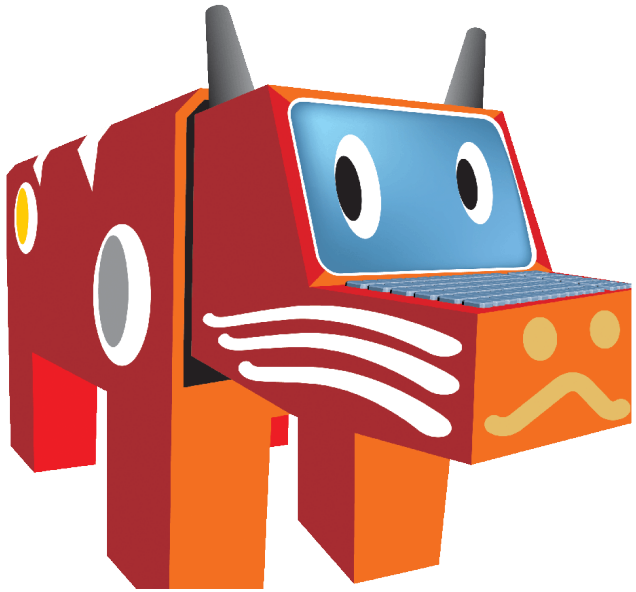
パソコン甲子園2012 予選

プログラミング部門　解説



会津大学
THE UNIVERSITY OF AIZU

To Advance Knowledge for Humanity
THE UNIVERSITY OF AIZU
1993

問1　10問解いたら何点取れる？

問題概要

- 10個の整数を読み込み、その合計を計算するプログラムを作成せよ.
- 与えられるデータは1セットのみ.

講評・解法

- 変数宣言、代入演算、加算、標準入出力処理が行えるかを問う最も基本的な問題である.
- １０回の入力・加算処理で書けるが、解答例のようなループ構造を用いて繰り返し処理を行う方が望ましい.

問1　10問解いたら何点取れる？

Cによる解答例

```
#include<stdio.h>

int main(){
    int i, p, sum = 0;

    for ( i = 0; i < 10; i++ ){
        scanf("%d", &p);
        sum += p;
    }

    printf("%d¥n", sum);

    return 0;
}
```

問1　10問解いたら何点取れる？

C++による解答例

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

int main(){
    int x, sum = 0;

    for ( int i = 0; i < 10; i++ ){
        cin >> x;
        sum += x;
    }

    cout << sum << endl;

    return 0;
}
```

問1　10問解いたら何点取れる？

Javaによる解答例

```
import java.util.*;

class Main{
    private void compute(){
        Scanner sc = new Scanner(System.in);
        int sum = 0;
        for ( int i = 0; i < 10; i++ ){
            int x = sc.nextInt();
            sum += x;
        }
        System.out.println(sum);
    }

    public static void main(String a[]){
        new Main().compute();
    }
}
```

問2　乗車券

問題概要

- 自動改札機の開閉を判断するプログラムを作成せよ.
- 「乗車券」「特急券」「乗車・特急券」の3種類の切符の投入状態が3つのブール値（0または1）で与えられる．乗車券と特急券が同時に投入された場合と、乗車・特急券が投入された場合にOpenと出力せよ.

講評・解法

- 入出力処理に加えて、if文で簡単な分岐処理を行うプログラムが実装できるかが問われている.
- 繰り返し処理を必要としない、最も簡単な部類の問題である.

問2　乗車券

Cによる解答例: 条件分岐

```
#include<stdio.h>

int main(){
    int b1, b2, b3;

    scanf("%d %d %d", &b1, &b2, &b3);

    if ( b1 && b2 || b3 ) printf("Open¥n");
    else printf("Close¥n");

    return 0;
}
```

問2　乗車券

C++による解答例: 3ビットで投入状態を記録する

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

main(){
  int st = 0; // 3ビットで投入状態を記録する
  int b;

  for ( int i = 0; i < 3; i++ ){
      cin >> b;
      if ( b == 1 ) st += (1<<i);
  }

  if ( st >= 3 ) cout << "Open" << endl;
  else cout << "Close" << endl;

}
```

問2　乗車券

Javaによる解答例：　文字列で投入状態を記録

```
class Main{
    public void open(){ System.out.println("Open");}
    public void close(){ System.out.println("Close");}

    public void run(){
        Scanner sc = new Scanner(System.in);
        String state = sc.nextLine();

        if ( state.equals("1 0 0") ) close();
        else if( state.equals("0 1 0") ) close();
        else if( state.equals("1 1 0") ) open();
        else if( state.equals("0 0 1") ) open();
        else if( state.equals("0 0 0") ) close();
    }

    public static void main(String[] a){ new Main().run();}
}
```

問3　家庭菜園

問題概要

- 家庭菜園にn+1本の苗が一列に並んでいるがそのうちの1本は雑草、n本が野菜の苗である．野菜の苗の長さは等差数列になっていることが分かっている．雑草の長さを求めるプログラムを作成せよ．nは4以上100以下である．

お知らせ

- 初めから入力データが等差数列になっている場合は、最初と最後の数値が該当し、解答が一つに定まりません．最初から等差数列になっている入力は与えられないことを問題文に明示すべきだったと認識しております．本件を今後の作題の参考にさせていただき、あいまいさの排除等、明解な問題の作成に努めてまいります．

問3　家庭菜園

講評

- 入力データを配列で管理した上で、簡単なアルゴリズムを考えそれを実装できるかが問われている.
- 典型的なものではなく、問題に特化したアルゴリズムを自ら考える必要がある.

解法

- 数列の1つの要素を雑草と仮定し、それを取り除いた数列が等差数列になっているかどうかを判定すればよい.
- すべての要素について雑草と仮定し、全探索を行う.
- n個の要素について、等差数列であるかの判定をO(n)で行うので、$O(n^2)$のアルゴリズムとなる.

問3　家庭菜園

C++による解答例

```
#include<iostream>
#include<vector>
using namespace std;
static const int N = 100;

bool isSequence(vector<int> v){
  int d = v[1] - v[0];
  for ( int i = 1; i < v.size(); i++ ){
    if ( v[i] -  v[i-1] != d ) return false;
  }
  return true;
}
```

問3　家庭菜園

```
main(){
  vector<int> A;
  int n, x;
  while( cin >> n && n){
    A.resize(n+1);
    for ( int i = 0; i < n+1; i++ ) cin >> A[i];

    for ( int i = 0; i < n+1; i++ ){
      vector<int> tmp = A;
      tmp.erase(tmp.begin() + i);
      if ( isSequence(tmp) ) {
        cout << A[i] << endl;
        break;
      }
    }
  }
}
```

問4　すべての数は6174に通ず

問題概要

- ４桁の数Nに対して、Nの数字を降順に整列した数をL、Nの数字を昇順に整列した数をSとする.
- LからSを引いた値を新しいNとする処理を何回繰り返せばNが6174になるかを計算するプログラムを作成せよ. ただし、すべての桁が同じ数字の場合はNAと出力する.

講評・解法

- 文字列あるいは配列に対する初等的な整列アルゴリズムの適用と、文字・数字・配列間の算術演算、繰り返し処理の制御等を行えるかが問われている.
- 基本的なアルゴリズムの知識に加えてある程度の実装力が必要になる.

問4　すべての数は6174に通ず

C++による解答例：シミュレーションを行う関数

```
int simulate(string d){
  if (d[0] == d[1] && d[1] == d[2] && d[2] == d[3]) return -1;

  string L, S;

  for (int cnt = 0; ; cnt++){
    if ( d == "6174" ) return cnt;
    L = S = d;
    sort(L.begin(), L.end(), greater<char>());
    sort(S.begin(), S.end());
    int x = atoi(L.c_str()) - atoi(S.c_str());
    for ( int i = 0, p = 1000; i < 4; i++, p /= 10){
      d[i] = '0' + x/p;
      x %= p;
    }
  }
}
```

問5　パイプ職人の給料

問題概要

- p本のパイプとそれらをつなぐことができるj本のジョイントが与えられる．i番目のジョイントは、i番目とi+1番目のパイプをつなげることができ、その長さはp(i)+j(i)+p(i+1)になる．
- 「パイプの本数×パイプの長さの総和」の最大値を求めるプログラムを作成せよ．
- 与えられるパイプの本数nの上限は65,000である．

講評

- 最適解を求めるための正しいアルゴリズムを考え、それを実装することができるかが問われてる．

問5　パイプ職人の給料

解法

- パイプの長さの総和は解を計算するために常に必要.
- 使用するジョイントの数をk個とすると、ジョイントの中で長いものからk個使用するのが最適となる.
- kを0からN-1まで決め打ちし、それぞれの場合の最大値を求める.
- ジョイントを長さの降順でソートしておけば、O(NlogN)のアルゴリズムを実装することができる.

問5　パイプ職人の給料

C++による解答例: 最大値を求める関数

```
typedef long long llong;
/**
 * n: パイプの数
 * sump: パイプの長さの総和
 * P: パイプの長さを保持する配列
 * J: ジョイントの長さを保持する配列
 */
llong solve(int n, int sump, int P[], int J[]){
    llong maxv = 0;
    sort(J, J+(n-1), greater<int>());
    for ( int j = 0, t = n, sumj = 0; t >= 1; t--, j++ ){
        llong v = t*(sumj + sump);
        sumj += J[j];
        maxv = max(v, maxv);
    }
    return maxv;
}
```

問6　マヤの大予言

問題概要

- 西暦とマヤ長期暦とを相互変換するプログラムを作成せよ. 西暦はy.m.dの形式、マヤ長期暦はb.ka.t.w.kiの形式で与えられる.
- 1ki＝1日、1w＝20ki、・・のような単位が問題文から得られる. 西暦の日の最大値は、大の月、小の月、うるう年かどうかで変わる.

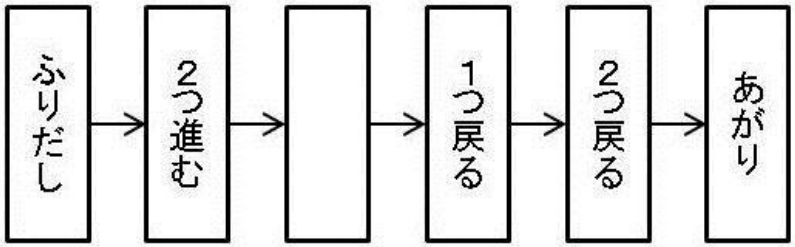
講評

- 日付の変換や閏年の判定など、やや複雑な場合分けが必要になる. 実装力が問われる問題である.

問7　すごろくの判定

問題概要

- すごろくのルーレットが出す数の最大値と各マスに書かれている指示が与えられる.
- このすごろくが「あがり」にたどり着けない場合が生じるかどうかを判定するプログラムを作成せよ. マスの個数の上限は250である.



講評

- 初等的なグラフのアルゴリズムを問題の解法に応用することができるかが問われている.

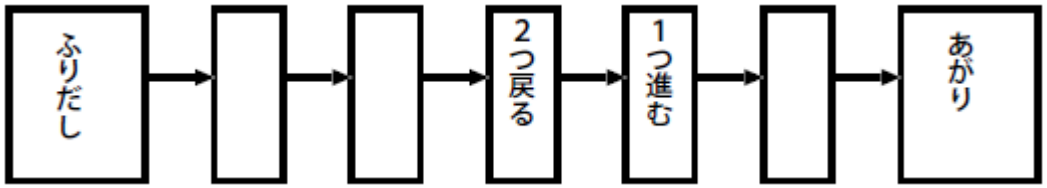
問7　すごろくの判定

解法

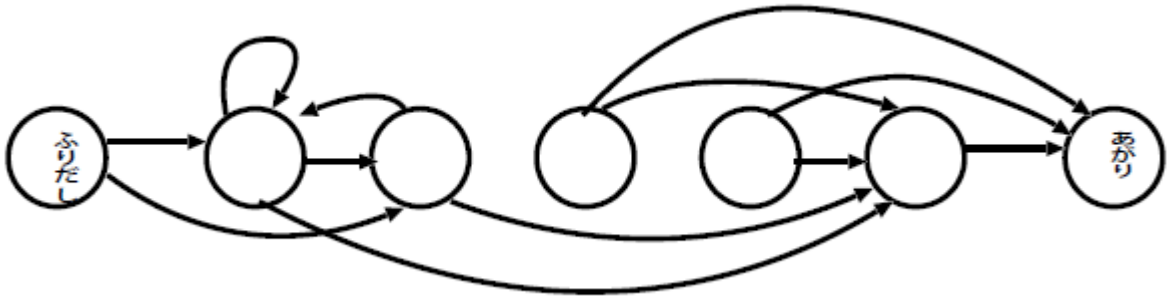
- 与えられた入力を基に、すごろくを有向グラフに変換し、有向グラフの問題に帰着させる.
- グラフの頂点は、各マス目、ふりだし、あがりから成り、各頂点について次に進み得るすべての頂点に辺を出す.

問7　すごろくの判定

解法：有向グラフへの変換（1）

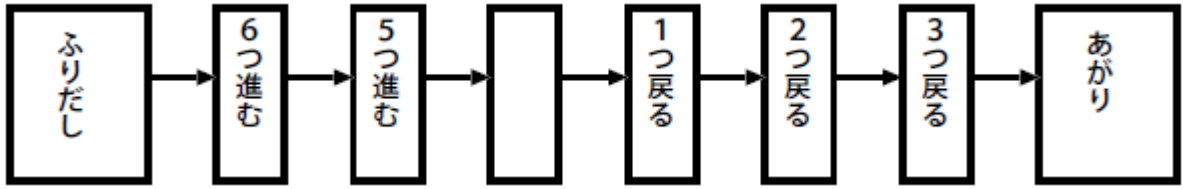


ルーレットが出す数の
最大値が3のとき

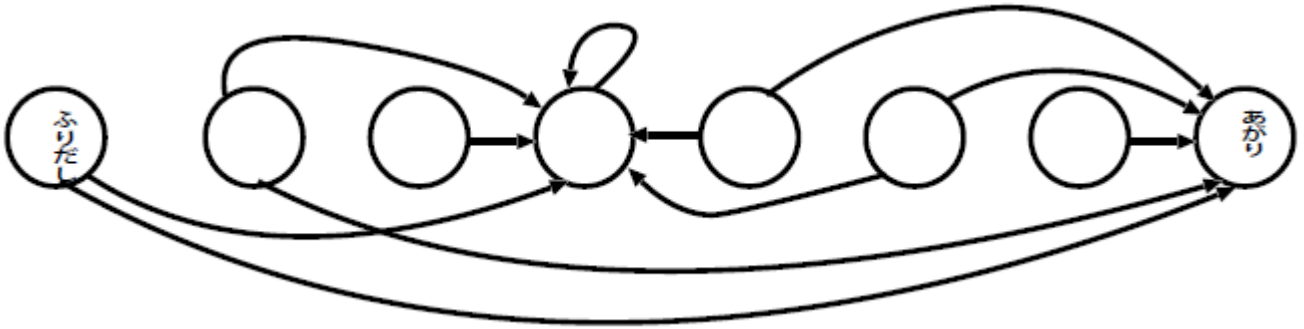


問7　すごろくの判定

解法：有向グラフへの変換（2）



ルーレットが出す数の
最大値が3のとき



問7　すごろくの判定

解法

- 「ふりだし」の状態から到達できるすべての状態について，それらがすべて「あがり」の状態に到達できるならばその時に限り必ずすごろくは「あがり」にいくことができる.
- ワーシャルのアルゴリズム等で、G[u][v]が1ならば頂点uから頂点vに辿り着けるような隣接行列を生成する.
- 「ふりだし」を頂点0、「あがり」を頂点n+1とすると、すべての頂点uについてG[0][u]が 1 かつG[u][n+1]が0であるようなuが存在する場合、「あがり」に辿り着けない場合が生じるすごろくとなる.
- ワーシャルのアルゴリズムを用いた場合、$O(n^3)$のアルゴリズムとなる.

問7　すごろくの判定

C++による解答例：　隣接行列の生成と判定

```
for ( int k = 0; k < n+2; k++ ){
  for ( int i = 0; i < n+2; i++ ){
    for ( int j = 0; j < n+2; j++ ){
      if ( G[i][k] && G[k][j] ) G[i][j] = 1;
    }
  }
}

bool valid = true;

for ( int i = 0; i <= n; i++ ){
  if ( G[0][i] && !G[i][n+1]) valid = false;
}

cout << (valid?"OK":"NG") << endl;
```

問8　ビート・パネル

問題概要

- 4x4のパネル型のボタンがビート音とともに複数光る.
- ボタンが押されたとき光は消え1つのボタンにつき1点スコアが得られる.
- 遊太君は両手の指を使って複数のボタンを押す、c種類の押し方を習得している.
- n回ビート音がなるときのスコアの最大値を求めるプログラムを作成せよ. n、cの上限はともに30とする.

講評

- 最適な組み合わせを求めるための効率的なアルゴリズムの実装（プログラミング手法）が行えるかが問われている.

問8　ビート・パネル

解法

- パネルのボタンの状態は0～$2^{16}-1$の2^{16}種類ある.
- ST[i][j]を状態jのパネルをi番目の押し方で押したときに得られるスコアとし、予め計算しておく.
- dp[i][j]をi回目のビート音が鳴った直後にパネルの状態がjであるときのスコアの最大値とする.
- i+1回目のビート音が鳴った直後に、その前の状態iから得られる状態をnx、 状態nxのパネルをk番目の押し方で押した後の状態をtxとすると、dp[i+1][tx]は max(dp[i+1][tx], dp[i][j] + ST[k][nx])で得られる.
- O(nc2^{16})のアルゴリズムとなる.

問8　ビート・パネル

C++による解答例

```
for ( int i = 0; i < n; i++ ){
  for ( int j = 0; j < (1<<16); j++ ){
    if ( dp[i%2][j] < 0 ) continue;
    int nx = (j | A[i+1]);
    dp[(i+1)%2][nx] = max(dp[(i+1)%2][nx], dp[i%2][j]);
    for( int k = 0; k < c; k++ ){
      int tx = (nx - (nx & T[k]));
      dp[(i+1)%2][tx] = max(dp[(i+1)%2][tx], dp[i%2][j] + ST[k][nx]);
    }
  }
}
```

問9　有限体電卓

問題概要

- 素数pの有限体の中で式の計算を行う電卓プログラムを作成せよ.
- pの上限は46000で、式は加減乗除（それぞれ、＋、－、*、/）とカッコ、0からp-1までの数から構成される.

講評

- 四則演算の構文解析を行うことができる高い実装力が必要.
- 有限体における割り算では、べき乗を高速に計算するためのアルゴリズムの知識が問われている.

問9　有限体電卓

C++による解答例：加減算の構文解析と計算

```
string expr;
int pos, p;
bool valid;

int expression(){
    int value = term();
    while( expr[pos] == '+' || expr[pos] == '-' ){
        if ( expr[pos] == '+' ) {
            pos++; value = (value + term())%p;
        } else {
            pos++;
            int m = p - term();
            value = (value + m)%p;
        }
    }
    return value;
}
```

問9　有限体電卓

C++による解答例：乗除算の構文解析と計算

```
int term(){
    int value = factor();
    while( expr[pos] == '*' || expr[pos] == '/' ){
        if ( expr[pos] == '*' ) {
            pos++;
            value = (value*factor())%p;
        } else {
            pos++;
            int f = factor();
            if ( f == 0 )  valid = false;
            int d = power(f, p-2);
            value = (value*d)%p;
        }
    }
    return value;
}
```

問9　有限体電卓

C++による解答例：数とカッコで囲まれた式の構文解析と計算

```
int factor(){
    int value = 0;
    if ( expr[pos] == '(' ){
        pos++;
        value = expression();
        pos++;
    } else if ( isdigit(expr[pos]) ){
        while( isdigit(expr[pos]) ) {
            value = value*10 + expr[pos] - '0';
            pos++;
        }
    }
    return value;
}
```

問9　有限体電卓

C++による解答例：べき乗の高速計算

```
int power(int x, int n){
    if ( n == 0 ) return 1;
    int res = power(x*x%p, n/2);
    if ( n & 1 ) res = res*x%p;
    return res;
}
```

問10　ねこまっしぐら

問題概要

- 単純多角形の塀に囲まれた屋敷内に猫達が侵入する.
- 猫は屋敷内を直進して多角形の頂点に置かれた餌場にたどり着くことができる.
- 猫達は塀のどこから侵入してくるか分からない.
- 猫が必ず餌にありつけるようにするためには、最低何箇所に餌を置く必要があるかを求めるプログラムを作成せよ.
- 多角形の頂点の数の上限は16とする.

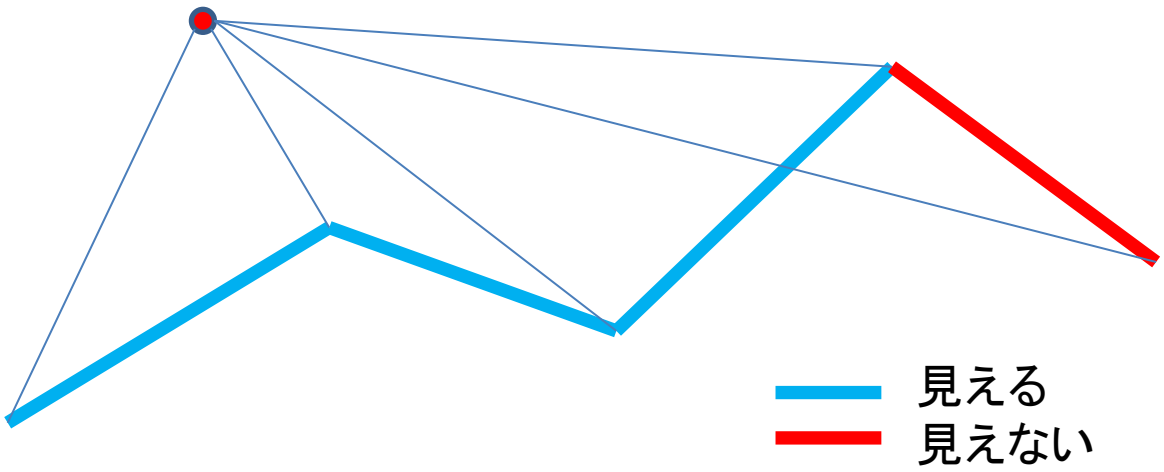
講評

- 問題を典型的な幾何学の問題に落とし、計算幾何学ライブラリ等を駆使しながら実装が行えるかが問われている.

問10　ねこまっしぐら

解法

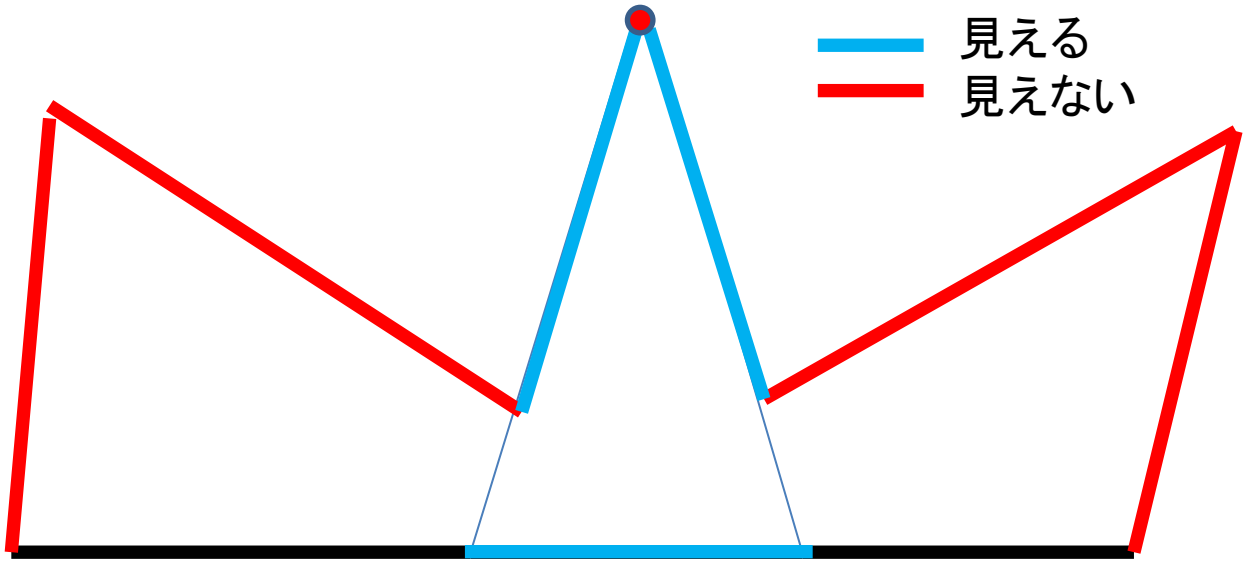
- 各頂点について餌を置く・置かない組み合わせ（2^{16}通り）をすべて試す.
- 餌が置かれた頂点を全て考慮した、各辺の可視性を調べる.
- ある頂点からある辺が見えるかどうかを判定する.



問10　ねこまっしぐら

解法：辺の可視性

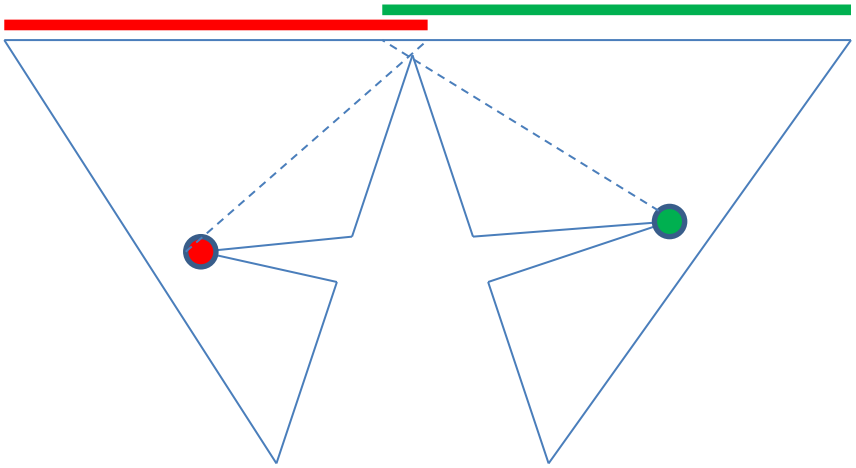
- 多角形を成す1辺の一部だけが見える場合もある.
- その場合には、見える部分だけを可視という情報にする必要がある.



問10　ねこまっしぐら

解法：辺の可視性

- 頂点から「辺全体」の可視性のみだと、下図の多角形は3つの餌（頂点）が必要になる.
- しかし、実際は2点で十分.



問10　ねこまっしぐら

解法：辺の可視性

- 従って、辺をいくつもの線分に分割する必要がある.
- 多角形の2頂点を通る直線と多角形の辺との交点を分割点にすることができる.

解法：線分の可視性

- 頂点と線分の端点からなる三角形の内部がその頂点によって可視か否かを判定する.
- 三角形内に障害物がなく（三角形が多角形の辺と交差しない）、三角形が多角形の内側にある（交差しないうえで、重心が多角形の内部にある）場合、その線分は可視となる.