$, $2 \leqq x \leqq 100000$, $1 \leqq x_i < t$, $1 \leqq h_i \leqq 100$.



問２　Bange Hills Tower

AIZU
THE UNIVERSITY OF AIZU

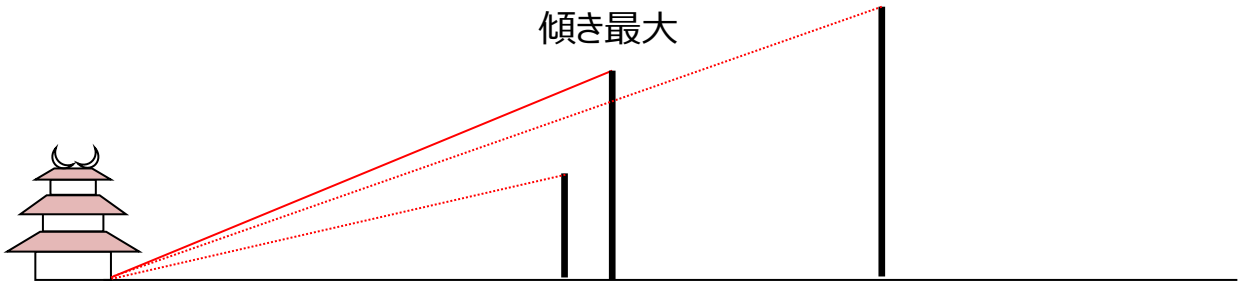
総評

- 正答数 28　FA 9:03　FiveAma
- 簡単な数学（計算）とプログラミングの基礎能力を問う問題.

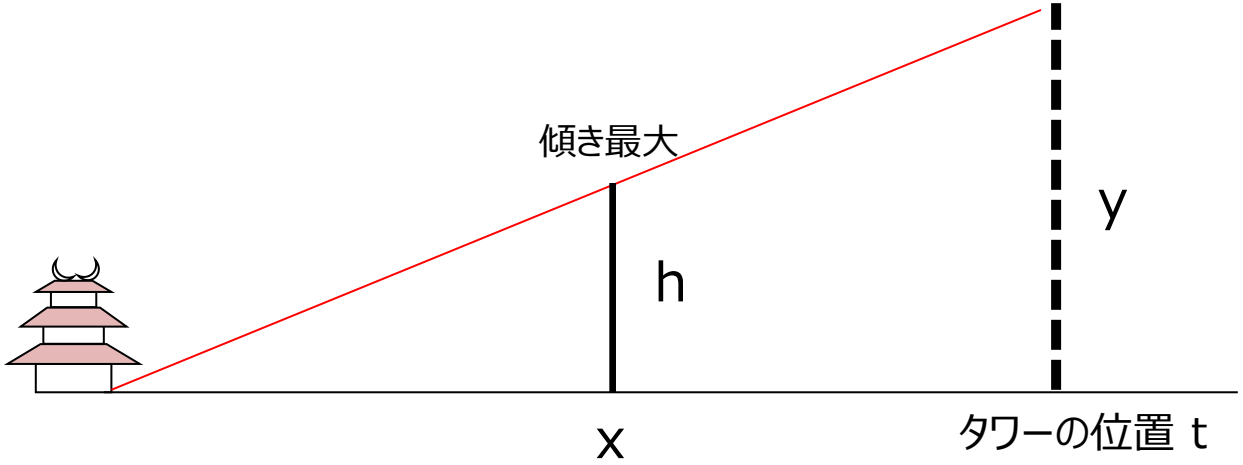
問２　Bange Hills Tower

解法

- 傾きh_i/x_iの最大値$\max(h_i/x_i)$を求める.



- 傾きが最大のときの直線の方程式が、タワーの座標tでとる値
  $y = \max(h_i/x_i) \times t$ が答え



問２　Bange Hills Tower

AIZU
THE UNIVERSITY OF AIZU

解答例（C/C++）

```
#include <stdio.h>

main() {
    int i, n;
    double t, slope, max_slope = 0.0, xi, hi;

    scanf("%d %lf", &n, &t);

    for ( i=0; i<n; ++i ) {
        scanf("%lf %lf", &xi, &hi);
        slope = hi/xi;
        if ( slope > max_slope ) max_slope = slope;
    }

    printf("%lf¥n", max_slope*t);
}
```

問２　Bange Hills Tower

AIZU
THE UNIVERSITY OF AIZU

解答例（Java）

```
import java.io.*;
import java.util.*;

class F02 {

    void solve(){
        Scanner sc = new Scanner( System.in );
        int n = sc.nextInt();
        double t = sc.nextDouble();

        double max_slope = 0.0;
        for ( int i=0; i<n; ++i ) {
            double xi = sc.nextDouble();
            double hi = sc.nextDouble();
            double slope = hi/xi;
            if ( slope > max_slope ) max_slope = slope;
        }
        System.out.println( max_slope*t );
    }
    public static void main( String[] a ) { new F02.solve(); }
}
```

問３　歳の差は。

AIZU
THE UNIVERSITY OF AIZU

問題概要

- はつみとタクの誕生年y_i、月m_i、日d_iが与えられる.
- 永遠の月日の中で、２人の歳の差が最大何歳離れるか求める.
- 歳は誕生日になった瞬間に１加算されるとする。また、誕生日がうるう年の２月２９日だった場合、うるう年でない年には３月１日になった瞬間１加算されるとする.
- $1 \leqq yi \leqq 3000$, $1 \leqq m_i \leqq 12$, $1 \leqq d_i \leqq D_{max}$.

Dmaxは以下の条件を満たす.

- 与えられた年がうるう年でなく月が2月のときは28.
- 与えられた年がうるう年で月が2月のときは29.
- 与えられた月が4,6,9,11月のときは30.
- それ以外の月は31.

総評

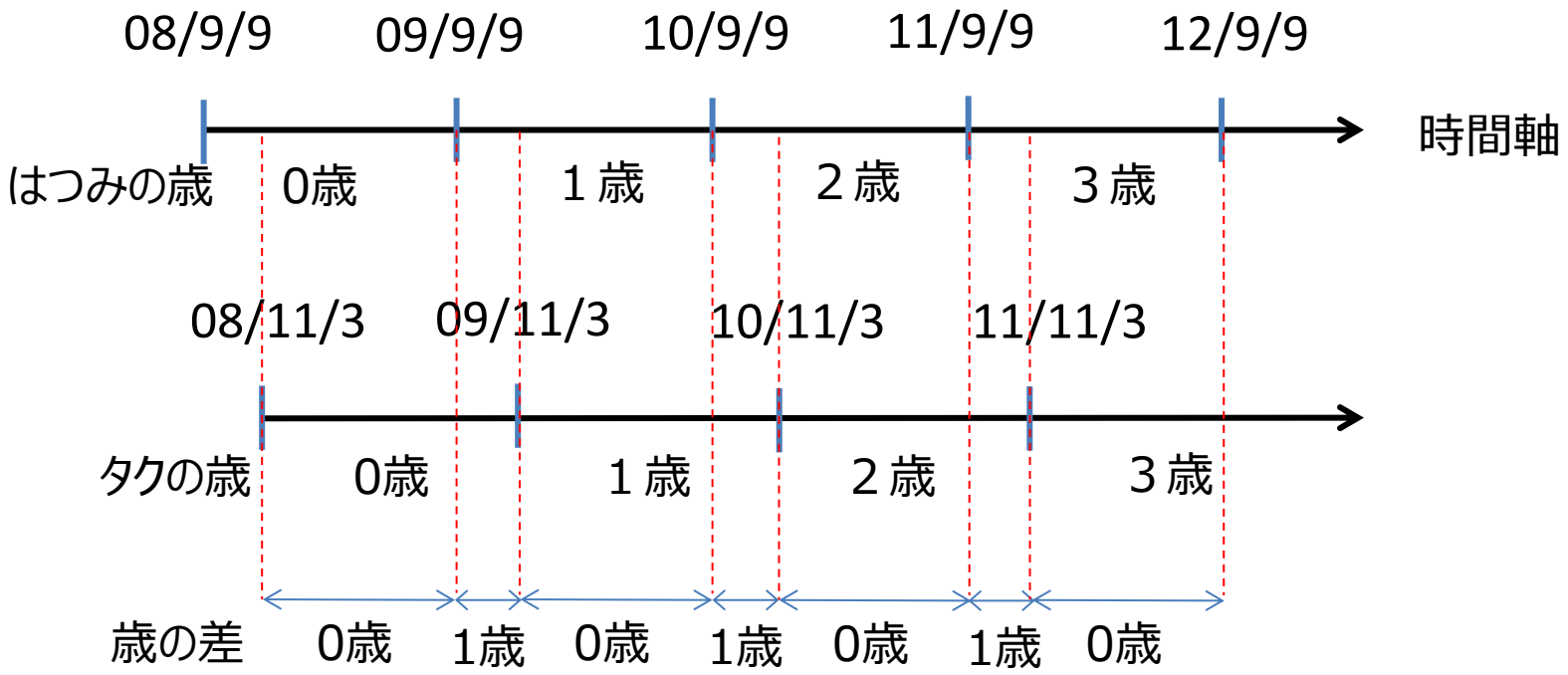
- 正答数 25　FA 22:14　C++Castle
- 論理的な考え方が必要な問題.

問３　歳の差は。

考察

- 誕生月日が一致する場合、誕生年の差が答えになることは明らか.
- 月日が違う場合どうなるのか？

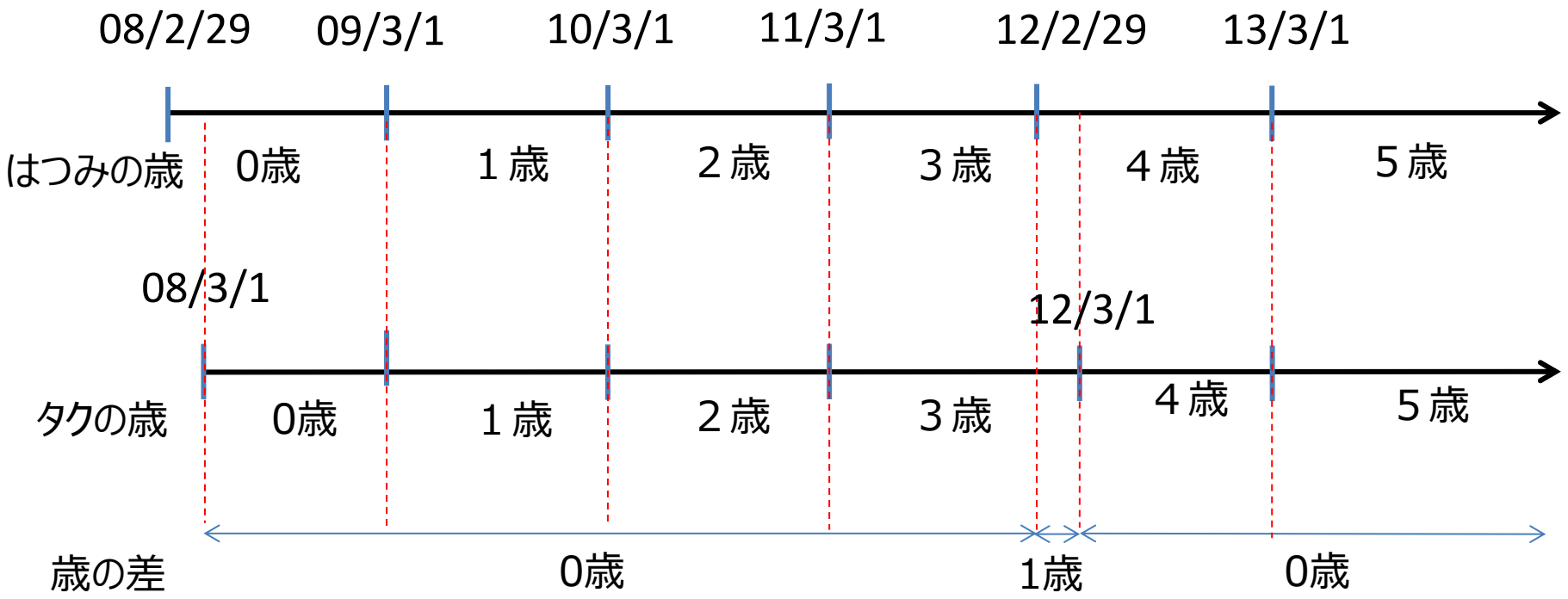
- はつみの誕生年月日2008/9/9、タクの誕生年月日2008/11/3の場合.



問３　歳の差は。

考察

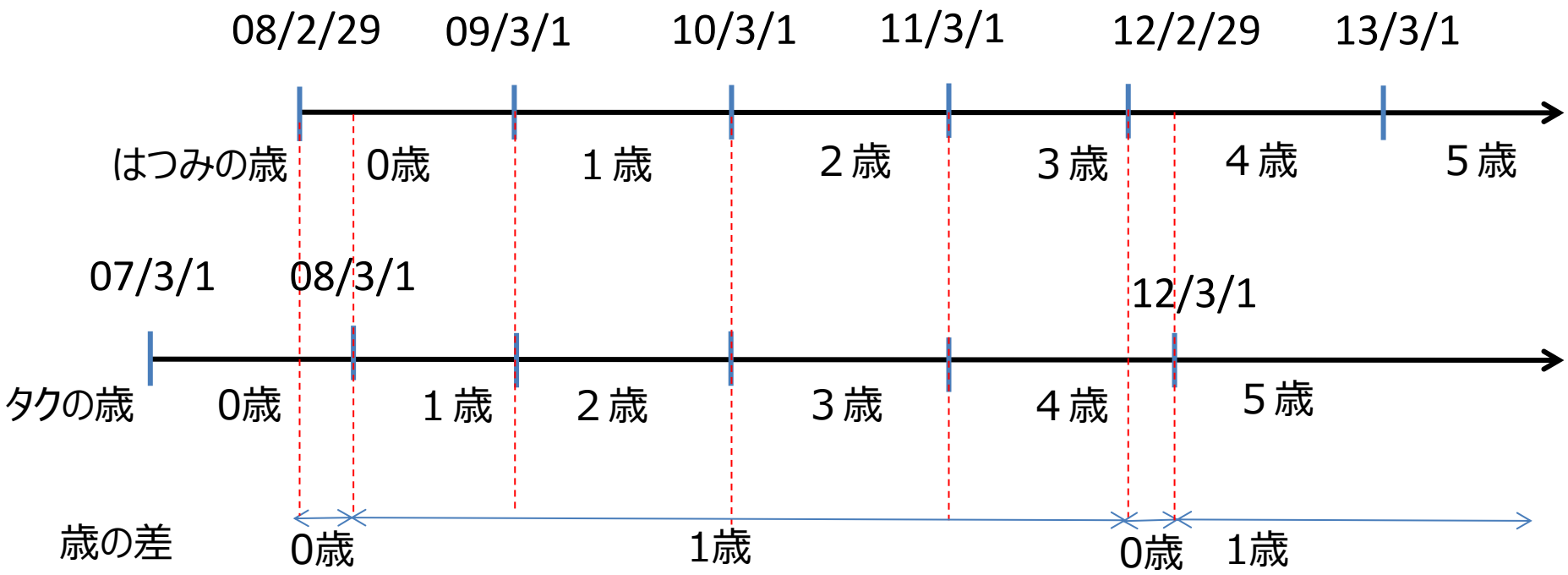
- はつみの誕生年月日2008/2/29、タクの誕生年月日2008/3/1の場合.



問３　歳の差は。

考察

- はつみの誕生年月日2008/2/29、タクの誕生年月日2007/3/1の場合.



- 月日が一致しない場合、誕生月日が早い方が相対的に＋１歳になる日が存在する.
- 片方が2/29に生まれて、もう片方が3/1に生まれた場合、うるう年以外では誕生日が一致するが、うるう年にはまた１日ずれるから、結局うるう年は関係ない.

問３　歳の差は。

AIZU
THE UNIVERSITY OF AIZU

解法

- 誕生年月日の早い方の誕生年月日をY, M, Dとする.
- 誕生年月日の遅い方の誕生年月日をy, m, dとする.
- M月D日がm月d日よりカレンダー上で早い日付の場合、答えはY-y+1.
- そうでなければY-y.

別解法

- 二人とも生まれて以降の日から、４００年分ほどループを回す.
- その間、毎日歳の差を計算して、最大値を求める.

問３　歳の差は。

AIZU
THE UNIVERSITY OF AIZU

解答例（C）

```
#include <stdio.h>

void swap( int* a, int* b ) { int t = *a; *a = *b; *b = t; }

main() {
    int y1, m1, d1, y2, m2, d2;

    scanf(“%d %d %d %d %d %d”, &y1, &m1, &d1, &y2, &m2, &d2;

    if ( y1 > y2 || (y1 == y2 && (m1 > m2 || (m1 == m2 && d1 > d2))) ) {
        swap( &y1, &y2 );
        swap( &m1, &m2 );
        swap( &d1, &d2 );
    }

    if ( m1 < m2 || (m1 == m2 && d1 < d2 ) )
        printf("%d¥n", y2 - y1 + 1);
    else printf("%d¥n", y2 - y1);
}
```

問３　歳の差は。

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

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

main() {
    pair< int, pair<int,int> > p1, p2;

    cin >> p1.first >> p1.second.first >> p1.second.second;
    cin >> p2.first >> p2.second.first >> p2.second.second;

    if ( p1 > p2 ) swap( p1, p2 );

    if ( p1.second < p2.second )
        cout << p2.first - p1.first + 1 << endl;
    else cout << p2.first - p1.first << endl;
}
```

問３　歳の差は。

AIZU
THE UNIVERSITY OF AIZU

解答例（Java）主要部分のみ

```
void solve(){
    Scanner sc = new Scanner( System.in );

    int[] p1 = new int[3];
    int[] p2 = new int[3];
    for ( int i=0; i<3; ++i ) p1[i] = sc.nextInt();
    for ( int i=0; i<3; ++i ) p2[i] = sc.nextInt();

    if ( p1[0] > p2[0] ||
         (p1[0] == p2[0] &&
          (p1[1] > p2[1] || (p1[1] == p2[1] && p1[2] > p2[2]))) ) {
        int t = p1[0];  p1[0] = p2[0];  p2[0] = t;
        t = p1[1];  p1[1] = p2[1];  p2[1] = t;
        t = p1[2];  p1[2] = p2[2];  p2[2] = t;
    }

    if ( p1[1] < p2[1] || (p1[1] == p2[1] && p1[2] < p2[2]) )
         System.out.println( p2[0] - p1[0] + 1 );
    else System.out.println( p2[0] - p1[0] );

}
```

問題4　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

問題概要

- N台の電子メトロノームがあり、i番目のメトロノームはt_i秒間隔で音が一瞬だけ鳴る.
- いくつかのメトロノームの鳴る間隔を調整することで、すべてのメトロノームを同時に鳴らしたい.
- このとき、すべてのメトロノームが同時に鳴る間隔を、できるだけ短くしたい.
- ただし、メトロノームの音が鳴る間隔は増やすことしかできない.
- メトロノームが同時に鳴る間隔を最小とするときの、メトロノームの間隔の調整量の和の最小値を求める.
- $1 \leqq N \leqq 10^5$, $1 \leqq t_i \leqq 10^4$.

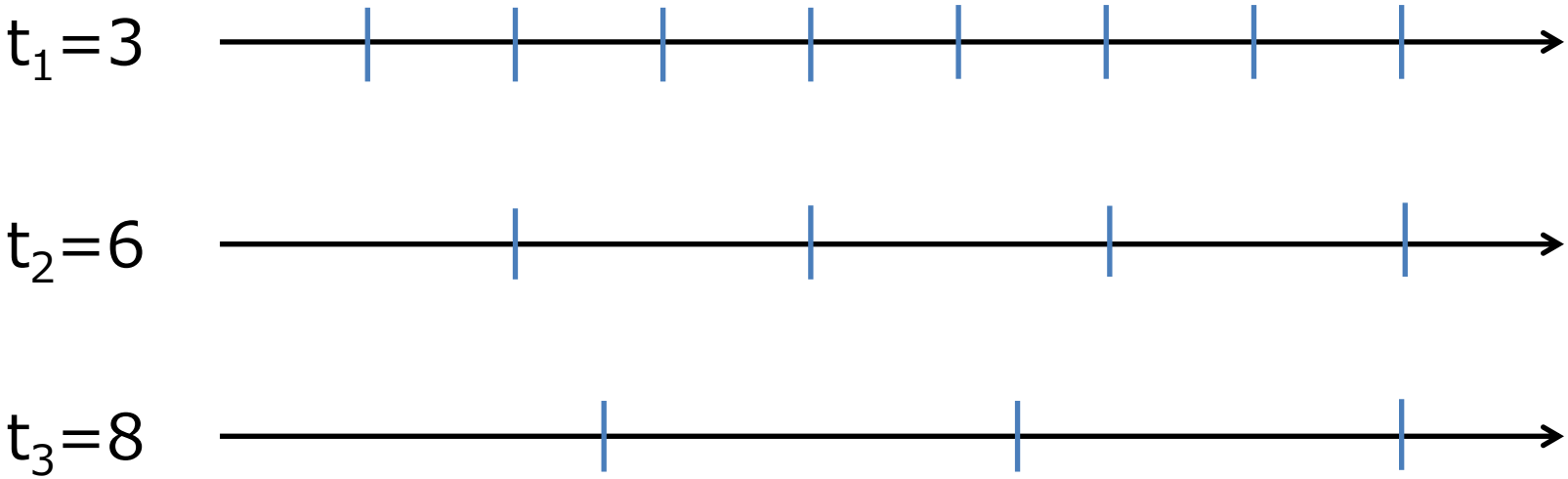
総評

- 正答数 27　FA 15:07　FiveAma
- 最小公倍数に関する問題.

問題4　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

考察



- すべてのメトロノームが同時に鳴る間隔を最小としたいので、一番間隔が長いものに合わせて他のメトロノームの間隔を調整すべき.
- 一番間隔が長いものは$t_3=8$なので、他のものはそれに合わせる.

問題4　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

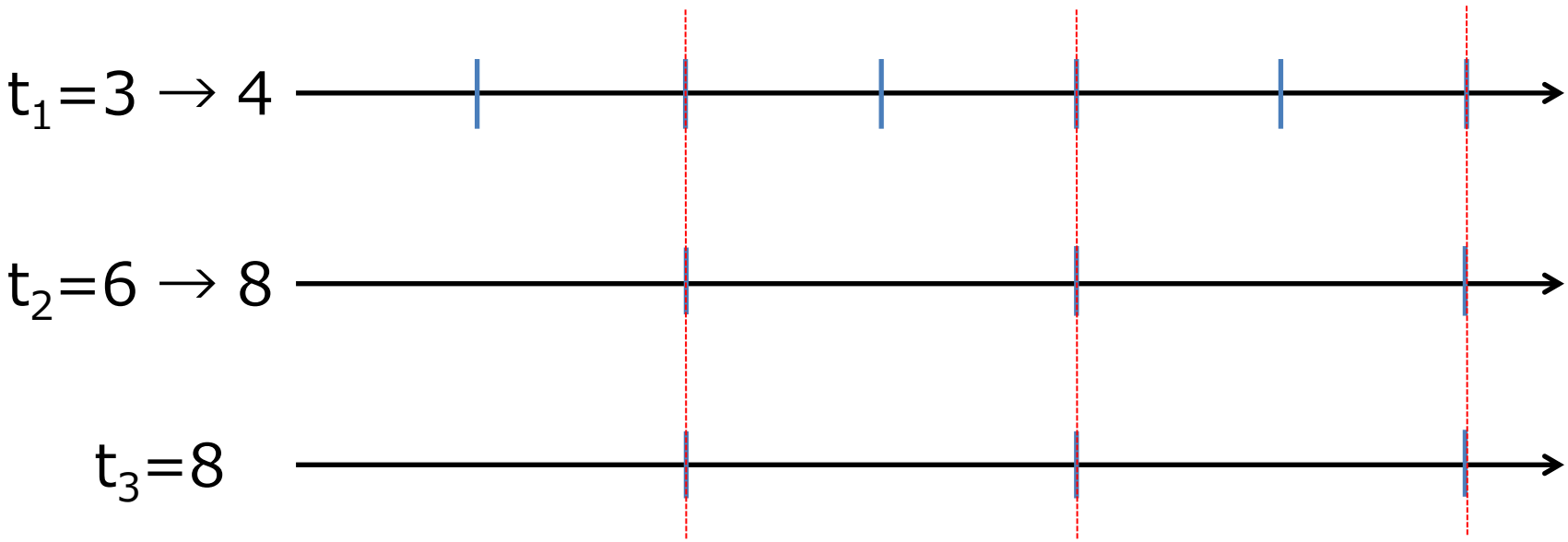
考察

- t_3=8なので、他のものはそれに合わせる.
- t_2=6 → t_2= 6 + 2 = 8.
- t_1=3 → t_1= 3 + 5 = 8 ?
- 合計で調整する量は 2 + 5 = 7?

問題4　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

考察



- t_3の約数1, 2, 4, 8の間隔で鳴らしても、t_3と同時に鳴らすことが出来る.
- ~~$t_1=3 \rightarrow t_1=3+5=8$?~~
- $t_1=3 \rightarrow t_1=3+1=4$ とすればt_1は最小の調整量にできる.
- 合計で調整する量は 2 + 1 = 3.

問題4　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

解法

- 一番周期の長いメトロノームが鳴る間隔に合わせる.
- 他のメトロノームは、一番周期の長いメトロノームの約数の間隔で鳴らせばよい.
- 全てのメトロノームの間隔の最小公倍数が、一番周期の長いメトロノームの間隔となる.

具体的には

- t_iのうち一番大きな値を見つける(その値をTとする).
- Tの約数をすべて求める.
- すべてのt_iについて、その値を増やしたときに最初に一致するTの約数a_iを探す.
- 答えはa_i-t_iをすべて足し合わせたもの.

問４　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
main(){
  int n, x;
  vector<int> t, r;
  int maxv = 0;
  cin >> n;
  for ( int i = 0; i < n; i++ ){
    cin >> x; t.push_back(x);
    maxv = max(maxv, x);
  }

  for ( int i = 1; i <= maxv; i++ ) if ( maxv%i== 0 ) r.push_back(i);

  long long ans = 0;
  for ( int i = 0; i < n; i++ ){
    for ( int j = 0; ; j++ ){
      if ( r[j] >= t[i] ){
        ans += r[j] - t[i];
        break;
      }
    }
  }
  cout << ans << endl;
}
```

問４　電子メトロノーム

AIZU
THE UNIVERSITY OF AIZU

解答例（Java）主要部分のみ

```
long max_t = 0;
long[] timing = new long[n];
for ( int i=0; i<n; ++i ) {
    timing[i] = sc.nextLong();
    if ( timing[i] > max_t ) max_t = timing[i];
}

ArrayList< Long > divisor = new ArrayList< Long >();
for ( Long i=1L; i<=max_t>>1; ++i ) {
    if ( max_t%i == 0 ) divisor.add( i );
}
divisor.add( max_t );

Long adj = 0L;
for ( Long t : timing ) {

    int j = 0;
    for ( ; j<divisor.size(); ++j ) {
        if ( divisor.get( j ) >= t ) break;
    }
    adj += divisor.get( j ) - t;
}
System.out.println( adj );
```

問題5　朝昼晩ごはん

AIZU
THE UNIVERSITY OF AIZU

問題概要

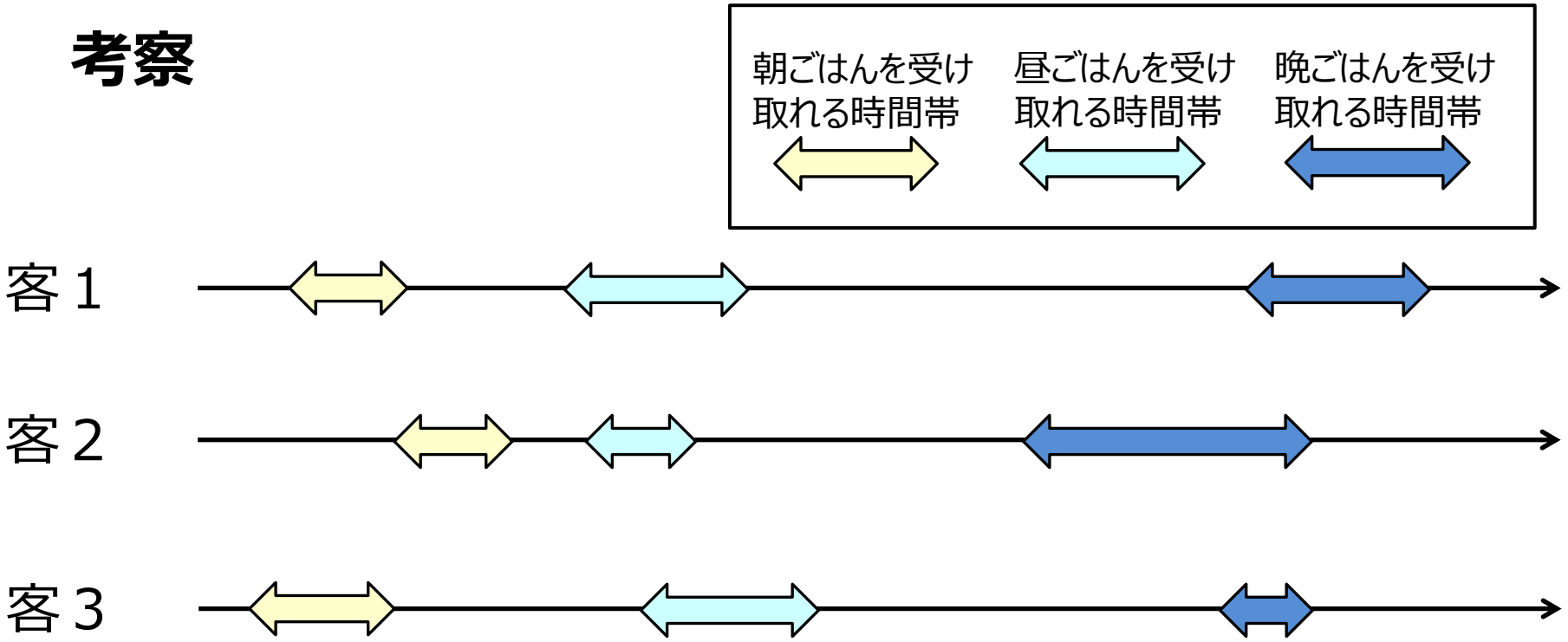
- N人のお客さんがいて、それぞれの客が朝ごはん、昼ごはん、晩ごはんを受け取ることができる時間帯が決まっている.
- i番目のお客さんが、朝ごはんを受け取ることが出来る時間帯はast_iからaet_iの間、昼ごはんはhst_iからhet_iの間、晩ごはんはbst_iからbet_iの間.
- ３つの食事を提供する時刻を決めることができるとき、朝、昼、晩ごはんをすべて受け取ることができる客の最大人数を求める.
- $1 \leqq N \leqq 50$, ast_iなどの時刻は、すべて１分刻みで0時0分から23時59分の間.
- hst_iはaet_iより後の時刻で、bst_iはhet_iより後の時刻であり、各時間帯は午前０時をまたがない.

総評

- 正答数 13　FA 38:04　Nodoguro
- 計算量の見積もりが大事な問題.

問題５　朝昼晩ごはん

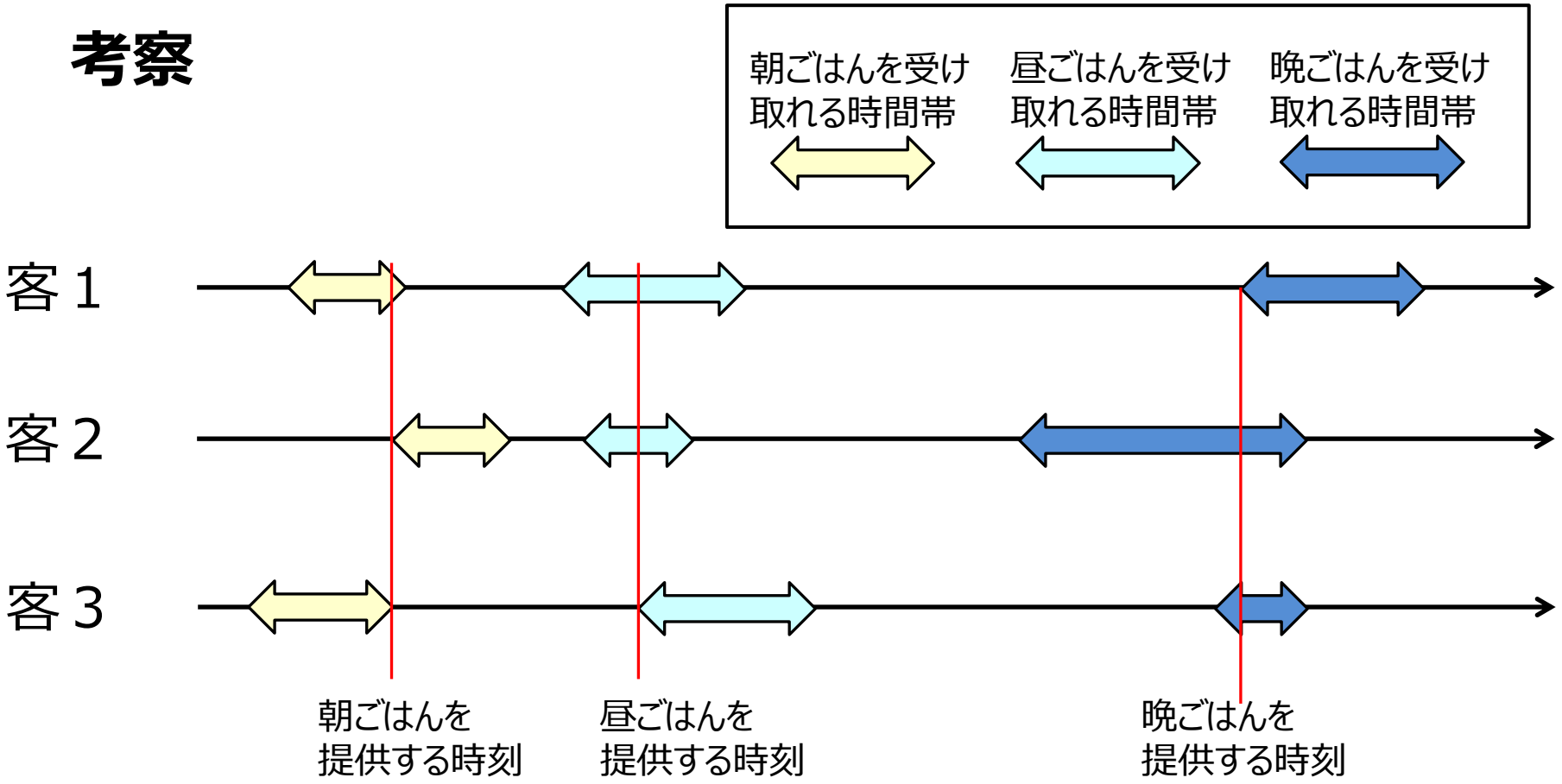
考察



問題5　朝昼晩ごはん

AIZU
THE UNIVERSITY OF AIZU

考察



- 朝、昼、晩ごはんを提供する時刻をこのように決めれば３人とも３食受け取れる.

問題5　朝昼晩ごはん

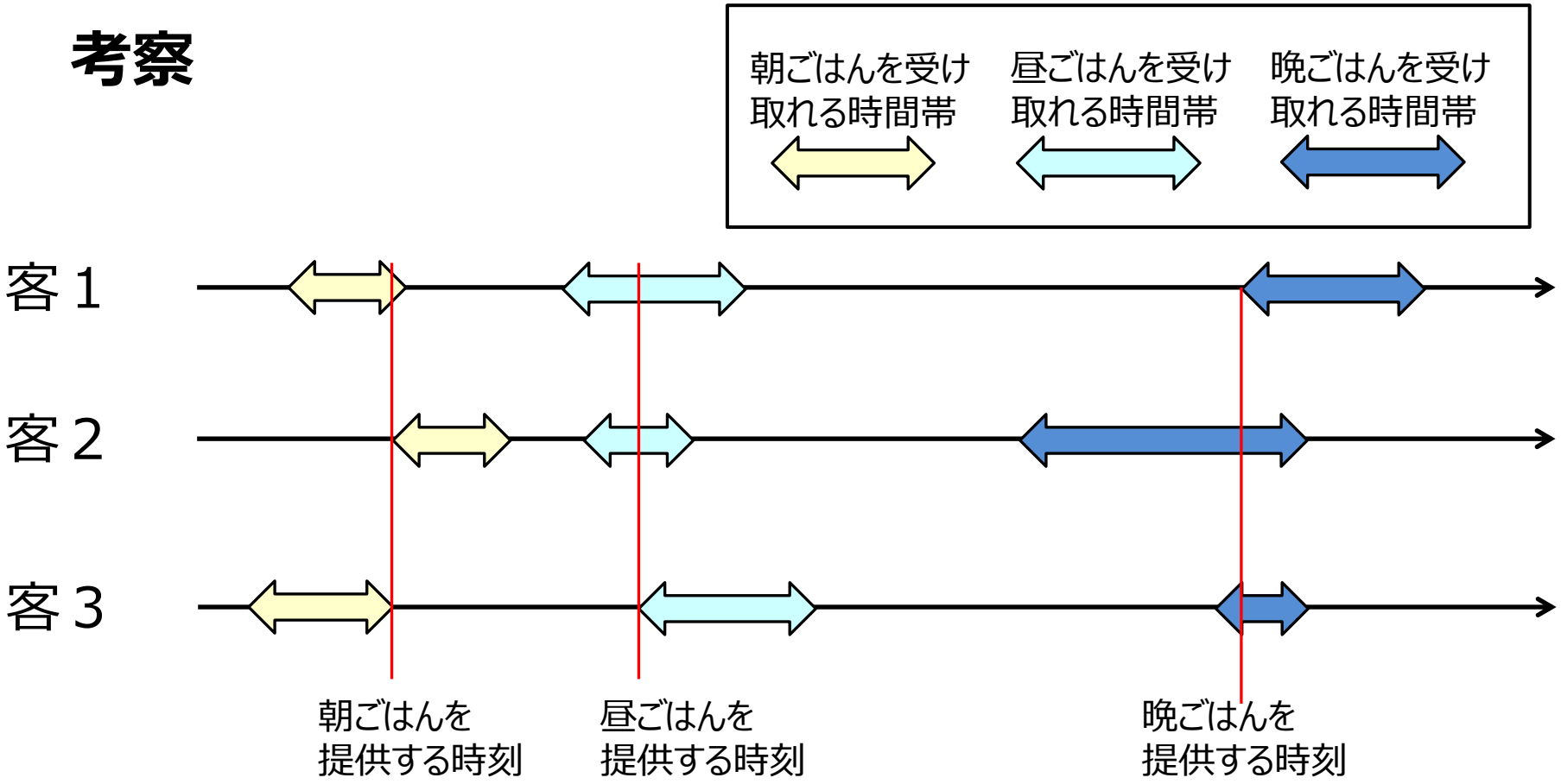
AIZU
THE UNIVERSITY OF AIZU

想定時間切れ解法

- 朝、昼、晩ごはん、それぞれ0時0分から23時59分まで、何人が朝昼晩御飯をすべて受け取れるか試す4重ループ.
- 時刻は24×60＝1440通りあり、朝昼晩についてすべて試すと1440^3＝30億程度.
- これに人数N＝50がかかるから、150億程度なので時間切れ.

問題5　朝昼晩ごはん

考察



- あるごはんを受け取れる人数が変わるのは、ある客の、そのごはんを受け取れる時間帯の開始時刻か終了時刻.
- 全員の開始時刻と終了時刻についてだけ、何人の客が３食とも受け取れるか調べればよい.

問題5　朝昼晩ごはん

AIZU
THE UNIVERSITY OF AIZU

解法

- ごはんを受け取れる人数が変わる瞬間は、だれかがごはんを受け取れるようになる開始時刻か終了時刻だけ.
- 実際は、全員分の開始時刻だけ、または全員分の終了時刻だけ調べればよい.
- 全員の、朝ごはんを受け取れる終了時刻、昼ごはんを受け取れる終了時刻、晩ごはんを受け取れる終了時刻について、何人が朝昼晩ごはんを３食とも受け取れるか調べる.
- 計算量 $O(N^4)$. N=50なら6×10^6=600万程度.

問題5　朝昼晩ごはん

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）主要部分のみ

```
int periods[N_MAX][3][2];   vector< int > check_points[3];
int n, h, m;  scanf("%d", &n);
for ( int i=0; i<n; ++i ) {
    for ( int j=0; j<3; ++j ) {
        for ( int k=0; k<2; ++k ) {
            scanf("%d %d", &h, &m);
            periods[i][j][k] = h*60 + m;
            if ( k == 1 ) check_points[j].push_back( periods[i][j][k] );
        }
    }
}
int max_n = 0;
for ( int b : check_points[0] ) {
    for ( int l : check_points[1] ) {
        for ( int d : check_points[2] ) {
            int res = 0;
            for ( int i=0; i<n; ++i ) {
                if ( b >= periods[i][0][0] && b <= periods[i][0][1] &&
                     l >= periods[i][1][0] && l <= periods[i][1][1] &&
                     d >= periods[i][2][0] && d <= periods[i][2][1] ) ++res;
            }
            max_n = std::max( max_n, res );
        }}}
```

問題5　朝昼晩ごはん

AIZU
THE UNIVERSITY OF AIZU

解答例（Java）主要部分のみ

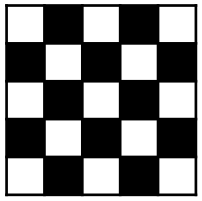
```
int n = sc.nextInt();
int[][][] periods = new int[n][3][2];
int[][] check_point = new int[3][n];
for ( int i=0; i<n; ++i ) {
    for ( int j=0; j<3; ++j ) {
        for ( int k=0; k<2; ++k ) {
            int h = sc.nextInt(), m = sc.nextInt();
            periods[i][j][k] = h*60 + m;
            if ( k == 1 ) check_point[j][i] = periods[i][j][k];
        }
    }
}
int max_n = 0;
for ( int b : check_point[0] ) {
    for ( int l : check_point[1] ) {
        for ( int d : check_point[2] ) {
            int res = 0;
            for ( int i=0; i<n; ++i ) {
                if ( b >= periods[i][0][0] && b <= periods[i][0][1] &&
                     l >= periods[i][1][0] && l <= periods[i][1][1] &&
                     d >= periods[i][2][0] && d <= periods[i][2][1] ) ++res;
            }
            max_n = Math.max( max_n, res );}}}
```

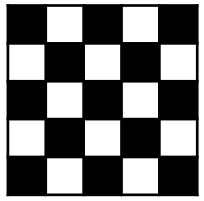
問６　市松模様

AIZU
THE UNIVERSITY OF AIZU

問題概要

- Ｗ×Ｈ個のマスから成る方眼紙が与えられる.
- 各マスは白か黒で塗られている.
- あなたは「任意の２つの行を入れ替える」「任意の２つの列を入れ替える」操作を何度でも行うことができる.
- 市松模様を作れるかどうか判定せよ.
- ２≦W, H≦1000.





総評

- 正答数 15　FA 50:53　それは違うよね.
- 漏れがないように考察することが重要な問題.

問６　市松模様

AIZU
THE UNIVERSITY OF AIZU

解法

- アドホックな判定アルゴリズム.
- 以下の条件を満たすとき、市松模様を作ることができる.
  - 入力をそれぞれW文字からなるH個の文字列とみる.
  - 入力の中には2種類の文字列s1, s2しか存在しない.
  - s1, s2 の各文字は互いに逆になっている.
    - 例：
      - s1: 1001010
      - s2: 0110101
  - s1, s2それぞれについて、白と黒の数の差は１以下である.
- set, sort 等を応用して、種類数のカウントと判定を実装.

問６　市松模様

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
int main(){
  int w,h;
  string s[1000];
  set<string> st;
  char c;
  cin>>w>>h;
  for(int i=0;i<h;st.insert(s[i++]))
    for(int j=0;j<w;j++)cin>>c,s[i]+=c;
  sort(s,s+h);
  bool ans=1;
  int cnt=0;
  for(int i=0;i<w;i++){
    if(s[0][i]=='1')cnt++;
    if(s[h/2-1][i]==s[(h+1)/2][i])ans=0;
  }
  if((cnt!=w/2&&cnt!=(w+1)/2)||st.size()!=2)ans=0;
  cout<<(ans?"yes":"no")<<endl;
  return 0;
}
```

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

問題概要

- 各桁が１から９までの数字からなる整数が与えられる.
- この整数を1か所以上で区切って、いくつかの整数に分ける.
- これらの整数の最大値と最小値の差を、最小化せよ.
- 与えられる整数は100000桁を超えない.
- 例：11121314

1|11|21|3|14 → 21 − 1 = 20

11|12|13|14 → 14 − 11 = 3　（最小）

総評

- 正答数 10　FA 1:17:58　本マグロ
- 漏れがないように考察することが重要な問題.

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解法

- アドホックなアルゴリズム.
- 与えられる整数を長さＫ（nの桁数）の文字列Ｓとして扱う.
- 以下の場合について最大値と最小値の差を求め、最小をとる
  - Ｓを等分割する場合
  - Ｓを等分割しない場合

11 | 12 | 13 | 14　　1 | 11 | 21 | 3 | 14

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解法

- Ｓを等分割する場合
  - Ｋの約数Dすべてについて、Ｓを連続するD文字の列に分け、最大値と最小値の差を求める.
  - O(K×Kの約数の数).

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解法

- Ｓを等分割しない場合
  - Ｓを等分割する方法から、差の最小値は最大でも８であると分かる.
  - ３桁以上の列を含めて分けることで最適解は得られない.
    - 3桁の最小値111と２桁の最大値99の差は８より大きい

```
111|99 → 111 − 99 = 12 > 8
```

  - 1桁と2桁の列で分けた場合のみを考えればよい.
    - ２以上の数字は２桁の列の10の位になることはない

```
21|9 → 21 − 9 = 12 > 8
```

    - 1は１桁の列には含めない

```
11|1 → 11 − 1 = 10 > 8
```

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解法

- Ｓを等分割しない場合（1桁と2桁の列で分ける場合）
  - 2以上の数字は10の位にならない.
  - 1は常に２桁の列に含まれる.
  - 11, 12, 13, ..., 19 は２桁にし、それら以外は１桁にする

12|9 → 12 − 9 = 3

11|12|9|14|19 → 19 − 9 = 10

（※等分割した場合の8が答えになる）

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
main(){
  cin >> S;

  int ans = 8;
  ans = min(ans, checkEqual(S));
  ans = min(ans, check12(S));

  cout << ans << endl;
}
```

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
int checkEqual(string S){
  string mins, maxs;
  int ans = 8;
  for ( int k = 1; k < S.size(); k++ ){
    if ( S.size()%k != 0 ) continue;
    mins = maxs = S.substr(0, k);
    for ( int s = 0; s < S.size(); s += k ){
      maxs = max(maxs, S.substr(s, k));
      mins = min(mins, S.substr(s, k));
    }
    ans = min(ans, sub(maxs, mins));
  }
  return ans;
}
```

問７　おみくじ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
int check12(string S){
  int maxv = 0;
  int minv = 10;
  int p = 0, v;
  while(p<S.size()){
    v = val(S[p]);
    if ( S[p] == '1' && p+1 < S.size()) {
      v = 10 + val(S[p+1]);
      p++;
    }
    p++;
    maxv = max(maxv, v);
    minv = min(minv, v);
  }
  return maxv - minv;
}
```

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

問題概要

- 調査開始地点(x_s,y_s)と終了地点(x_g,y_g)、N個の点から成る凸多角形によって表されるオノガワ湖の領域が与えられる．各点の座標(x_i,y_i)．
- 調査開始地点から調査終了地点までの最短の距離を求める．
- ただし、途中でオノガワ湖の沿岸に立ち寄らなければならない。また、オノガワ湖は沿岸に沿って歩くことはできても、湖に入ることはできない．
- $3 \leqq N \leqq 100$, $0 \leqq x_i, y_i, x_s, y_s, x_g, y_g \leqq 10^4$.

総評

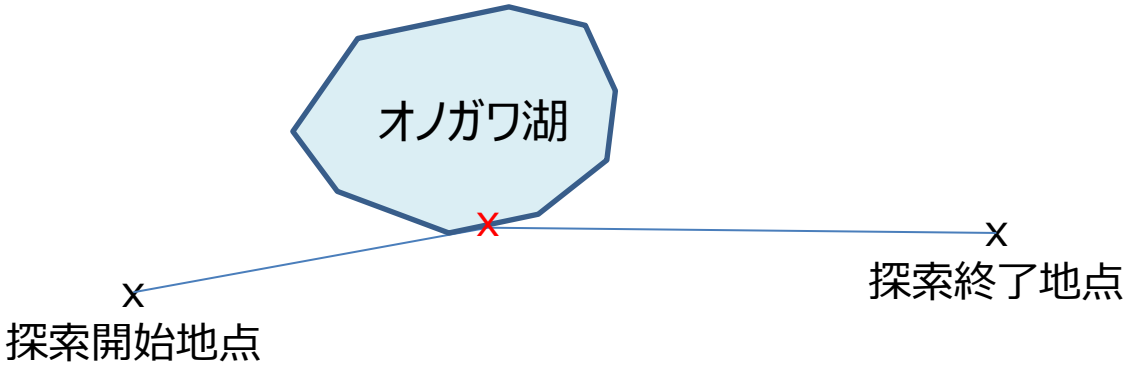
- 正答数 1　FA 2:55:26　大好き、一面の雪
- 計算幾何学の問題．ライブラリがあると解きやすい．

問８　オノガワ湖調査

解法

- 場合分け幾何.
- 探索開始地点と終了地点を結ぶ線分が、凸多角形と交差しない場合.

- 探索開始地点と終了地点から見える凸多角形の線分すべてについて、最小距離となる経由地点を３分探索.

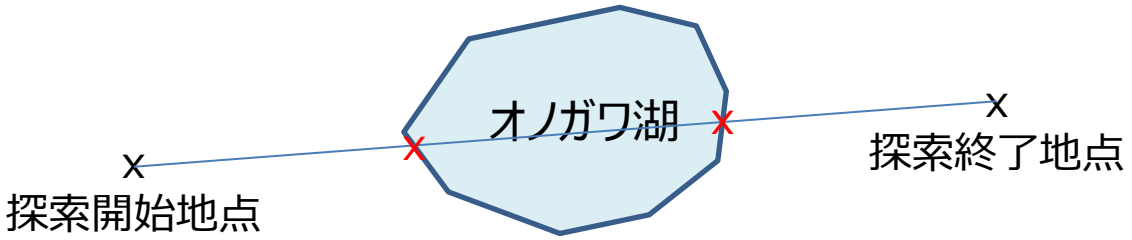


問８　オノガワ湖調査

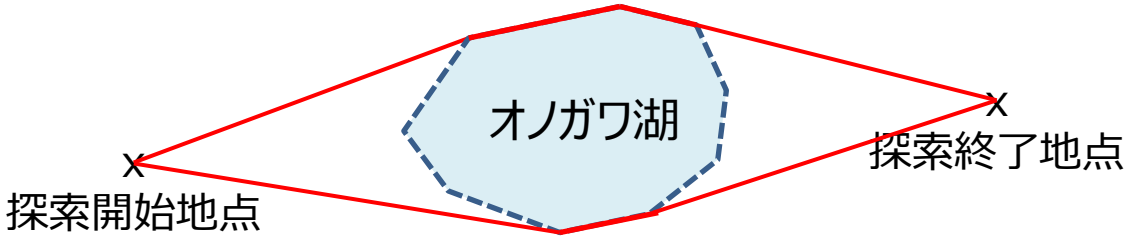
AIZU
THE UNIVERSITY OF AIZU

解法

- 探索開始地点と終了地点を結ぶ線分が、凸多角形と交差する場合.



- 探索開始地点と終了地点を含むすべての頂点で凸包を作る.



- 探索開始地点から終了地点まで、凸包の頂点を時計回りと反時計回りに辿ったとき、小さい方の距離が答え.

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
double solve(){
  Segment path = Segment(source, target);
  for ( int i = 0; i < N; i++ ){
    if ( isIntersect(path, Segment(gon[i], gon[(i+1)%N])) ){
      return solve1();//交差する場合
    }
  }
  return solve2();//交差しない場合
```

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
double solve1(){
  Polygon ext = gon;
  ext.push_back(source);
  ext.push_back(target);
  Polygon cvx = andrewScan(ext);
  int n = cvx.size();
  double ans = INF;
  int source_p = -1;
  int target_p = -1;
  for ( int i = 0; i < cvx.size(); i++ ){
    if ( source == cvx[i] ) source_p = i;
    if ( target == cvx[i] ) target_p = i;
  }
  double d1 = 0;
  double d2 = 0;
  int pre = source_p;
  for ( int i = source_p; ; i++ ){
    d1 += getDistance(cvx[pre%n], cvx[i%n]);
    if ( i%n == target_p ) break;
    pre = i;
  }
  pre = target_p;
  for ( int i = target_p; ; i++ ){
    d2 += getDistance(cvx[pre%n],
                      cvx[i%n]);
    if ( i%n == source_p ) break;
    pre = i;
  }
  return min(d1, d2);
}
```

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
double solve2(){
  vector<int> node(N); // counter
  vector<int> edge(N); // counter
  for ( int i = 0; i < node.size(); i++ ){
    node[i] = edge[i] = 0;
  }

  for ( int i = 0; i < N; i++ ){
    int s = i;
    int t = (i+1)%N;
    if ( ccw(gon[s], gon[t], source) != COUNTER_CLOCKWISE ){
      edge[s]++;
      node[s]++; node[t]++;
    }
    if ( ccw(gon[s], gon[t], target) != COUNTER_CLOCKWISE ){
      edge[s]++;
      node[s]++; node[t]++;
    }
  }
```

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
  double mindist = INF;

  for ( int i = 0; i < N; i++ ){
    if ( node[i] != 2 ) continue;
    mindist = min(mindist, getDistance(source, gon[i]) +
                  getDistance(target, gon[i]));
  }

  for ( int i = 0; i < N; i++ ){
    if ( edge[i] != 2 ) continue;
    mindist = min(mindist, getShortestDistance(Segment(gon[i],
                                gon[(i+1)%N]), source, target));
  }

  return mindist;
}
```

問８　オノガワ湖調査

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
double getShortestDistance(Segment seg, Point p1, Point p2){
  Point l = seg.p1;
  Point r = seg.p2;
  for ( int i = 0; i < 50; i++ ){
    Point ll = l + (r - l)*(1.0/3);
    Point rr = l + (r - l)*(2.0/3);
    double d1 = getDistance(ll, p1) + getDistance(ll, p2);
    double d2 = getDistance(rr, p1) + getDistance(rr, p2);
    if ( d1 > d2 ){
      l = ll;
    } else {
      r = rr;
    }
  }
  return getDistance(l, p1) + getDistance(l, p2);
}
```

問９　抽選ボックス

AIZU
THE UNIVERSITY OF AIZU

問題概要

- 抽選ボックスの中にＮ種類のボールが入っている．各種類のボールの数はM以上．
- i種類目のボールには整数a_iが書かれている．ボックスには１つの整数Tが書かれている．
- あなたは２つの整数A, Bを宣言し、ボックスの中から最大M個のボールを取り出すことができる．取り出したボールに書かれた整数の和をSとする．
- S mod T ≧ A かつ S / T ≧ B となるボールの取り出し方は存在するか．
- A, Bの組はＱ個のクエリとして与えられる．
- 1≦N,M≦100000, 1≦T≦1000, 1≦ a_i ≦1,000,000,000
- 0≦A<T, 0≦B≦1,000,000,000

総評

- 正答数 3　FA 1:05:26　solars
- 最適な組み合わせを高速に求めるアルゴリズムが要求される．

問９　抽選ボックス

AIZU
THE UNIVERSITY OF AIZU

考察

- ＳをＴで割った余りが同じ場合はＳが大きい方が優位な選び方である.
- m個までボールを取り出せるときの、ＳをＴで割った余りが j となる最大のＳを動的計画法で求める．dp[m][j]

dp[m+1][(j+a[i])%T] = max(dp[m+1][(j+a[i])%T], dp[m][j] + a[i])

- O(NMT)で時間・メモリ制限.

問９　抽選ボックス

AIZU
THE UNIVERSITY OF AIZU

解法

- 動的計画法+ダブリング
- 基本方針
  - dp[i][j] = 2^i 個までボールを取り出せるときの、SをTで割った余りがjとなる最大のSとする.
- 初期化
  - dp[0][j]にボールに書かれた値の中で余りがjになるもので最大の値を代入する.
  - 1個も選ばないパターン(SをTで割った余りが0でSが0)も考慮することに注意.

問９　抽選ボックス

AIZU
THE UNIVERSITY OF AIZU

動的計画法+ダブリング

- dpテーブルの計算
  - dp[i+1][(j+k)%T] = 全てのjとkの中で(dp[i][j]+dp[i][k]) が最大のもの
  - この更新を2^iがMを超えない間行う.
- 解の更新
  - dpの値を利用して、M個まで選んだ時に、余りがXとなる最大のSを求める.
  - Mを二進数とし、Mのi番目のビットが立っている場合dp[i][0 ~ T-1]の値を利用して、dpと同様の更新を行う.
- あらかじめ累積最大値を求めておくことによって、各クエリに対してO(1)で答えることができる.
- $O(T^2 \log M + Q)$.

問９　抽選ボックス

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
for ( int m = 1; m <= B_MAX; m++ ){ // 2, 4, 8, 16, 32, ...
  for ( int i = 0; i < T; i++ ){
    for ( int j = 0; j < T; j++ ){
      if ( dp[m-1][i] >=0 && dp[m-1][j] >= 0 )
        dp[m][(i+j)%T] = max(dp[m][(i+j)%T], dp[m-1][i] + dp[m-1][j]);
    }
  }

  if ( !(M & (1<<m)) ) continue;

  for ( int i = 0; i < T; i++ ) tmp[i] = ans[i];
  for ( int i = 0; i < T; i++ ){
    for ( int j = 0; j < T; j++ ){
      if ( dp[m][i] >= 0 && ans[j] >= 0 )
        tmp[(i+j)%T] = max(tmp[(i+j)%T], dp[m][i] + ans[j]);
    }
  }
  for ( int i = 0; i < T; i++ ) ans[i] = tmp[i];
}
```

問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

問題概要

- Ｎ人からなるクラスの友達関係が与えられる.
- 友達関係を辿って到達できる人は知り合いである.
- ある友達関係が解消されたとしても、知り合いのままの人は仲間である.
- 以下の条件を満たすように、クラスメイトをパーティに招待する人数を最大化する
  - Tさんが招待リストに入っているとき
    - Ｔさんと仲間であるＵさんは招待リストに入っている
    - ＵさんがＴさんと仲間ではない場合、ＵさんがＴさんと友達であるか、ＵさんがＴさんの仲間の誰かと友達であるなら、Ｕさんはリストに入っていない
- 2≦N≦100000, 1≦ M ≦200000

総評

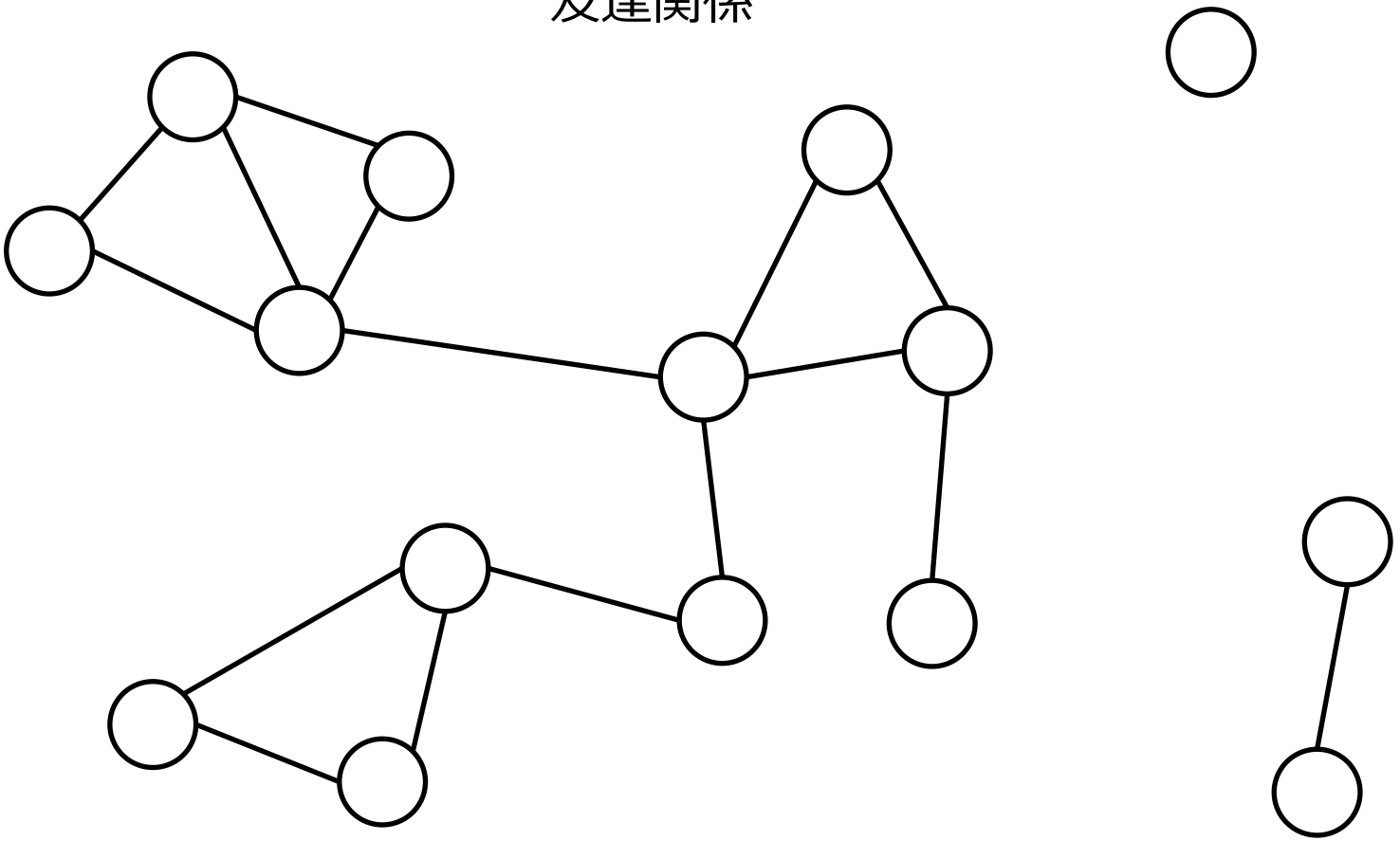
- 正答数 2　FA 2:14:38　大好き、一面の雪.
- 論理的に問題を読み解く力が必要.
- 高等的なグラフの知識が要求される問題.　ライブラリを持っていると解きやすい.

問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

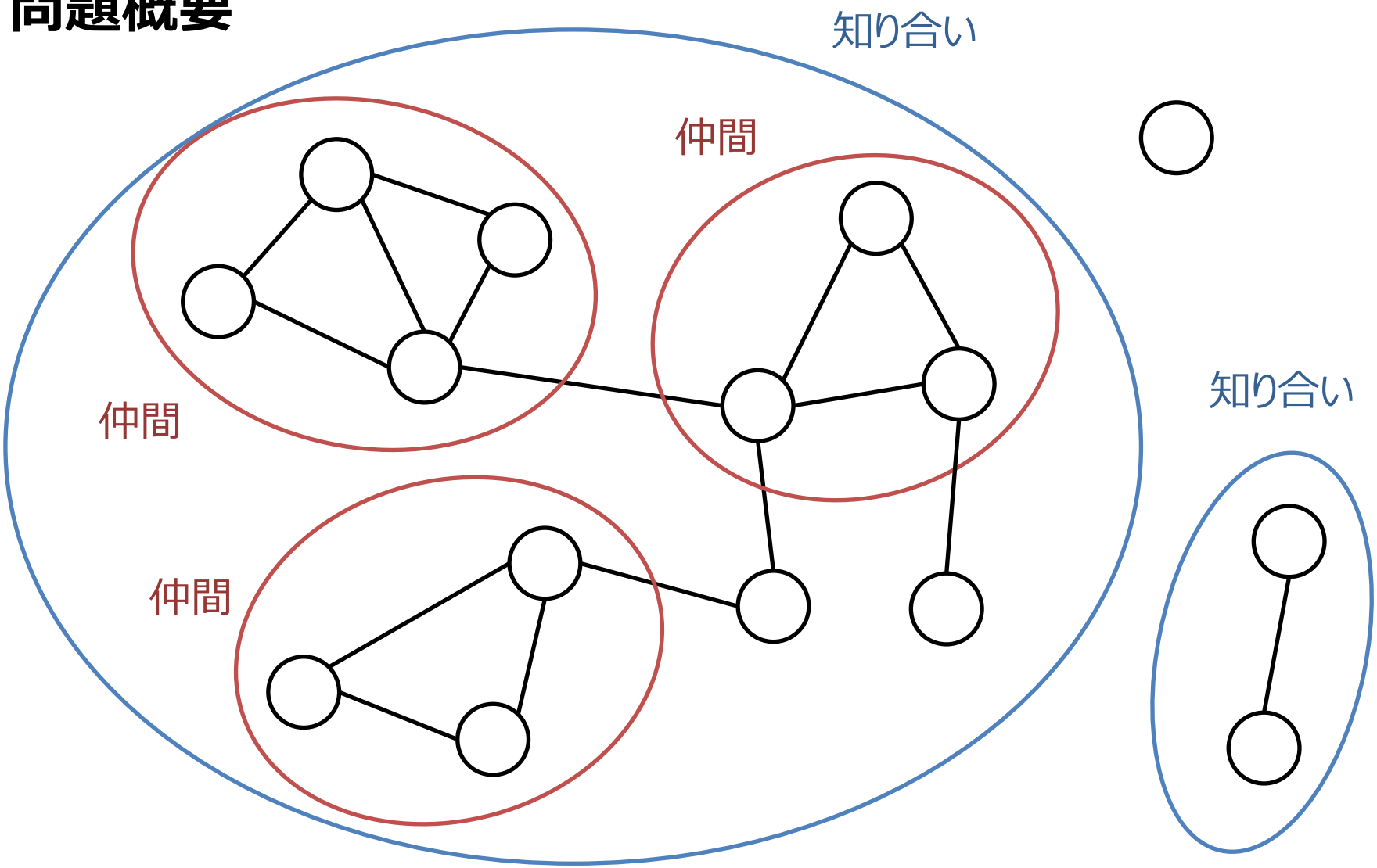
問題概要

友達関係



問１０　パーティ

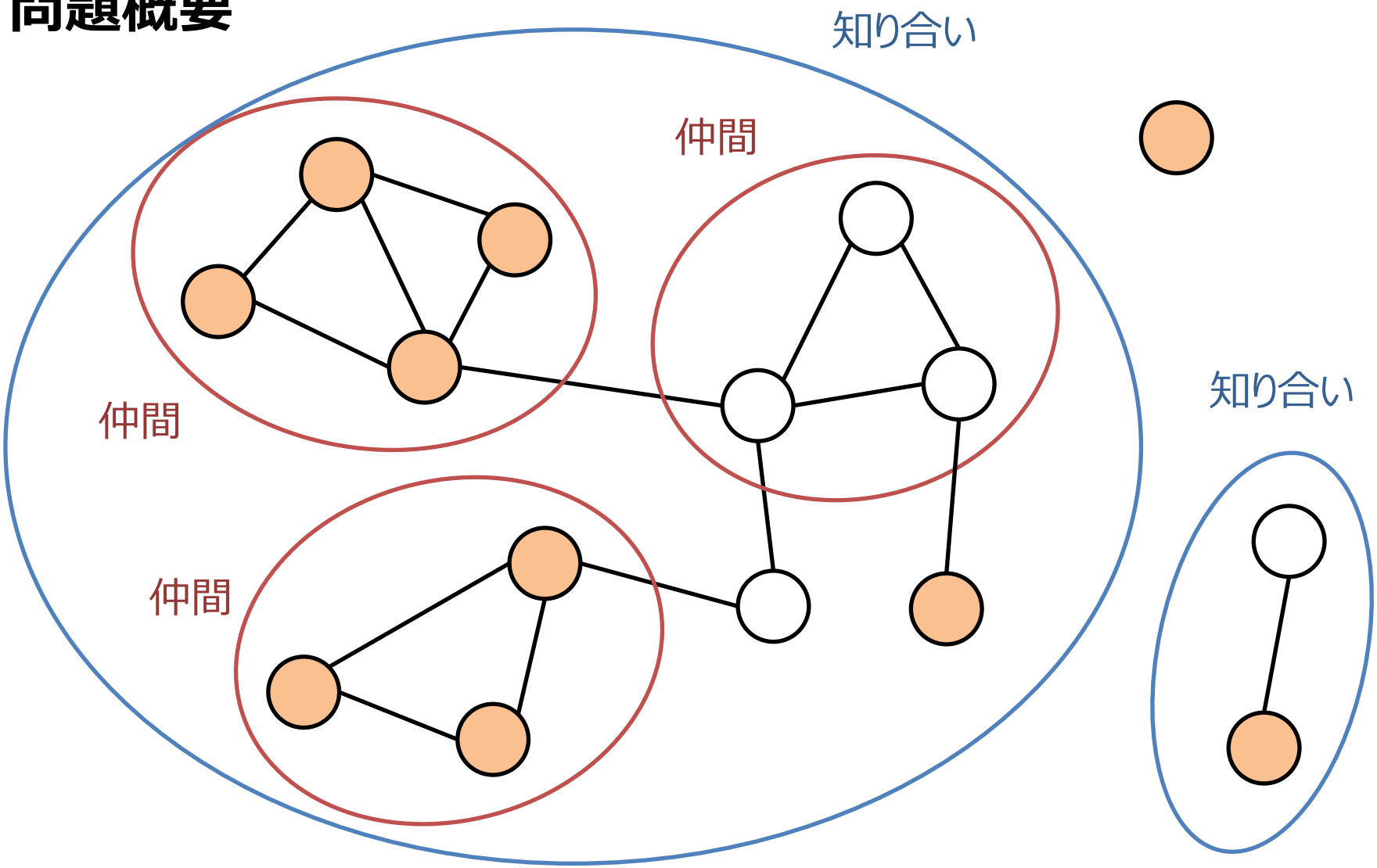
問題概要



問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

問題概要

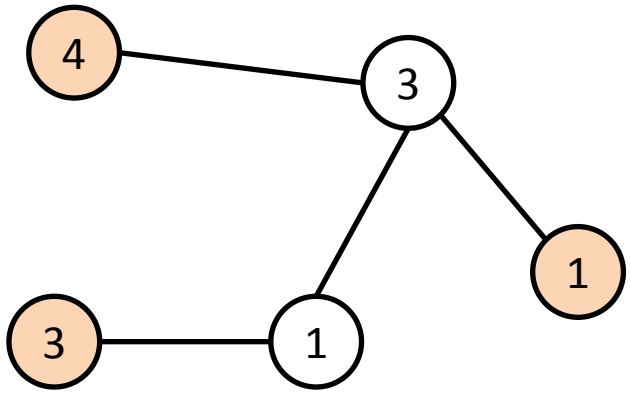


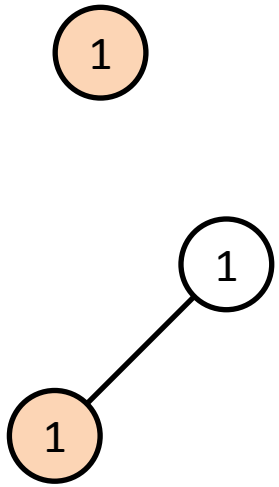
問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

解法

- グラフの二重辺連結成分分解を行う.
- 仲間が１つの節点、グラフの「橋」が辺となるような、森（木の集まり）ができる.
  - グラフの「橋」とは、グラフの連結成分の辺であって、その辺を削除するとグラフが非連結になる辺である
- 各節点は、重みとしてその中に含まれるクラスメイトの数を持つ.
- 各木に対して、最大重み独立集合を求める.





問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

最大重み独立集合

- rを根とする根付き木を考える.
- 根をuとする部分木Tuを部分問題として、動的計画法で解く.
- 部分問題
  - dp[u][0]: uを含まないTuの最大重み独立集合の解
  - dp[u][1]: uを含むTuの最大重み独立集合の解
- 解は max(dp[r][0], dp[r][1])
- 漸化式
  - uが葉の場合
    - dp[u][0] = 0
    - dp[u][1] = w[u]
  - uが葉でない場合
    - dp[u][0] = u の全ての子vについて max(dp[v][0], dp[v][1]) の和
    - dp[u][1] = w[u] + uの全ての子vのdp[v][0]の和

問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
void  bridges(){
  for ( int i = 0; i < N; i++ ) visited[i] = isRoot[i] = false;
  timer = 1;
  for ( int r = 0; r < N; r++ ) {
    if( !visited[r] ) {
      isRoot[r] = true;
      dfs( r, -1 );
    }
  }

  for ( int i = 0; i < N; i++ ){
    if ( isRoot[i] ) continue;
    if ( lowest[i] == prenum[i] ){
      int s = min(i, parent[i]);
      int t = max(i, parent[i]);
      B.push_back(make_pair(s, t));
      BM[make_pair(s,t)] = true;
    }
  }
}
```

問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
  void dfs( int current, int prev ){
    prenum[current] = lowest[current] = timer;
    timer++;
    visited[current] = true;
    for ( int i = 0; i < G[current].size(); i++ ){
        int next = G[current][i];
        if ( !visited[next] ){
            parent[next] = current;
            dfs( next, current );
            lowest[current] = min( lowest[current], lowest[next] );
        } else if ( next != prev ){
            lowest[current] = min( lowest[current], prenum[next] );
        }
    }
}
```

問１０　パーティ

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
int getMaximumIndependentSet(int r ){
  rec(r, -1);
  return max(dp[r][0], dp[r][1]);
}
```

```
void rec(int u, int p){
  visited[u] = true;

  if ( T[u].size() == 1 && p != -1 ){
    dp[u][0] = 0;
    dp[u][1] = weight[u]; assert( weight[u] != -1 );
    return;
  }

  for ( int i = 0; i < T[u].size(); i++ ){
    int v = T[u][i];
    if ( v != p ) rec(v, u);
  }

  dp[u][1] = weight[u];
  for ( int i = 0; i < T[u].size(); i++ ){
    int v = T[u][i];
    if ( v != p ){
      dp[u][0] += max(dp[v][0], dp[v][1]);
      dp[u][1] += dp[v][0];
    }
  }
}
```

問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

問題概要

- AW×AH個のピクセルから成る写真１とBW×BH個のピクセルから成る写真２が与えられる.
- 各ピクセルは建物の種類と雲を表す文字を持つ(建物は英数字、雲は?).
- 写真1の範囲は写真2のどこかに写っている.
- 写真2が写真1に当てはまる可能性がある位置の個数を求める.
- ただし、雲の下に何があるかわからないので、雲はすべてのものに当てはまる.
- 1≦AW, AH≦700,1≦BW,BH≦100.

AW=9, AH=6, BW=BH=4のときの例：

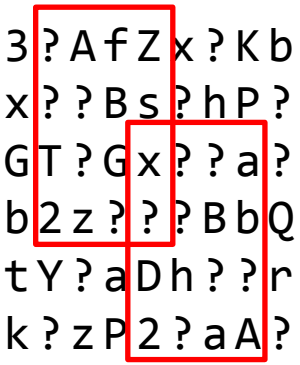


写真1

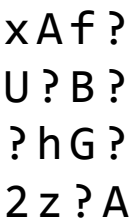


写真2

総評

- 正答数 0
- 2次元の曖昧な文字列マッチング

問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

考察

- 基本的には２次元の文字列マッチング(写真1がテキスト、写真2がパターン).
- ただし、「?」はすべての文字にマッチする(don't care symbol).
- そのため、２次元マッチングでよく使われるAho-Corasick法、ローリングハッシュは使えない.
- 一般に、すべての文字にマッチする文字を含む文字列マッチングでは、テキストの長さn、パターンの長さmの積 nm の比較が必ず必要になることが知られている.

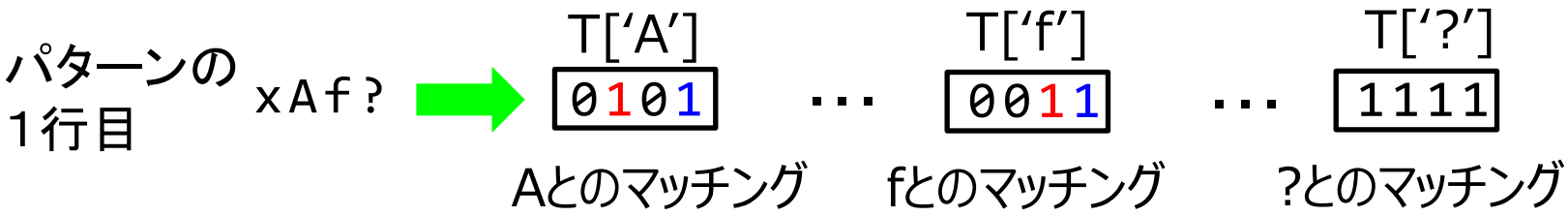
  ⇒ 基本的には総当たりで解くしかない.
- AW・AH・BW・BHの上限は700×700×100×100=49×10^8.
  ⇒ 単純な総当たりだと時間制限.
- 問題をビットで表し、ビット並列を利用することで高速化を図れる.
- long long intを使って64倍速高速化⇒約7600万

問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

解法

- 各行の処理をビット並列(long long intを使うので64倍速).
- パターンの各行に対して前処理を行う.
- 各英数字と「?」 cについて、パターンの行とcとのマッチングを表すビット列T[c]を作り、long long int型の配列に格納(長さBW/64+1).
- 各ビットの値は、パターン中の文字がcか?なら1、それ以外は0.
- ただし、cが「?」のときはすべて1.



（左端が最下位ビット、右端が最上位ビット）

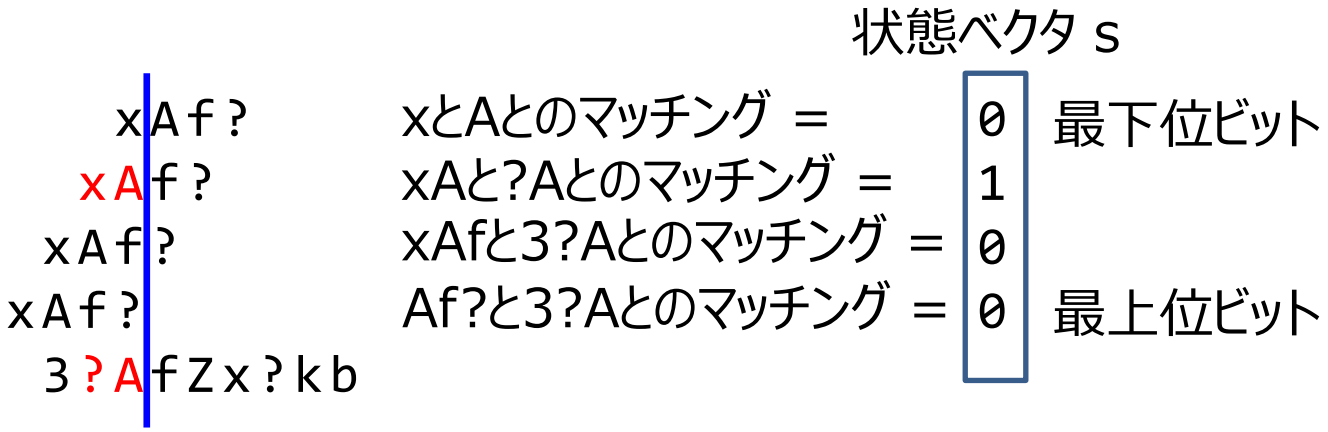
問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

解法

- パターンの各行について、テキストの部分文字列とパターンの先頭部分とのマッチング結果を、状態ベクタと呼ばれる列に保存.
- 状態ベクタの各要素の値は、対応するマッチングが成功なら1、失敗なら0.

テキストの３文字目まで比較したときの状況

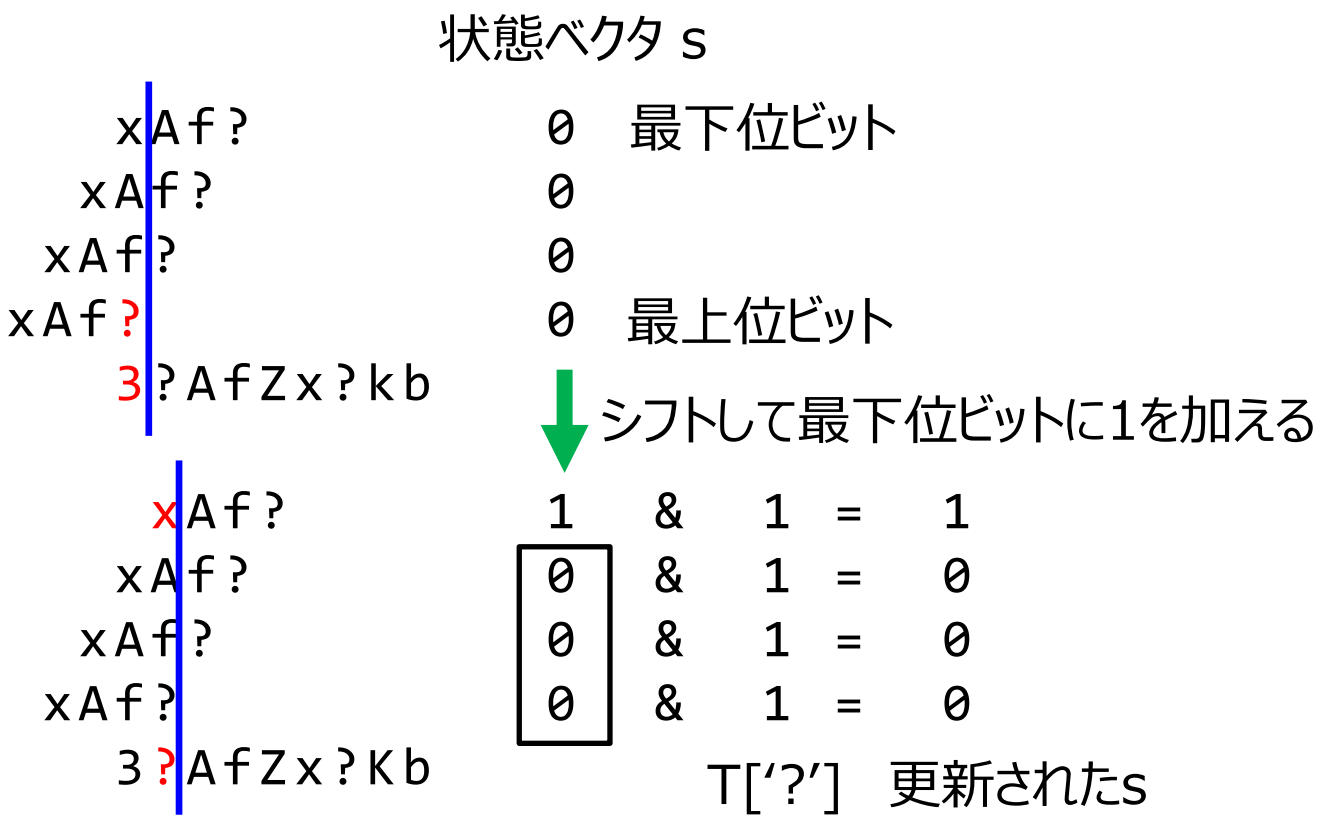


- 状態ベクタの各要素の初期値は0.
- 状態ベクタは long long int の配列（長さはBW/64+1)で表す。

問１１　航空写真

解法

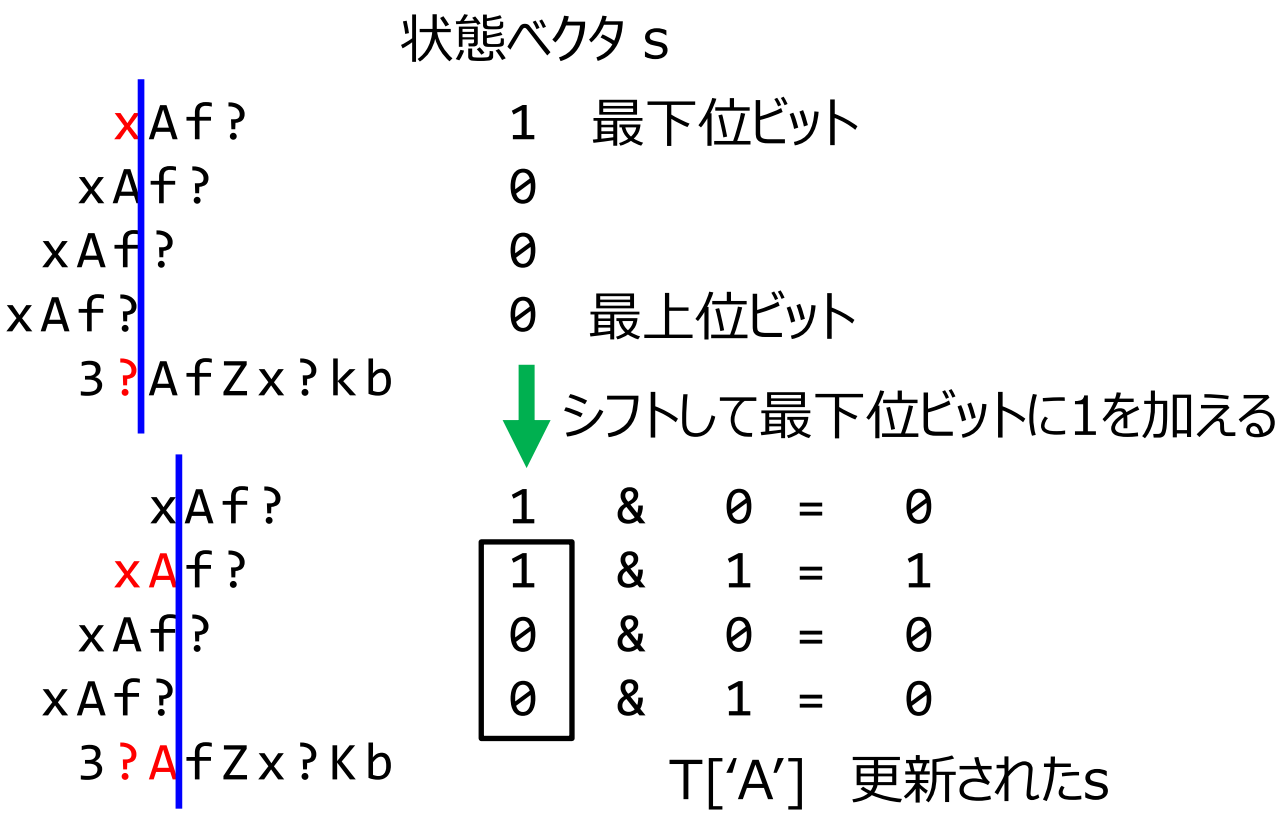
- テキストの次の文字cと比較するとき、状態ベクタを１ビットシフトし、最下位ビットに1を加える.
- T[c]と状態ベクタとの論理積の結果で、状態ベクタを更新する.



問１１　航空写真

解法

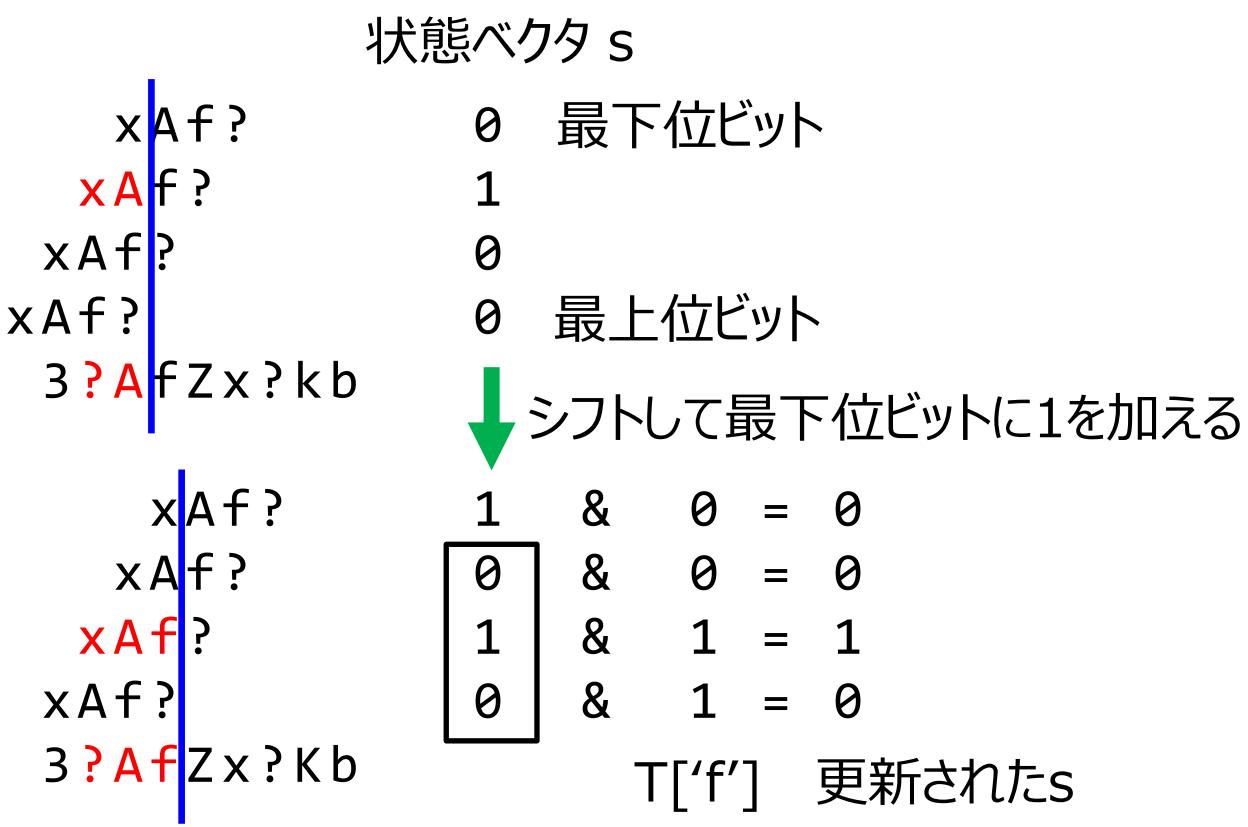
- テキストの次の文字cと比較するとき、状態ベクタを１ビットシフトし、最下位ビットに1を加える.
- T[c]と状態ベクタとの論理積の結果で、状態ベクタを更新する.



問１１　航空写真

解法

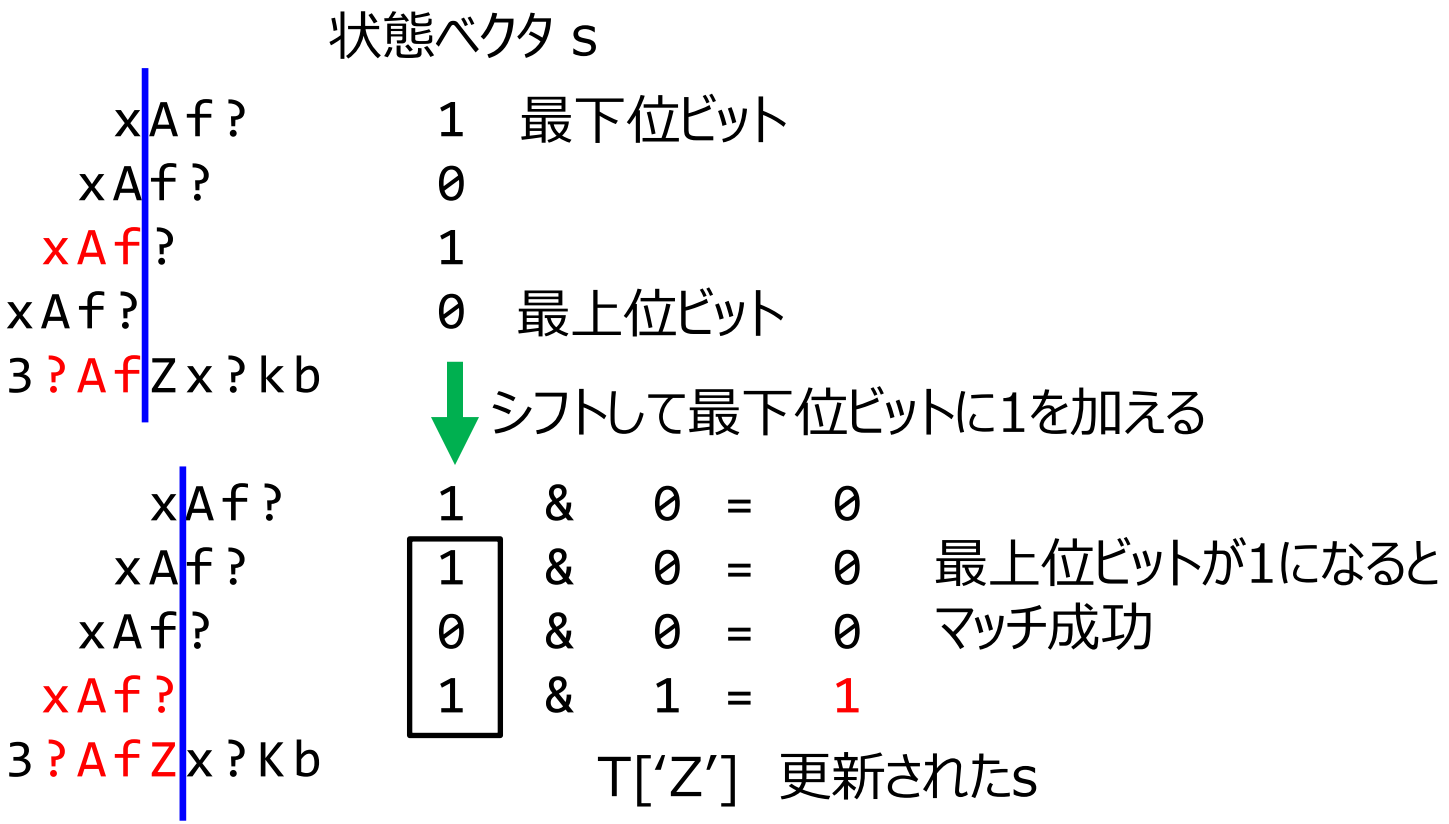
- テキストの次の文字cと比較するとき、状態ベクタを１ビットシフトし、最下位ビットに1を加える.
- T[c]と状態ベクタとの論理積の結果で、状態ベクタを更新する.



問１１　航空写真

解法

- テキストの次の文字cと比較するとき、状態ベクタを１ビットシフトし、最下位ビットに1を加える.
- T[c]と状態ベクタとの論理積の結果で、状態ベクタを更新する.



問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

解法

- シフトして論理積をとるので、Shift-and アルゴリズムと呼ばれる.
- 論理和で行うこともできるのでShift-or アルゴリズムとも呼ばれる.

問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
main(){
  cin >> AW >> AH >> BW >> BH;
  for ( int i = 0; i < AH; i++ ) scanf("%s", A[i]);
  for ( int i = 0; i < AH; i++ ) scanf("%s", B[i]);

  makeMask();
  int ans = 0;
  for ( int h = 0; h < AH; h++ ){
    for ( int i = 0; i < BH; i++ )
      for ( int j = 0; j < K; j++ ) s[i][j] = 0;
    for ( int w = 0; w < AW; w++ ){
      int ok = 0;
      for ( int r = 0; r < BH; r++){
          if ( h + r >= AH ) continue;
          for ( int k = K-1; k >= 0; k-- ){
            int ca = (s[r][k-1] & (1LL<<(BIT-1)))>0?1:0;
            if ( k == 0 ) ca = 1;
            s[r][k] = ((s[r][k] << 1) | ca ) & T[r][k][A[h+r][w]];
          }
          if ( s[r][K-1] & Y ) ok++;
      }
      if ( ok == BH ) ans++;
    }
  }
  cout << ans << endl;
}
```

問１１　航空写真

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
void makeMask(){
  R = BW%BIT;
  K = BW/BIT + (R>0?1:0);
  Y = (1LL<<(R + (BIT-1))%BIT);
  string L = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789?";
  for ( int r = 0; r < BH; r++ ){
    for ( int c = 0; c < L.size(); c++ ){
      for ( int k = 0; k < K; k++ ){
        for ( int b = 0; b < BIT; b++ ){
          if ( k*BIT+b >= BW ) continue;
          if ( B[r][k*BIT+b] == L[c] || L[c] == '?' || B[r][k*BIT+b] == '?'){
            T[r][k][L[c]] += (1LL<<b);
          }
        }
      }
    }
  }
}
```

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

問題概要

- 長さがそれぞれa_iであるN本の棒が与えられる.
- そのうちのM本はb_iのところで直角に折れ曲がっており、(N-M)本は真っすぐである.
- 折れ曲がった棒の中からＸ本、真っすぐな棒の中からＹ本選んで作ることのできる直方体の形の種類の数を求めよ.
- $2X + Y = 12$, $N \leqq 6000$, $a_i \leqq 6000$, $1 \leqq b_i \leqq 3000$, $1 \leqq a_i - b_i \leqq 3000$

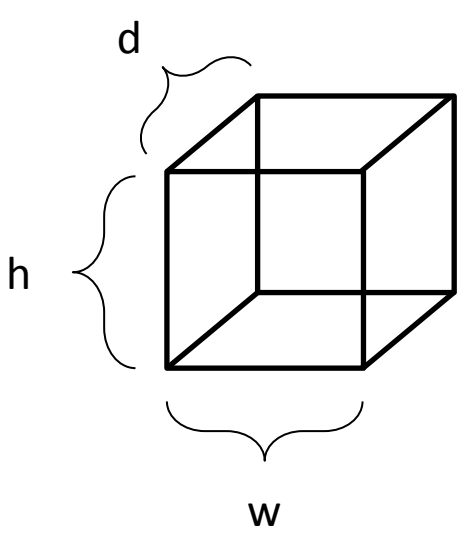
総評

- 正答数 0
- 漏れがないように考察することが重要な問題．場合分けが多い.
- 実装力が問われる問題.

問１２　鉄の棒

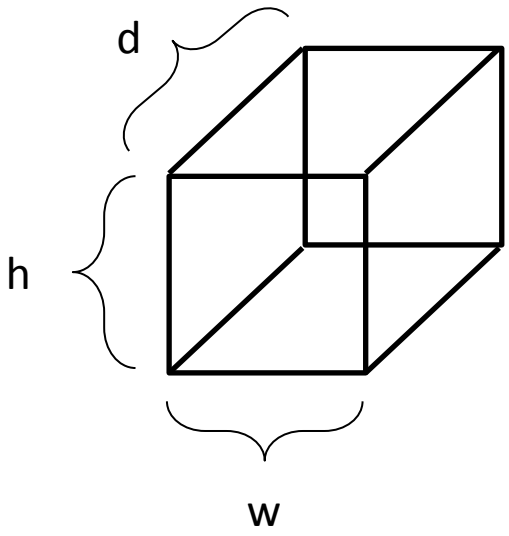
解法

- 場合分けを行い、可能な組み合わせを数え上げる.
- 制約に注意し、高速化を図る.



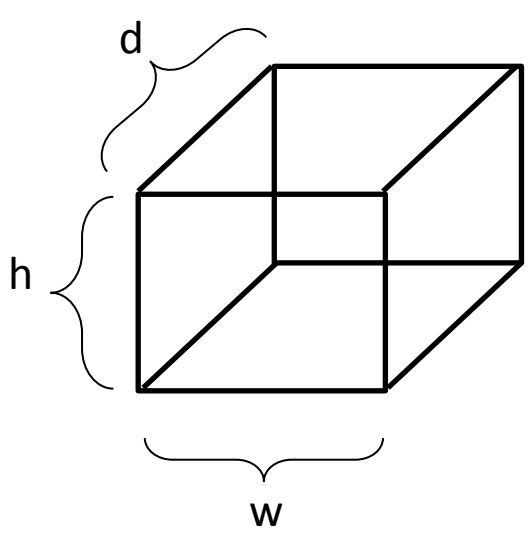
h = w = d

立方体



h = w < d
h = w > d

直方体①



h < w < d

直方体②

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

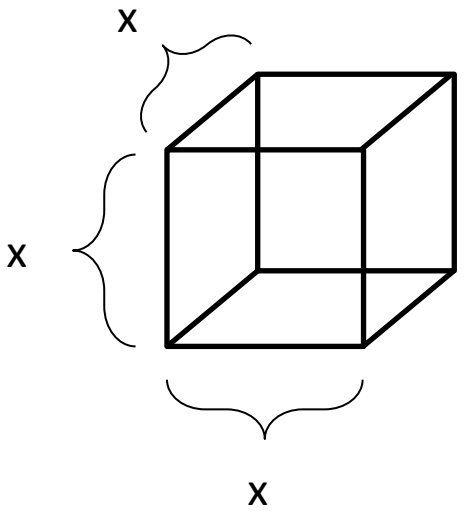
前処理

- 棒の長さの最大値をL_MAXとする.
- 長さが(a+b)cm で a cm のところで折れ曲がった棒の数を L[a][b]に記録.
- 長さが a cm の真っすぐな棒の数をI[a]に記録.

問１２　鉄の棒

立方体

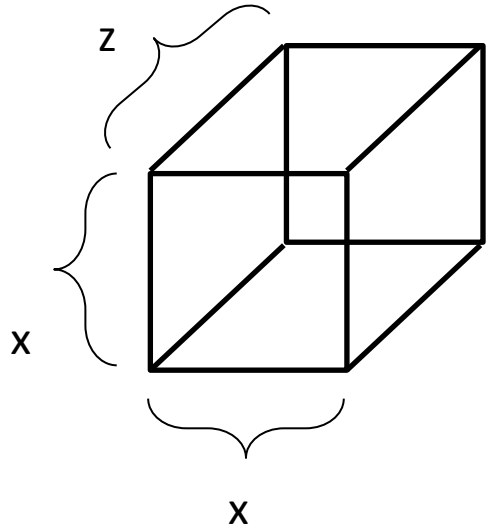
- 一辺の長さ x を 1 からL_MAXまで調べる.
  - L[x][x] ≧ X かつ I[x]x ≧ Y →　カウント
  - ただし、制約より、xがL_MAX/2より大きい場合は、真っすぐな棒でしか作れない
    - X = 0 かつ　L[x] ≧ Y →　カウント
- O(L_MAX).



問１２　鉄の棒

直方体①

- 正方形の一辺の長さ x を１からL_MAX/2まで、残りの１辺の長さzを１からL_MAXまで調べる.
- O(L_MAX^2).

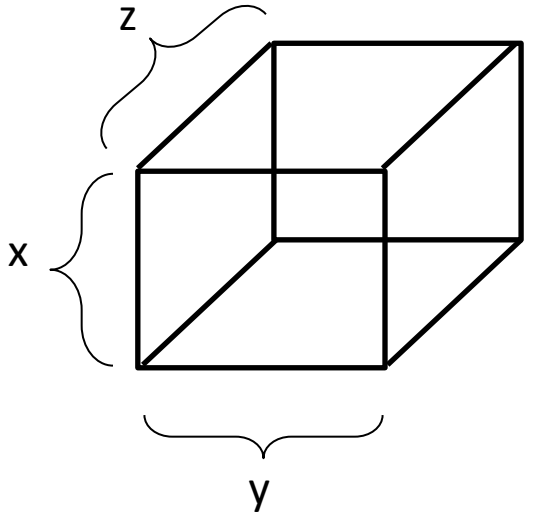


X	(Lxx, Lxz)
0	(0, 0)
1	(0, 1), (1, 0)
2	(0, 2), (2, 0) (1, 1)
3	(0, 3), (3, 0) (1, 2), (2, 1)
4	(0, 4), (4, 0) (1, 3), (3, 1) (2, 2)
5	(2, 3), (3, 2) (1, 4)
6	(2, 4)

Lxx: (x + x) cm の曲がった棒の数
Lxz: (x + z) cm の曲がった棒の数
Ix: x cm の真っすぐな棒
Iz: z cm の真っすぐな棒
Ix = 8 − Lxx*2 − Lxz
Iz = 4 − Lxz

L[x][x] ≧ Lxx かつ L[x][z] ≧ Lxz かつ I[x] ≧ Ix かつ I[z] ≧ Iz → カウント

問１２　鉄の棒



直方体②

- X = 0 の場合
  - I[x] ≧ 4 である x の数を数える：n
  - ${}_nC_3$ をカウント
- X ≧ 1 の場合
  - 曲がった棒を１つ決め打ちする
  - 曲がった棒の種類(x, y)すべてについて、残りのzを１からL_MAXまで調べる
  - 重複しないように、一度数えた形を記録する
    - ハッシュ 等を利用する
- O(N × L_MAX)

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

直方体②

- 曲がった棒の種類(x, y)すべてについて、残りのzを１からL_MAXまで調べる

X	(Lxx, Lxz, Lyz)
1	(0, 0, 1)の順列
2	(0, 0, 2)の順列 (0, 1, 1)の順列
3	(0, 0, 3)の順列 (0, 1, 2)の順列 (1, 1, 1)
4	(0, 0, 4)の順列 (0, 1, 3)の順列 (0, 2, 2)の順列 (1, 1, 2)の順列
5	(1, 1, 3)の順列 (1, 2, 2)の順列
6	(2, 2, 2)

Lxx: (x + x) cm の曲がった棒の数
Lxz: (x + z) cm の曲がった棒の数
Lyz: (y + z) cm の曲がった棒の数
Ix: x cm の真っすぐな棒
Iy: y cm の真っすぐな棒
Iz: z cm の真っすぐな棒
Ix = 4 − Lxy − Lxz
Iy = 4 − Lxy - Lyz
Iz = 4 − Lxz - Lyz

L[x][y] ≧ Lxy かつ L[x][z] ≧ Lxz かつ L[y][z] ≧ Lyz かつ
I[x] ≧ Ix かつ I[y] ≧ Iy かつ I[z] ≧ Iz　→　カウント

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
main(){
  int b;
  cin >> N >> M >> X >> Y;

  for ( int i = 0; i < N; i++ ) cin >> A[i];

  set<pair<int, int> > LS;
  for ( int i = 0; i < M; i++ ){
    cin >> b;
    int r = A[i]-b;
    L[r][b]++; L[b][r]++;
    if ( b == r ) L[b][r]--;
    LS.insert(make_pair(min(r, b), max(r, b)));
  }
  for ( int i = M; i < N; i++ ) I[A[i]]++;

  for (set<pair<int, int> >::iterator it = LS.begin(); it != LS.end(); it++ )
    LV.push_back(*it);

  ans = 0;
  cube();
  prism();
  cuboid();

  cout << ans << endl;
}
```

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
void cuboid(){
  if ( X == 0 ){
    cuboidBar();
    return;
  }

  unordered_set<long long> vis;

  for ( int i = 0; i < LV.size(); i++ ){
    int x = LV[i].first;
    int y = LV[i].second;
    if ( x == y ) continue;
    for ( int z = 1; z <= S_MAX; z++ ) {
      if ( x == z || y == z ) continue;
      int a = x;
      int b = y;
      int c = z;
      if ( a > b ) swap(a,b);
      if ( b > c ) swap(b, c);
      if ( a > b ) swap(a,b);
      long long code = a*100000000+b*10000+c;
      if ( valid(x, y, z) ) vis.insert(code);
    }
  }
  ans += vis.size();
}
```

```
void cuboidBar(){
  long long cnt = 0;
  for ( int i = 1; i <= S_MAX; i++ )
    if ( I[i] >= 4 ) cnt++;
  if ( cnt < 3 ) return;
  ans += cnt*(cnt-1)*(cnt-2)/6;
}
```

問１２　鉄の棒

AIZU
THE UNIVERSITY OF AIZU

解答例（C++）

```
bool check(int x, int y, int z, int Lxy, int Lxz, int Lyz ){
  if ( Lxy == 0 ) return false;
  int Ix = 4 - Lxy - Lxz;
  int Iy = 4 - Lxy - Lyz;
  int Iz = 4 - Lxz - Lyz;
  if ( Lxy*2 + Lxz*2 + Lyz*2 + Ix + Iy + Iz != 12 ) return false;
  ...
  return L[x][y] >= Lxy && L[x][z] >= Lxz && L[y][z] >= Lyz &&
         I[x] >= Ix && I[y] >= Iy && I[z] >= Iz;
}

bool valid(int x, int y, int z){
  for ( int a = 0; a <= 4; a++ ){
    for (int b = 0; b <= 4; b++ ){
      int c = X - a - b;
      if ( c < 0 ) continue;
      if ( X != a + b + c) continue;
      if ( X == 5 && ((a==0 || b==0 || c==0 )||(a==4 || b==4 || c==4 ))) continue;
      if ( X == 6 && !(a == 2 && b == 2 && c == 2 ) ) continue;
      if ( check(x, y, z, a, b, c) ) return true;
    }
  }
  return false;
}
```