パソコン甲子園2022

（に・ぜろ・に・に）

全国高等学校パソコンコンクール
プログラミング部門 本選問題解説

問題セット

1 卓球

2 ベコレンジャーカード

3 直方体ボート

4 演算ユニットの接続

5 どろだんご

6 一筆書き

7 素数日の並び

8 ゲームブック

9 縄張りの大きさ

10 数列の復元

11 N数列

12 三角形パズル

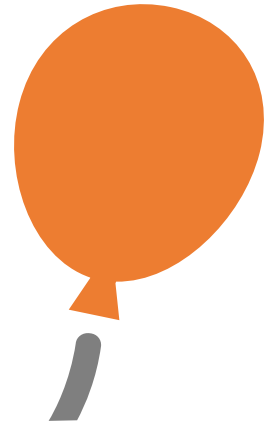
問題セット

#	タイトル	分野	得点	難易度		正解数（凍結前）
				思考	実装	
1	**卓球**	基礎	2	☆	☆	34
2	**ベコレンジャーカード**	基礎	3	★	☆	32
3	**直方体ボート**	探索	4	★	★	33
4	**演算ユニットの接続**	組み合わせ（貪欲法）	6	★☆	★★	25
5	**どろだんご**	組み合わせ（全探索）	7	★★	★★	19
6	**一筆書き**	グラフとデータ構造	9	★★☆	★★★	16
7	**素数の日の並び**	シミュレーション	10	★★	★★★★	1
8	**ゲームブック**	組み合わせ（動的計画法）	11	★★★☆	★★	10
9	**縄張りの大きさ**	文字列検索	11	★★★☆	★★★	6
10	**数列の復元**	組み合わせ（数学）	11	★★★★	★★☆	14
11	**N数列**	データ構造	12	★★★★	★★★★	4
12	**三角形パズル**	計算幾何学	14	★★★★☆	★★★★☆	2

問題１　卓球　　（２点）

正答数:　　34チーム

最初の正解:　　松本秀峰中等教育学校
P22 逆転コードマン
1分57秒 (13:16:57)



問題１　卓球

概要

- 卓球の試合では，両者の点数の和が 6 の倍数になっているときだけ，試合中にタオルを使ってよい.
- 与えられた 2 つの整数pとqについて，$p+q$が 6 の倍数であるかを判定せよ.

問題１　卓球

解法

- 与えられた２つの整数pとqについて，$p+q$が６で割り切れるかを判定する．

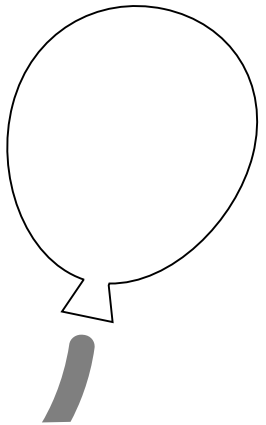
解答例１

```
int main(){
  int p, q;
  cin >> p >> q;
  cout << (((p+q)%6==0)? 1 : 0 ) << endl;
  return 0;
}
```

問題２　ベコレンジャーカード　（３点）

正答数:　32チーム

最初の正解:　静岡県立浜松工業高等学校
P26 味のしない輪ゴム
7分49秒 (13:22:49)



問題２　ベコレンジャーカード

概要

- ４枚のカードに書かれたキャラクターを表す整数 $c_i (1 \leq c_i \leq 4)$ が与えられる.
- このうち１枚のカードを紛失した．どのカードを紛失したかは分からない.
- 残ったカードで同じキャラクターのペア１組を作れるか判定せよ.
    1. どのカードを紛失しても必ずペアが作れる場合
    2. ペアが作れる可能性がない場合
    3. ペアを作れる可能性はあるが必ずではない場合

問題２　ベコレンジャーカード

解法

- 与えられた４枚のカードの状態について，場合分けを行う
  1. どのカードを紛失しても必ずペアが作れる場合
     - 同じキャラクターが３つ以上含まれる　　[1, 1, 1, 4], [4, 4, 4, 4], etc.
     - 同じキャラクターのペアが２つ含まれる　　[1, 1, 2, 2], etc.
  2. ペアが作れる可能性がない場合
     - 全てのキャラクターが異なる　　[1, 2, 3, 4], etc.
  3. ペアを作れる可能性はあるが必ずではない場合
     - 上記の場合以外　　[1, 1, 2, 3], etc.

問題２　ベコレンジャーカード

解答例１

```
sort(c, c+4); // キャラクターを昇順に並べておく

if ( c[0] == c[1] && c[1] == c[2] ||                    // [a, a, a, *]
     c[1] == c[2] && c[2] == c[3] ||                    // [*, a, a, a]
     c[0] == c[1] && c[2] == c[3] ) {                   // [a, a, b, b]
    cout << 1 << endl;
} else if ( c[0] != c[1] && c[1] != c[2] && c[2] != c[3] ){ // [a, b, c, d]
    cout << 2 << endl;
} else {
    cout << 3 << endl;
}
```

問題２　ベコレンジャーカード

解答例２:カードの枚数を数える

最大の枚数	カードの枚数（整列）	
1	1, 1, 1, 1	答えは2
2	2, 1, 1 or 2, 2	2番目に大きい枚数を見て答える. 2番目に大きい枚数が2ならば、最大の枚数は必ず2．この時の答えは1，それ以外の時は答えは3.
3	3, 1	答えは1
4	4	答えは1

```
// カードのカウンタ，枚数と答えの対応
vector<int> n(N,0),res={-1,2,3,1,1};

int main(){
    int i,c;
    for(i=0;i<N;++i) {cin>>c; n[c-1]++;}
    sort(n.begin(),n.end());
    cout << res[n[2]==2?3:n[3]] << endl;

    return 0;
}
```

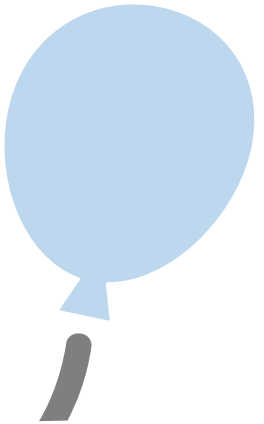
問題３　直方体ボート（４点）

正答数:　33チーム

最初の正解:　愛光高等学校

P34

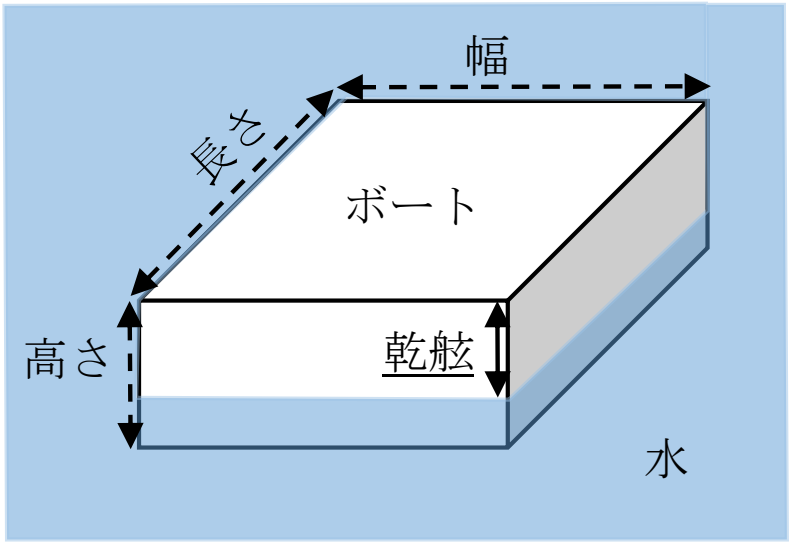
7分44秒 (13:22:44)



問題３　直方体ボート

概要

- 幅が長さの３分の１，高さが幅の２分の１である，重さ50キログラムのボートを作る．
- このボートに積み荷を載せて水に浮かべる．
- 積み荷を載せたボートが水に浮くのは，ボートと積み荷の重さの合計と，ボートによって押しのけられた水の重さがちょうど釣り合うとき（$1{,}000\text{cm}^3$の水の重さは1キログラム）．
- ボートの上面から水面までの距離（乾舷）を30 cm以上にしたい．
- ボートに載せる積み荷の重さ$W(1 \leq W \leq 1{,}000)$が与えられたとき，ボートの長さ，幅，高さがそれぞれ最小何cm（整数値）になるか求めよ．



問題３　直方体ボート

解法

- 高さx，幅$2x$，長さ$6x$のボートを作る．
- $(x-30)(2x)(6x) \geq 1000(W+50)$となる最初の$x$を求める．
- $1 \leq W \leq 1{,}000$より線形探索を応用できる．

問題３　直方体ボート

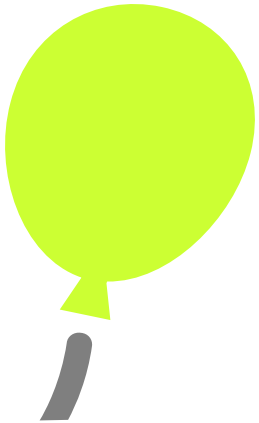
解答例

```
int solve(int W){
  for (int x = 31; ; x++ ){
    if ( 12*x*x*x - 360*x*x >= 1000*(W + 50) ) return x;
  }
}

int main(){
  int W;
  cin >> W;
  int x = solve(W);
  cout << 6*x << " " << 2*x << " " << x << endl;
  return 0;
}
```

問題４　演算装置の接続（６点）

正答数:　25チーム

最初の正解:　弓削商船高等専門学校
P07 ぬるぽ
8分22秒 (13:23:22)



問題４　演算装置の接続

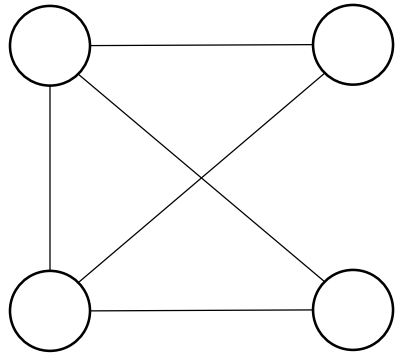
概要

- 接続ポートの数がそれぞれa_iであるN個の演算装置を以下のように接続することはできるか？
  - ✓ 一つのポートは他の演算装置の一つのポートにだけ接続できる.
  - ✓ 全ての接続ポートを利用する.
  - ✓ 一つの演算装置の複数のポートが同じ演算装置に直接接続していない.
- $1 \leq N \leq 200000,\ 1 \leq a_i \leq 200000, a_i$の総和$\leq 200000$.

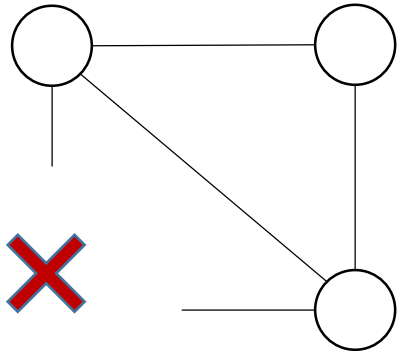
問題４　演算装置の接続

概要

入力例１	出力例１
4 3　3　2　2	Yes



入力例２	出力例２
3 2　3　3	No



問題４　演算装置の接続

素朴な解法

$for\ i \leftarrow 1\ to\ N\ -1$

　$[a_i, a_N]$を降順に整列

　$if\ i + a_i > N$:

　　$print$ **No**

　$a_{i+1},\ a_{i+2}, \ldots, a_{i+a_i}$から1を引く

　$a_i \leftarrow 0$

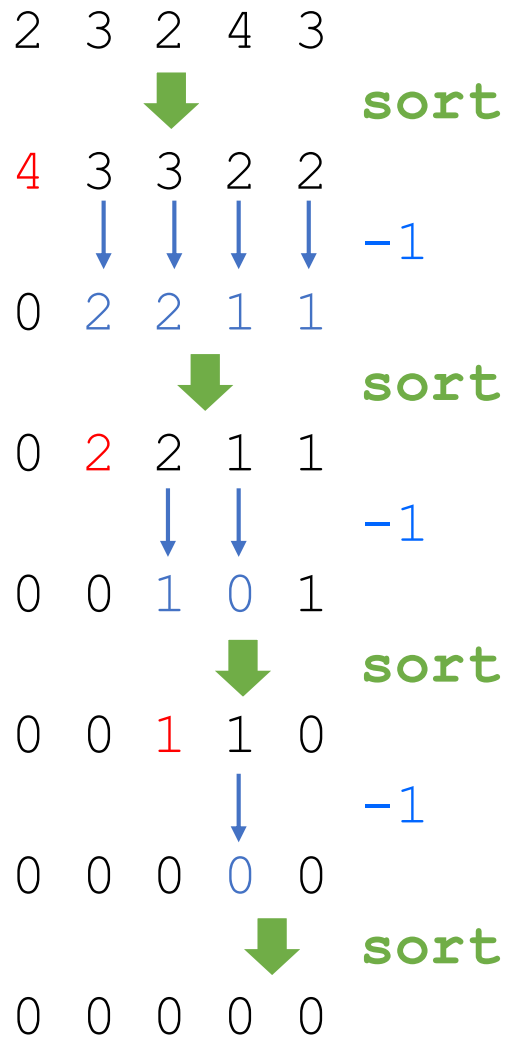
if 全てのiについて$a_i = 0$:

　$print$ **Yes**

$else$:

　$print$ **No**

毎回ソートするため$O(N^2 \log N)$ 時間制限



問題４　演算装置の接続

解法

優先度付きキュー$Q \leftarrow \{a_1, a_2, \ldots, a_N\}$
$while\ Q\ is\ not\ empty$:
　$t \leftarrow Q.pop(\ )$
　$for\ i \leftarrow 1\ to\ t$
　　$if\ Q\ is\ empty$:
　　　$print$ **No**
　　$v \leftarrow Q.pop(\ )$
　　$if\ v - 1 > 0$:
　　　$tmp.push(v - 1)$
　$Q \leftarrow tmp$
$print$ **Yes**

$\sum a_i = S$とすると$O(S \log N)$

問題４　演算装置の接続

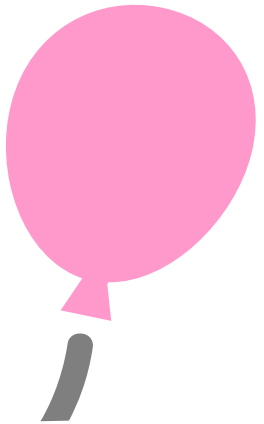
解答例

```
bool solve(){
  priority_queue<int> pq;
  for ( int i = 0; i < N; i++ ) pq.push(A[i]);
  while(!pq.empty()){
    vector<int> tmp;
    int t = pq.top(); pq.pop();
    for ( int i = 0; i < t; i++ ){
      if ( pq.empty() ) return false;
      int v = pq.top(); pq.pop();
      if ( v - 1 > 0 ) tmp.emplace_back(v - 1);
    }
    for ( int v : tmp ) pq.push(v);
  }
  return true;
}
```

問題５　どろだんご（７点）

正答数:　19チーム

最初の正解:　開成高等学校
P08 可変引数テンプレート
40分17秒 (13:55:17)



問題５　どろだんご

概要

- 様々な粘土v_iを持つ土と様々な純度p_iの水をそれぞれ数種類使ってどろだんごを作る．
- 一つのどろだんごを完成させるまで，様々な土と水を選びそれらを順番に塗っていく．同じ種類の土や水を２回以上塗ることはできない．
- どろだんごの品質は，大きさ，柔らかさ，美しさの尺度により決まる．
- 大きさs，柔らかさh，美しさbのどろだんごに土か水を塗ると，その大きさ，柔らかさ，美しさは以下のようになる．
  - 粘度vの土を塗ると，大きさと美しさがそれぞれv，$h \times v$増える．柔らかさは変わらない．
  - 純度pの水を塗ると，柔らかさと美しさがそれぞれp，$s \times p$増える．大きさは変わらない．
- どろだんごの美しさのバリエーションはどれくらいあるか？

問題５　どろだんご

考察

v_i	p_j	s	h	b
		1	**1**	**1**
2		+2		+ (1 × 2)
		3	**1**	**3**
	4		+4	+ (3 × 4)
		3	**5**	**15**
7		+7		+ (5 × 7)
		10	**5**	**50**
	3		+3	+ (10 × 3)
		10	**8**	**80**
	2		+2	+ (10 × 2)
		10	**10**	**100**

- bの値はsとhの値で決まる
- bの値は常に$s \times h$

- sの値はv_iの組み合わせで決まる
- hの値はp_iの組み合わせで決まる

- sとhのそれぞれの全ての組み合わせについて，$s \times h$として得られる数を数える

問題５　どろだんご

解法

- 土の種類の数N $(1 \leq N \leq 10)$，水の種類の数M $(1 \leq M \leq 10)$.
- $2^N \times 2^M$を全探索する.
- 結果をsetで数えて$O(2^{N+M} \log 2^{N+M})$.

問題５　どろだんご

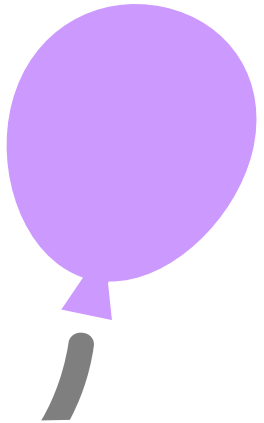
解答例

```
int N, M, V[10+1], P[10+1];
set<ull> cnt;

cin >> N >> M;
for ( int i = 0; i < N; i++ ) cin >> V[i];
for ( int i = 0; i < M; i++ ) cin >> P[i];

for ( int i = 0; i < (1 << N); i++ ){
  for ( int j = 0; j < (1 << M); j++ ){
    ull p = 1, q = 1;
    for ( int k = 0; k < N; k++ )
      if ( (1 << k) & i ) p += V[k];
    for ( int k = 0; k < M; k++ ){
      if ( (1 << k) & j ) q += P[k];
    }
    cnt.insert(p*q);
  }
}

cout << cnt.size() << endl;
```

問題６　一筆書き（６点）

正答数:　16チーム

最初の正解:　渋谷教育学園渋谷高等学校
P06 がんばりやさん
35分30秒 (13:50:30)



問題６　一筆書き

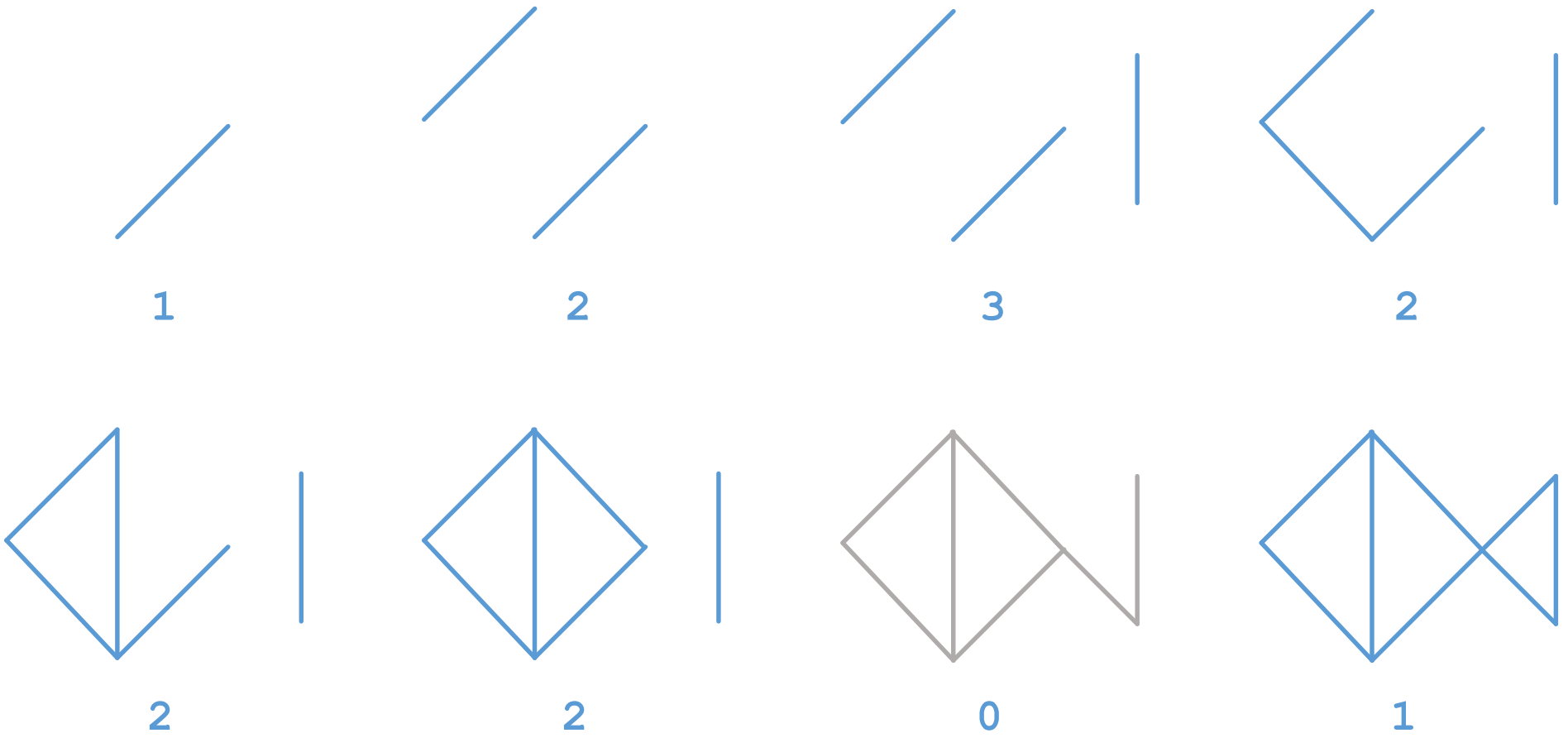
概要

- PCK君が以下のように画用紙に絵を描く.
  - 一回の動作で，１つの線分を描く.
  - 既に描かれている線分に触れる場合は，お互いの端点だけで重なる.
- 連結した線分のグループを１つの図形とする.
- 線分を描く動作の情報が順番に与えられる．動作が終わるごとに，画用紙に含まれる一筆書きが可能な図形の数を報告せよ.
- 線分の数N $(1 \leq N \leq 200{,}000)$.
- 線分の始点の座標$x_1, y_1 (0 \leq x_1, y_1 \leq 10^9)$, 終点の座標$x_2, y_2 (0 \leq x_2, y_2 \leq 10^9)$.

問題６　一筆書き

入出力例

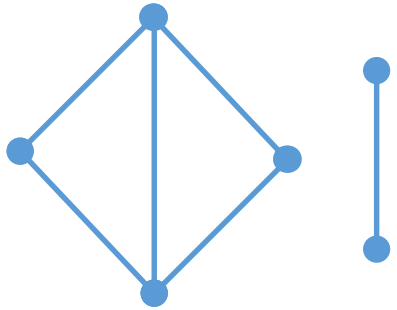
入力例
8 5 0 8 3 5 6 2 3 10 5 10 1 2 3 5 0 5 0 5 6 8 3 5 6 8 3 10 1 10 5 8 3



問題６　一筆書き

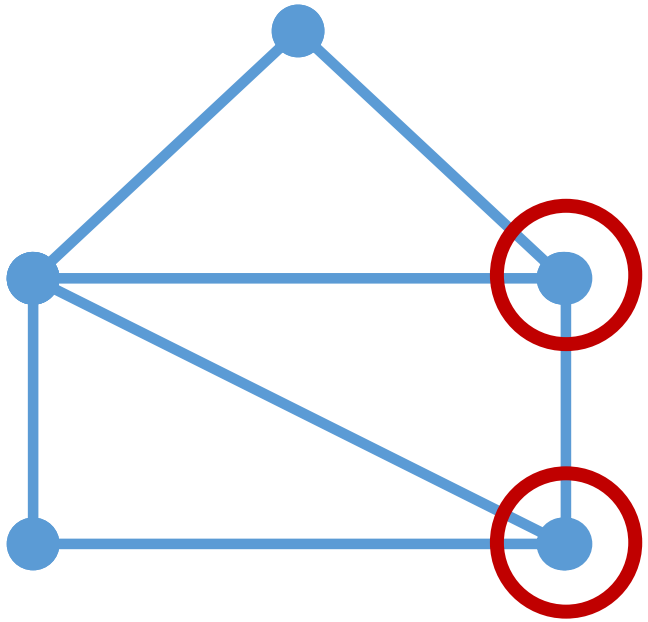
考察

- 問題の制約から，図形はグラフの連結成分として表すことができる.
- 連結成分ごとに，一筆書きが可能かを判定すればよい.

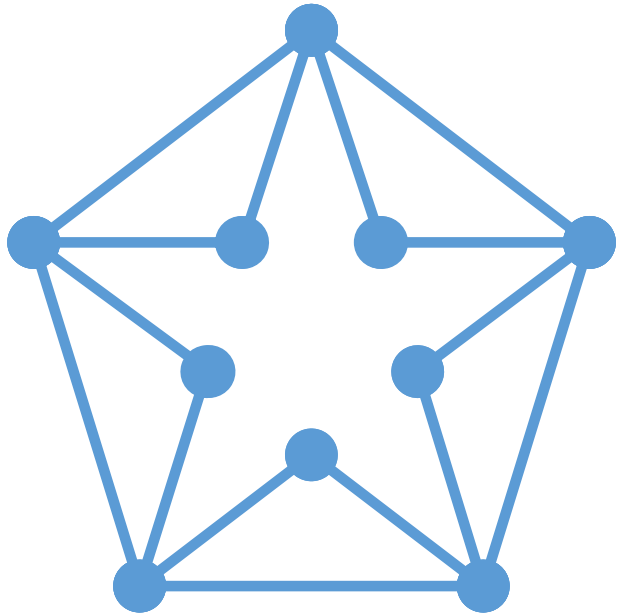


問題６　一筆書き

一筆書きができる条件



グラフの2つの頂点の次数が奇数であり，
残りの頂点の次数が偶数



グラフの全ての頂点の次数が偶数

問題６　一筆書き

前処理

- 座標圧縮: 点の座標　→　頂点番号へ変換

```
int getId(int x, int y){
  if ( M.count(make_pair(x, y)) == 0 ){
    M[make_pair(x, y)] = N++;
  }
  return M[make_pair(x, y)];
}
```

- 入力を線分のリストへ変換する

```
for ( int i = 0; i < Q; i++ ){
    cin >> x1 >> y1 >> x2 >> y2;
    u = getId(x1, y1);
    v = getId(x2, y2);
    edges.emplace_back(make_pair(u, v));
}
```

問題６　一筆書き

素朴な解法

- 線分が追加される度に深さ優先探索（幅優先探索）し，連結成分ごとに奇数の次数をもつ頂点の数を数える．
- $O(N^2)$　→　時間制限

- 更新された連結成分のみ探索し，差分で計算する．
- $O(N^2)$　→　時間制限

問題６　一筆書き

解法

- 連結成分を「互いに素な集合」で管理する.
  - 初期状態で次数が 0 の頂点の扱いに注意.
- Union-find 木を用いて$O(N \log N)$.
  - Rank + 経路圧縮

問題６　一筆書き

解法：実装例

- Union-find 木における各集合ごとに，次数が奇数である頂点の数を保持する
  - $\deg[i]$頂点iの次数
  - $\mathrm{odd}[i]$集合iに含まれる頂点のうち，次数が奇数のものの個数
  - u, v: 追加する線分の端点の頂点番号
  - S_u, S_v: u, vがそれぞれ属する集合（Union-find木における代表の頂点番号）
  - res: 現在一筆書きができる集合の数

問題６　一筆書き

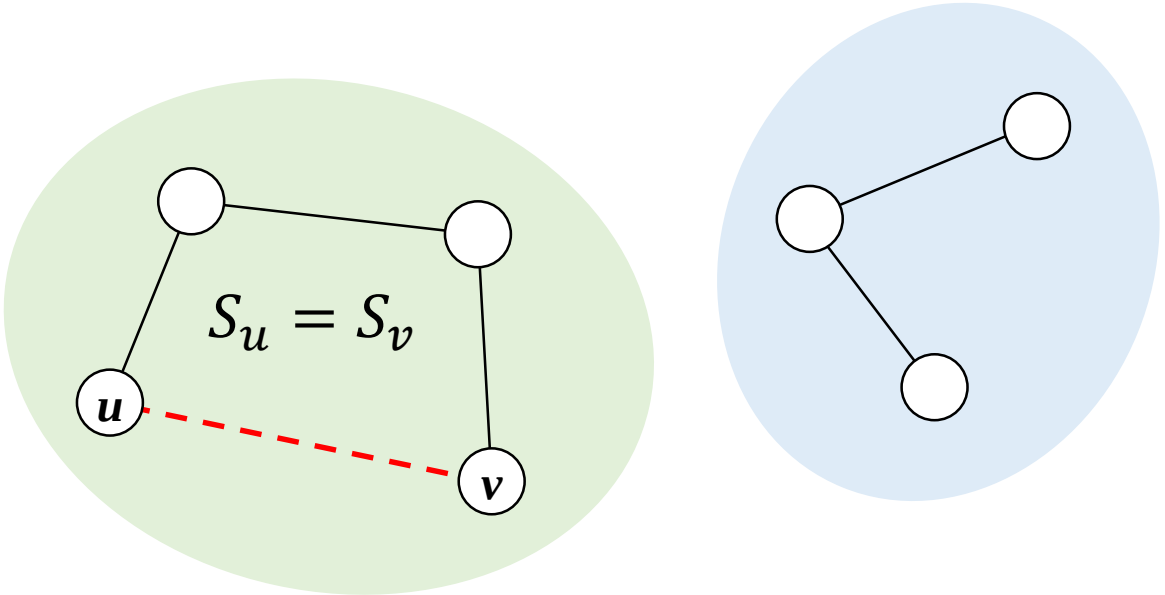
実装例：同じ連結成分内の頂点を繋ぐ

$if\ S_u = S_v$:
　$\deg[u] \leftarrow \deg[u] + 1$
　$if\ \deg[u]$ が偶数:
　　$\mathrm{odd}[S_u] \leftarrow odd[S_u] - 1$
　$else$:
　　$\mathrm{odd}[S_u] \leftarrow odd[S_u] + 1$

　$\deg[v] \leftarrow \deg[v] + 1$
　$if\ \deg[v]$ が偶数:
　　$\mathrm{odd}[S_v] \leftarrow odd[S_v] - 1$
　$else$:
　　$\mathrm{odd}[S_v] \leftarrow odd[S_v] + 1$

$if\ S_u$の一筆書きが不可能→可能に変化:
　$res \leftarrow res + 1$

$if\ S_u$ の一筆書きが可能→不可能に変化:
　$res \leftarrow res - 1$

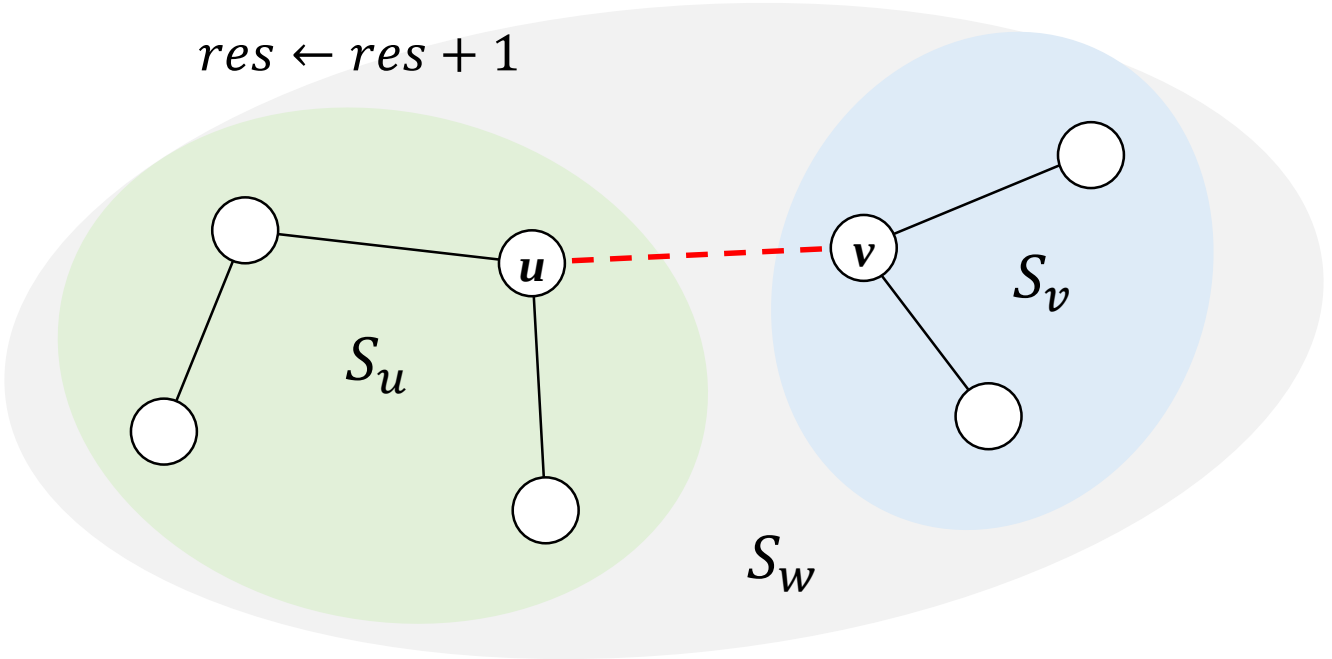


問題６　一筆書き

実装例：異なる連結成分内の頂点を繋ぐ

$if\ S_u \neq S_v$:
　$if\ S_u$の一筆書きが可能:　$res \leftarrow res - 1$
　$if\ S_v$の一筆書きが可能:　$res \leftarrow res - 1$
　$newOdd \leftarrow odd[S_u] + odd[S_v]$
　$\deg[u] \leftarrow \deg[u] + 1$
　$if\ \deg[u]$が偶数:
　　$newOdd \leftarrow newOdd - 1$
　$else$:
　　$newOdd \leftarrow newOdd + 1$
　$if\ \deg[v]$が偶数:
　　$newOdd \leftarrow newOdd - 1$
　$else$:
　　$newOdd \leftarrow newOdd + 1$

u, v を合併
$S_w \leftarrow findSet(u)$
$odd[S_w] \leftarrow newOdd$
$if\ S_w$ が一筆書き可能:
　$res \leftarrow res + 1$



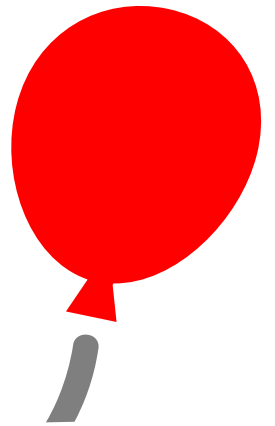
問題７　素数の日の並び（１０点）

正答数:　1チーム

最初の正解:　筑波大学附属駒場高等学校

P04 simasiman

2時間47分35秒 (16:02:35)



問題７　素数の日の並び

概要

- １週を日曜日から始める7日間とする.
- たとえば2022年11月1日が含まれる週から，12月31日が含まれる週までの週の並びを見ると，右の図のようになっている.
- このような週の並びで，日曜日から土曜日までの1週間分の日の並びに，最大でいくつの素数の日が入っているか？
- $y_1 \leq y_2 \leq 100{,}000{,}000.$

2022年11月,12月						
日	月	火	水	木	金	土
30	31	**1**	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	**31**

問題７　素数の日の並び

概要

- 開始年月と終了年月が与えられたとき，その開始年月の1日が含まれる週から，終了年月の最後の日が含まれる週までの期間において，一つの週に現れる素数の日の数の最大値と，素数の日の数が最大になる週の数を求める

例えば，右図の場合，各週の素数の日は以下のようになる．

- 1週目の素数の日は31，2，3，5の4つ．
- 2週目の素数の日は7，11の2つ．
- 3週目の素数の日は13，17，19の3つ．
- 4週目の素数の日は23だけで1つ．
- 5週目の素数の日は29，2，3の3つ．
- 6週目の素数の日は5，7の2つ．
- 7週目の素数の日は11，13，17の3つ．
- 8週目の素数の日は19，23の2つ．
- 9週目の素数の日は29，31の2つ．

⇨ 素数の日の数の最大値は4
そのような週の数は1

2022年11月,12月						
日	月	火	水	木	金	土
30	31	**1**	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	**31**

問題７　素数の日の並び

考察

- 素数2だけは，すぐ隣にある素数3と隣接していて，他の偶数は素数ではないから週の日数が7日なら，2を含む場合に4つになるケースが最大であることがわかる.
- さらに，4つ並ぶ場合は(2,3,5,7)か(31,2,3,5)か(29,31,2,3)か(29,2,3,5)しかないということが言える.

問題７　素数の日の並び

考察

- ある年の初めから，各月初日までの経過日数をmod7で表すと以下のようになる.

閏年でないとき

月	１	２	３	４	５	６	７	８	９	１０	１１	１２
日数	31	28	31	30	31	30	31	31	30	31	30	31
累積	0	+31	+59	+90	+120	+151	+181	+212	+243	+273	+304	+334
Mod7	+0	+3	+3	+6	+1	+4	+6	+2	+5	+0	+3	+5

閏年であるとき

月	１	２	３	４	５	６	７	８	９	１０	１１	１２
日数	31	29	31	30	31	30	31	31	30	31	30	31
累積	0	+31	+60	+91	+121	+152	+182	+213	+244	+274	+305	+335
Mod7	+0	+3	+4	+0	+2	+5	+0	+3	+6	+1	+4	+6

- 累積日数のmod7が，+0から+6まですべてそろうので，どの年にも必ず日付が１２３４５６７となる並びの月が存在する.

問題７　素数の日の並び

考察

- 丸1年分の期間を見れば，１週間に現れる素数の日の数の最大値は必ず４になる.
- 丸1年に満たない期間について，愚直に集計しても高々365日.
- 期間が1年以上なら，年単位で集計することが可能.

問題７　素数の日の並び

解法

- ある年の，素数の日の数が4になる週の数は，1月1日が何曜日なのかと，閏年かどうかで決まる．
- 前処理で，素数の日の数が4になる週の数をカウントしておく．Table[閏年/閏年でない][1月1日の累積日数 mod 7]表を用意して埋める．
- 例えばy1 = 2003, m1 = 9, y2 = 2022, m2 = 11なら，2004年から2021年までは，表からO(y2 – y1)で求まる．2003年9月から12月と，2022年1月から11月までは，愚直にカウントする．
- ただし，重複が出る分を除去する必要がある．
- 計算量O(y2 – y1)．

問題７　素数の日の並び

解法

前処理：

table[2][7];　//[閏年かどうか][1月1日の曜日]

閏年と閏年じゃない年について1/1が月,火,...,日だったとき

その年に素数日4の週がいくつあるかをカウントしておく表

ただし1月の週は前の年の週，12月の週は次の年の週と

重複して数えられる場合があるので重複を除去する

```
for ( int i=0; i<2; ++i ) //閏年かどうか
    for ( int j=0; j<7; ++j ) //j=0 -> 1/1が日曜, j=1 -> 1/1が月曜..
        table[i][j] = 素数の日の数が4になる週を数える
```

問題７　素数の日の並び

解法

//末端処理

$if\ y_1 = y_2$:

　$res \leftarrow y_1$年m_1月からy_2年m_2月まで愚直に求める

$else$:

　$res \leftarrow y_1$年m_1月からy_1年12月まで愚直に求める

　$res \leftarrow y_2$年1月からy_2年m_2まで愚直に求める

//年ごとに処理

$if\ y_2 - y_1 \geq 2$:

　$wd \leftarrow (y_1+1)$年1月1日の曜日

　$for\ y_1+1\ to\ y_2 - 1$:

　　$res \leftarrow table$[閏年かどうか][wd]を加算

　　$wd \leftarrow$ 閏年かどうかによって+365または+366して，mod7を取る

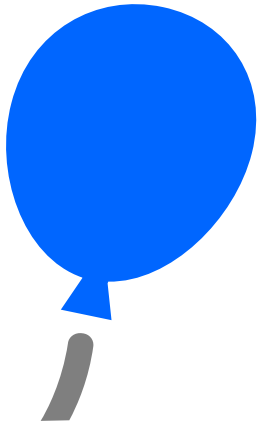
問題８　ゲームブック（１１点）

正答数:　10チーム

最初の正解:　灘高等学校

P01 KobLa

1時間9分33秒 (14:24:33)



問題８　ゲームブック

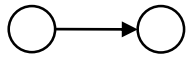
概要

- Nページのゲームブックを作成する.
- 各ページにはいくつかのページ番号を書き込む．以下の条件を全て満たす，ページ番号の書き方が何通りあるか求めよ.
  - ✓各ページには，そのページ番号よりも大きいページ番号を書き込む.
  - ✓1ページ目から，直接またはいくつかのページを経由して，任意のページにたどり着ける.
  - ✓任意のページから，直接またはいくつかのページを経由して，Nページ目にたどり着ける.
- $2 \leq N \leq 500$.
- 答えを998,244,353で割った余りを出力する.

問題８　ゲームブック

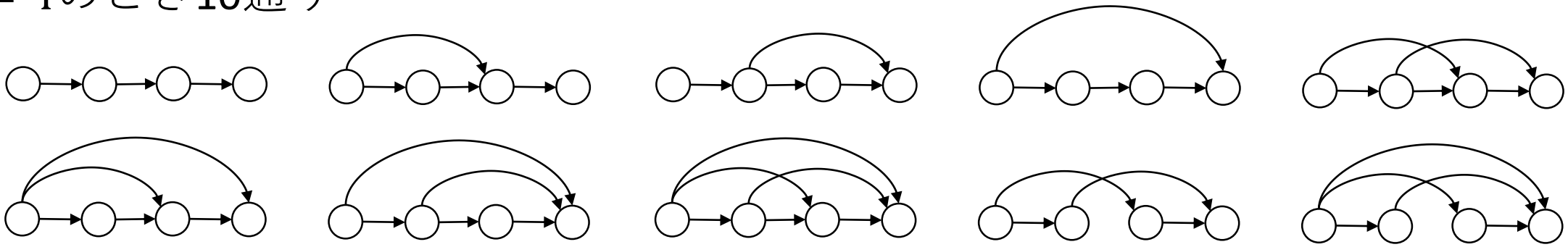
テストケース

$N = 2$のとき1通り



$N = 3$のとき2通り

$N = 4$のとき10通り



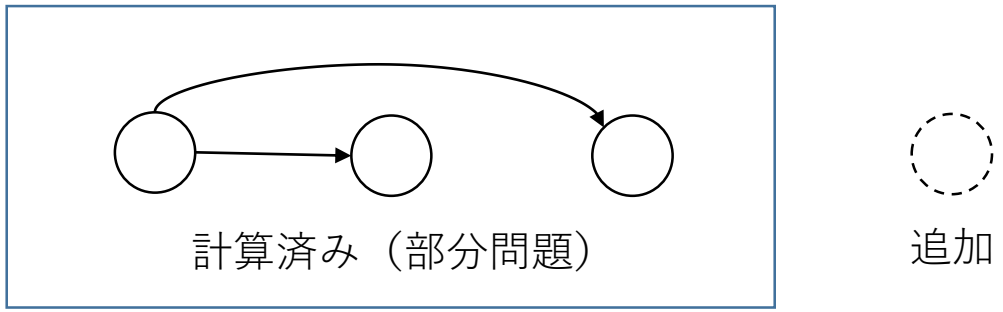
問題８　ゲームブック

解法（概要）

- $O(N^3)$の動的計画法.
- $dp[i][j]$:=頂点iまで計算し，出次数が0の頂点の個数がjである場合の数.
- $dp[N][1]$が答えとなる.

問題８　ゲームブック

解法（詳細）

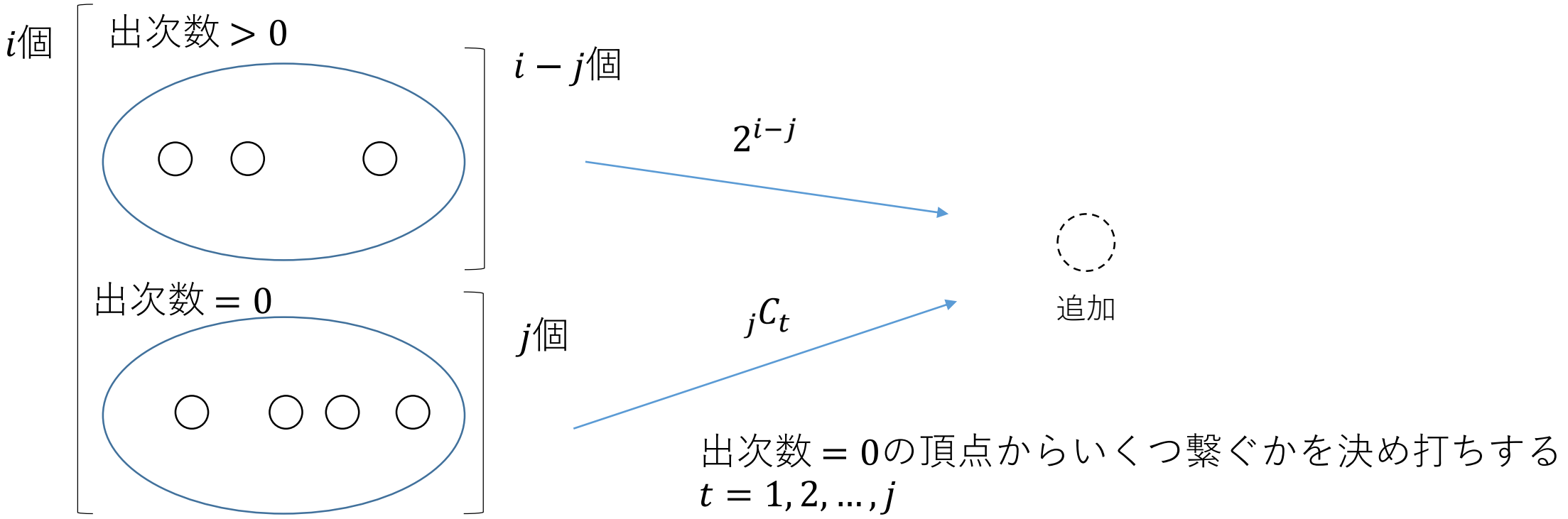


- 「１ページ目から任意のページにたどり着ける」を満たしつつ頂点を追加していく.
  - １本以上の辺が追加する頂点に入る.
- 頂点N以外の任意の頂点の出次数は１以上である.
  - 頂点Nのみが出次数が0である頂点となる.

問題８　ゲームブック

解法（状態遷移のイメージ）

- 出次数が0である頂点の個数をjとする（出次数が0でない頂点の個数は$i-j$）.



問題８　ゲームブック

解法（状態遷移のイメージ）

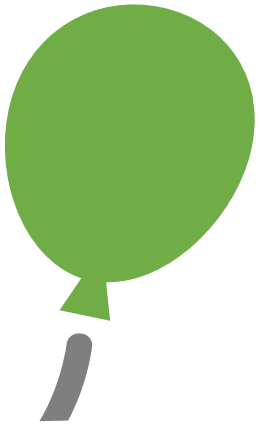
$dp[1][1] = 1$
$for\ i = 1\ to\ N-1$:
　　$for\ j = 1\ to\ i$:
　　/* 出次数が 0 の頂点とt個繋ぐ */
　　$for\ t = 1\ to\ j$:
　　$dp[i+1][j-t+1]\ +=\ {}_jC_t \times 2^{i-j} \times \mathrm{dp}[i][j]$
　　/* 出次数が 0 の頂点と一つも繋がない場合 */
　　$dp[i+1][j+1]\ +=\ (2^{i-j}-1) \times \mathrm{dp}[i][j]$

$print\ dp[N][1]$

問題９　縄張りの大きさ（１１点）

正答数:　6チーム

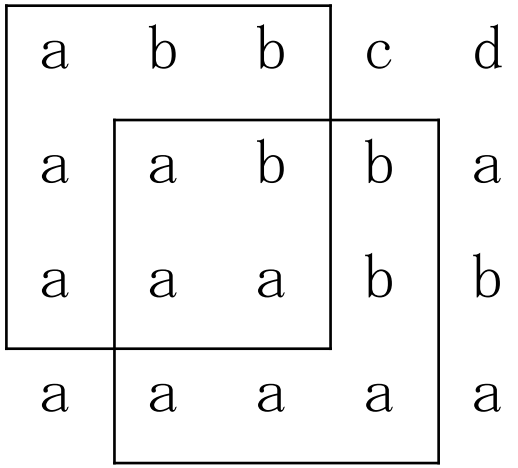
最初の正解:　大阪公立大学工業高等専門学校
P05 森林（グラフ）
1時間16分37秒 (14:31:37)



問題９　縄張りの大きさ

概要

- $H \times W$個の英小文字からなる２次元のテキストTと，整数kが与えられる．
- Tに含まれる，k回以上現れる正方形のパタンPのうち，最も大きいものの大きさを求めよ．そのようなパタンが存在しない場合は「-1」と報告する．
- $1 \leq H \leq 600,\ \ 1 \leq W \leq 600.$



- $k = 2$ のとき，答えは３

問題９　縄張りの大きさ

解法：基本方針

- 正方形の大きさを決め打ちして，その大きさのパタンのうち最も多く出現するものがk回以上現れるかを判定する.
- 最も大きいものが答え.

- 正方形の大きさをどのように探索するか？

  ⇒ 二分探索が適用可能.
- ある特定の大きさのパタンをどのように分類して数えるか？

  ⇒ 解法１：ローリングハッシュ（確率的な解法）

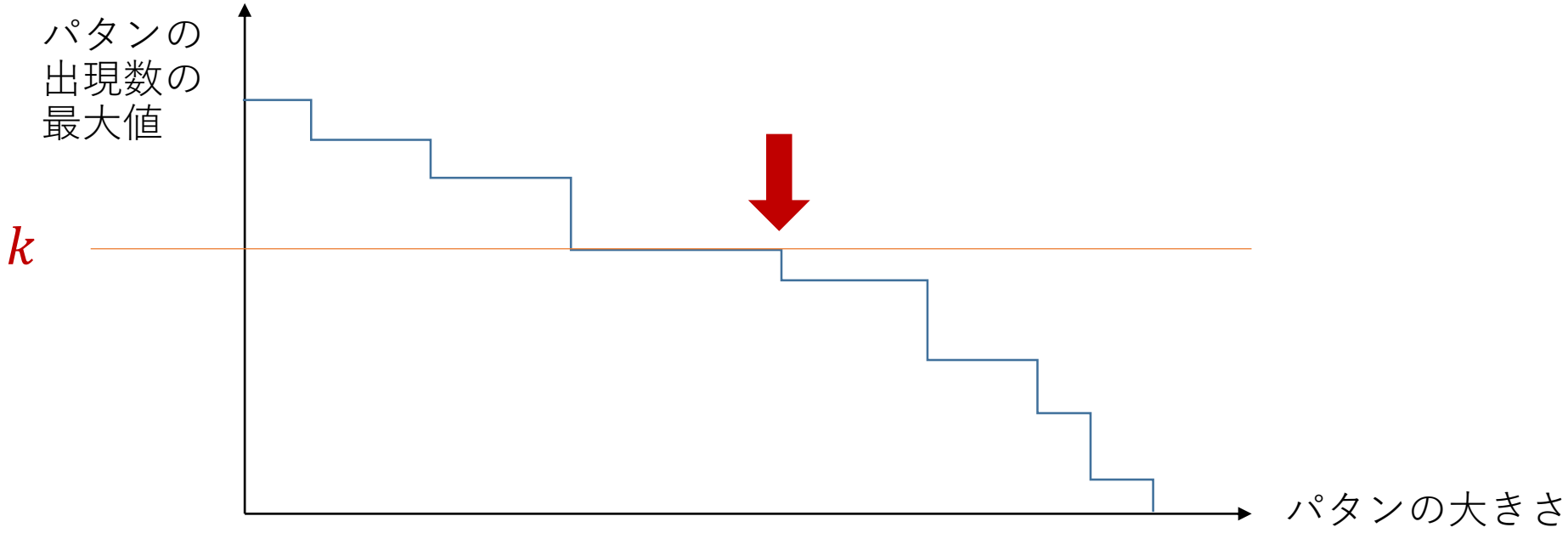
  　解法２：ダブリング（決定的な解法）
- 解法２をKMRアルゴリズムという.

問題９　縄張りの大きさ

解法：正方形の大きさの探索

- パタンが大きくなるほど，パタンの出現数の最大値は減っていく（増えない）
- パタンが小さくなるほど，パタンの出現数の最大値は増えていく（減らない）

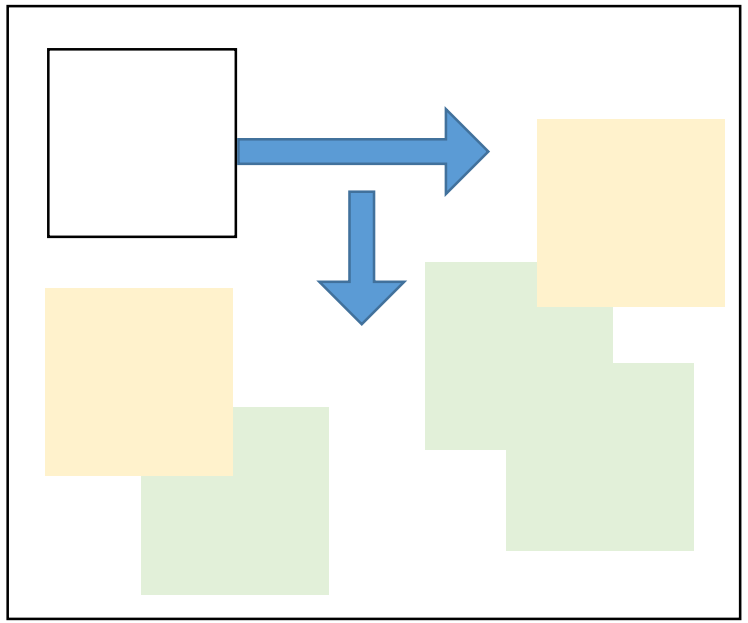
⇒ 二分探索が適用できる



問題９　縄張りの大きさ

解法：特定の大きさのパタンを数える（解法1）

- ローリングハッシュで全ての位置における正方形のパタンのハッシュ値を求める.
  - １つの正方形のハッシュ値を$O(1)$で計算.
  - 全てのハッシュ値を$O(HW)$で計算.
- ハッシュ値をmapで数える.
  - std::unordered_map　　$O(1)$が期待できる
  - std::map　　$O(\log HW)$



問題９　縄張りの大きさ

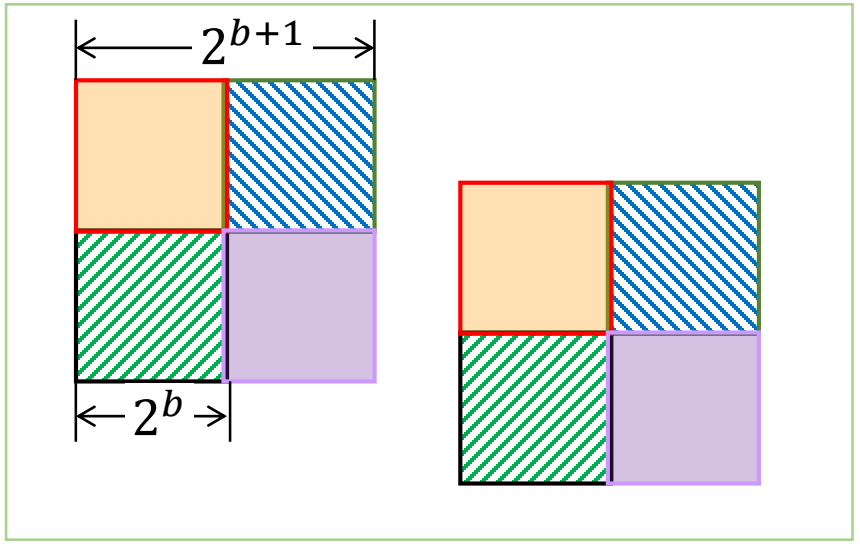
計算量：（解法１：二分探索＋ローリングハッシュ）

- 二分探索，ローリングハッシュ，unordered_mapを使って$O(HW \log HW)$

問題９　縄張りの大きさ

解法：特定の大きさのパタンを数える（解法２）

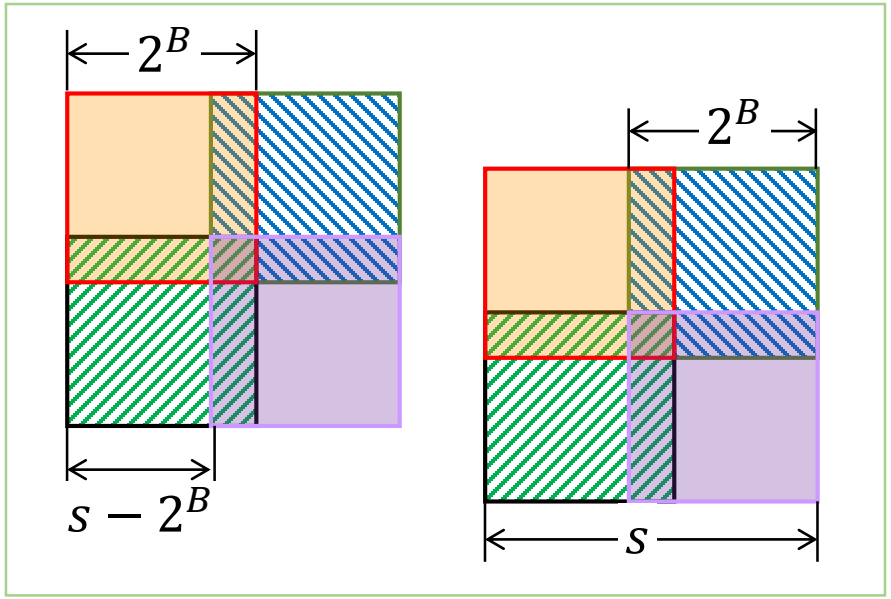
- ダブリングを使う.
- 前処理として，一辺が$2^b (b = 0,1, \ldots, N)$の正方形のパタンの出現情報を生成・保存する（Nは$2^N \leq \min(H, W) < 2^{N+1}$を満たす整数）.
- 一辺が2^bの正方形のパタンの出現情報から2^{b+1}の正方形パタンの出現情報を生成.
- 一辺が2^{b+1}のパタンは，2^bのパタンを４つまとめることで得られる（ダブリング）.
- 一辺が2^{b+1}の２つのパタンが等しい
  ＝対応する一辺が2^bのパタンがすべて等しい.



問題９　縄張りの大きさ

解法：特定の大きさのパタンを数える（解法２）

- 一辺がsの正方形のパターンの出現情報は，$2^B \leq s < 2^{B+1}$となる2^Bのパターンの出現情報から得られる．
- 一辺が2^Bの４つのパターンをまとめて，一辺がsのパターンを作る（ダブリング）．
- 下方向と右方向にそれぞれ$s - 2^B$だけずらしてパターンを重ねる（$(s - 2^B) + 2^B = s$）．



- 一辺がsのパターンが同じ ＝ 対応する大きさ2^Bの４つのパターンが同じ．

問題９　縄張りの大きさ

テキスト	0	1	2
0	d	a	b
1	a	b	c
2	b	c	a

解法：特定の大きさのパタンを数える（解法２）

- パタンをIDで管理。同じパタンは同じIDにする.
- 一辺sのパタンのIDは2^B（$2^B < s \leq 2^{B+1}$）の
  パタンのIDの列から生成（ダブリング）.

$s = 2^1$のパタンのIDは，2^0のパタンのIDの列から生成.

位置	(0,0)	(0,1)	(1,0)	(1,1)
IDの列	(4,1,1,2)	(1,2,2,3)	(1,2,2,3)	(2,3,3,3)

↓ IDの列の辞書式順序でソート

位置	(0,1)	(1,0)	(1,1)	(0,0)
IDの列	(1,2,2,3)	(1,2,2,3)	(2,3,3,3)	(4,1,1,2)
新ID	1	1	2	3

パターンの出現情報

	4	1	2
一辺2^0	1	2	3
	2	3	3

↓

	3	1	-
一辺2^1	1	2	-
	-	-	-

問題９　縄張りの大きさ

計算量：（解法２：二分探索＋ダブリング）

- 前処理で生成・保存するパタンの出現情報の個数は$O(\log HW)$
- IDは高々HW種類.
- M種類の要素N個の辞書式順序によるソートの計算量は$O(N+M)$

  ⇒ 新しいIDの生成の計算量は$O(HW)$.

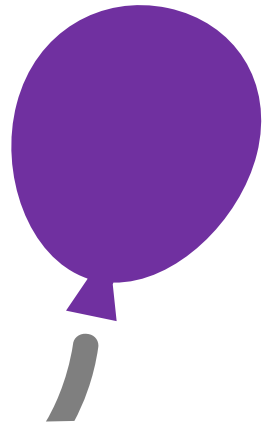
- ダブリングのための前処理で$O(HW \log HW)$.
- 二分探索，ダブリングで$O(HW \log HW)$
- 全体として$O(HW \log HW)$.

問題１０　数列の復元（１１点）

正答数:　14チーム

最初の正解:　大阪公立大学工業高等専門学校
P05 森林（グラフ）

44分1秒 (13:59:01)



問題１０　数列の復元

概要

- 長さNの数列$T = (T_1, T_2, \cdots, T_N)$と，正整数$X,Y$が与えられる．
- 全ての要素の値が0である長さNの数列$A = (A_1, A_2, \cdots, A_N)$に対して以下の操作$f_k$を複数回行うことができる．ただし$k = 1, 2, \cdots, N$である．

  f_k：数列Aのk番目の要素にXを加算し，それ以外の要素からYを減算する．
- 数列Aを数列Tと等しくするための各操作の回数の列を出力せよ．ただし，そのような各操作の回数の列が複数存在する場合は，操作の回数の総和が最も少ないものの列を出力せよ．
- $1 \leq N \leq 2 \times 10^5$，$1 \leq X, Y \leq 10^8$，$-10^9 \leq T_i \leq 10^9$．

問題１０　数列の復元

入出力例

入力例１	出力例１
5 4 1 -5 10 0 0 -5	0 3 1 1 0

入力例２	出力例２
5 4 2 -5 10 0 0 -5	-1

問題１０　数列の復元

解法

- 操作回数をC_iとする.
- $X, Y > 0$,　$C_i, T_i \geq 0$,　これらは全て整数.
- 次の式を満たすC_iを求める問題.

$$
\begin{aligned}
&XC_1 - YC_2 - YC_3 - \cdots - YC_N = T_1 \\
-&YC_1 + XC_2 - YC_3 - \cdots - YC_N = T_2 \\
&\vdots \\
-&YC_1 - YC_2 \cdots - YC_{i-1} + XC_i - YC_{i+1} \cdots - YC_N = T_i \\
&\vdots \\
-&YC_1 - YC_2 - YC_3 - \cdots - YC_{N+1} + XC_N = T_N
\end{aligned}
$$

- 操作iをi本目の式として表す.

問題１０　数列の復元

解法

- i本目の式

$$XC_i - Y(\sum C_k - C_i) = T_i$$

$$C_i = \frac{T_i - Y \times \sum C_k}{X + Y}$$ からC_iを求めることができる.

操作回数の和$SC = \sum C_k$ が求まればよい.

- $A = (0, 0, \cdots, 0)$である。
- 一回の操作ごとの$\sum A_i$の変化量を$\Delta T = X - Y(N-1)$とする.

問題１０　数列の復元

解法

- $\Delta T \neq 0$の場合

  操作回数の和$SC = \frac{\sum T_i}{\Delta T}$

  割り切れない場合や，値が負となる場合は不可能と判定.

- $\Delta T = 0$の場合

  すべての操作を１回ずつ行うと，すべての操作が打ち消される.

  最小回数の操作方法では，少なくとも１つの操作は一度も行われない.

  そのような要素は常に$-Y$されるため$\min(T_i)$に等しくなる.

  よって，操作回数の和$SC = -\frac{\min(T_i)}{Y}$

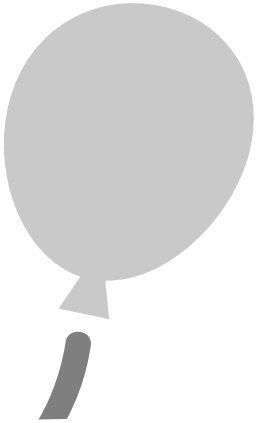
- $\Delta T \neq 0$の場合は解は一意に定まる．$\Delta T = 0$の場合は総和の最小性を仮定すると一意に定まる.

- $\frac{T_i - Y \times SC}{X+Y}$が割り切れない場合や負の値になる場合，$SC \neq \sum C_i$の場合は不可能と判定する.

問題１１　N数列（１２点）

正答数:　4チーム

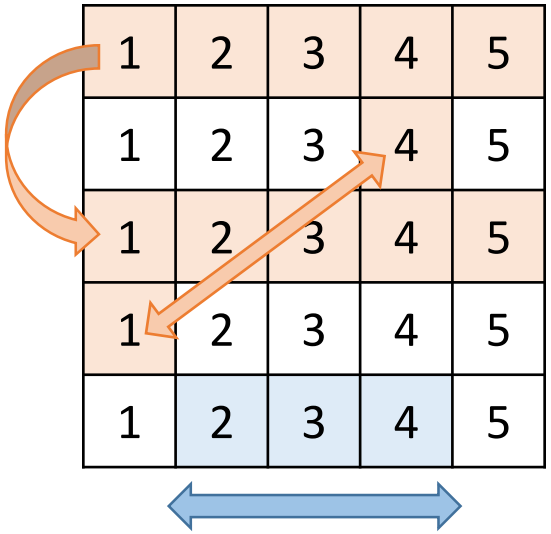
最初の正解:　大阪公立大学工業高等専門学校
P05 森林（グラフ）
25分49秒 (13:40:49)



問題１１　N数列

概要

- N個の要素からなるN本の数列$A_1, A_2, \cdots, A_N$があり，数列A_iのj番目の要素を$A_{i,j}$と表す．最初はすべての数列A_iについて$(A_{i,1}, A_{i,2}, \cdots, A_{i,N}) = (a_1, a_2, \cdots, a_N)$であるとする．
- 数列 $A_1, A_2, \cdots, A_N$ に対して，以下の３種類の操作を行う．
  1. 数列A_yを数列A_xで置き換える．
  2. 数列A_xのs番目の要素$A_{x,s}$の値と，数列A_yのt番目の要素$A_{y,t}$の値を入れ替える．
  3. 数列A_xに対して，s番目からt番目の要素の総和を出力する．
- $2 \leq N \leq 10^5$，$1 \leq Q \leq 10^5$，$1 \leq a_i \leq 10^9$.



問題１１　N数列

解法１

- 永続データ構造（セグメントツリー）.

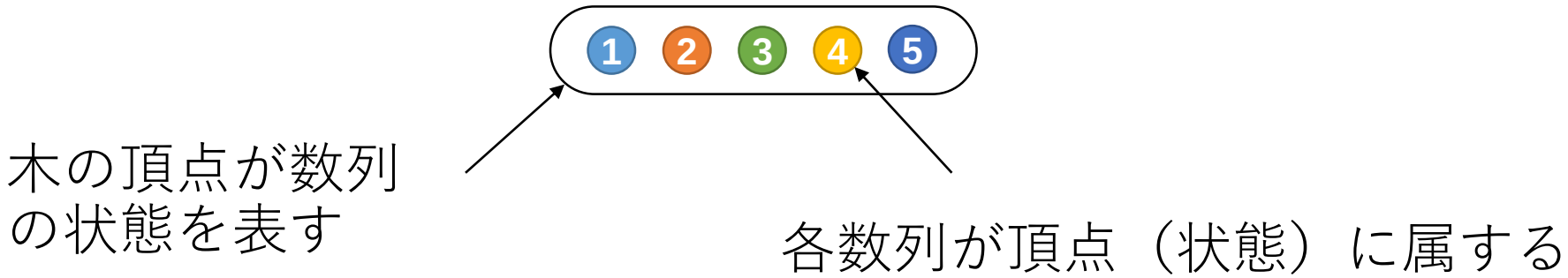
解法２（こちらを解説）

- 操作過程を表すツリーを構築.
- 交換クエリを更新クエリとして扱う.
- 操作ツリーに対してDFSを行いながらセグメント木を更新・ロールバック・検索する.
  - 要素更新と区間和が行えるセグメントツリーがあれば十分.

問題１１　N数列

操作過程を表すツリーを構築

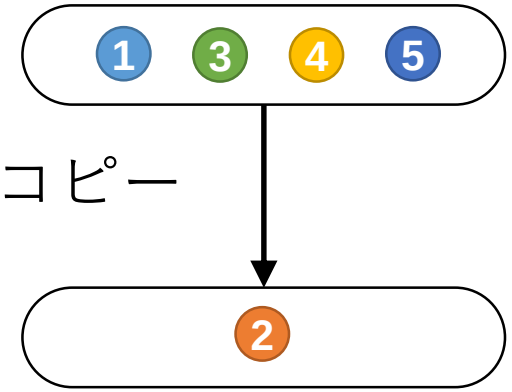
- 初期状態：頂点0が全ての数列を表している



問題１１　N数列

操作過程を表すツリーを構築

- クエリ１：数列のコピー



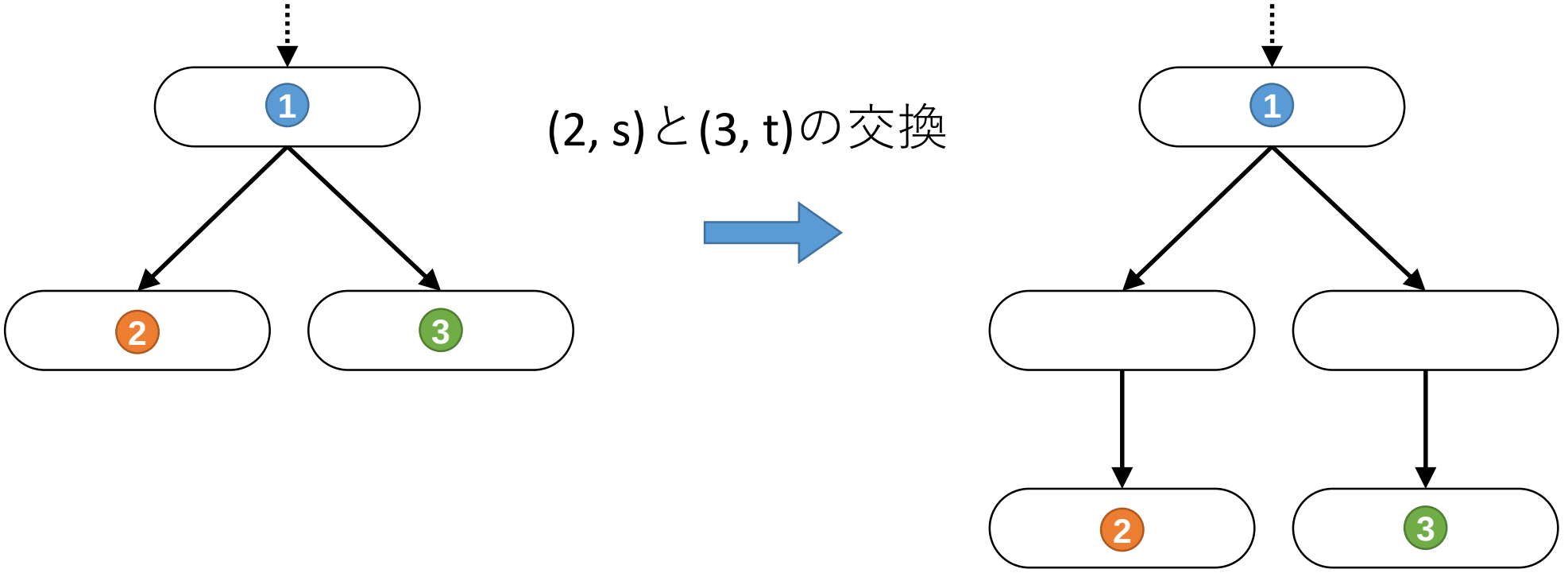
数列１を数列２にコピー

新しい頂点が数例２を表すようになる

問題１１　N数列

操作過程を表すツリーを構築

- クエリ２：要素の交換



問題１１　N数列

交換クエリから更新クエリへ

- 交換クエリ（$(r_1, c_1), (r_2, c_2)$の交換）が行われるとき，
  $(r_1, c_1) \to (r_2, c_2)$，$(r_2, c_2) \to (r_1, c_1)$のコピーを同時に行うと考えてよい．
- それぞれの$(r_1, c_1), (r_2, c_2)$の値を瞬時に求めることはできない．
  （すべての値を持つことはできないので）
- (r_1, c_1)の値を求めるには，r_1を表す頂点の祖先であってc_1番目が変更された頂点のうち最も近いものが求まればよい．

問題１１　N数列

交換クエリから更新クエリへ

1. **クエリを先読みして操作ツリーを構築しつつ，以下の値を計算**
   - $required[i][j]$:=頂点iにおいて何番目の情報が必要とされているかを表す.
   - $change[i]$:= 非負整数kのとき頂点iのk番目が交換クエリによって更新されることを表す．-1のとき更新されないことを表す.

問題１１　N数列

交換クエリから更新クエリへ

2. 操作ツリーに対してDFSを行い以下の値を計算

- $closest[i][j]$:= 頂点iの祖先であってj番目が変更された頂点のうち最も近いものの頂点番号.

$dfs1(u)$:
　$if\ change[u] \neq$ -1:　　// ノードuのchanged[u]列目が変更されている
　　$tmp \leftarrow previous[change[u]]$
　　$previous[change[u]] \leftarrow u$

　// ノードuにおいて，列cが必要とされている場合,
　// uにおける列cが更新された最も近いノード番号を記録
　$for\ c\ in\ required[u]$: $closest[u][c] \leftarrow previous[c]$
　$for\ v\ in\ adj[u]$: $dfs1(v)$

　$if\ change[u] \neq$ -1 : $previous[change[u]] \leftarrow tmp$

問題１１　N数列

交換クエリから更新クエリへ

3. クエリの順番に以下の値を計算
   - $value[i][j]$:=頂点iのj番目が更新される値を表す($value[0] \leftarrow a$)
   - $value[a][b] \leftarrow value\left[closest[c][d]\right][d]$

$for\ query\ in\ inputQueries$:

$\{com, a, b, c, d\} \leftarrow\ query$

$if\ \ com\ = 2$:

$value[a][b] \leftarrow\ value[closest[c][d]][d]$;

$queries[a] \leftarrow \{com, b, value[a][b]\}$

$else\ if\ com\ =\ 3$:

$queries[a] \leftarrow \{com, b, c, nrangeQ++\}$

問題１１　N数列

操作ツリーに対してDFSを行いながらセグメント木を更新・ロールバック・検索

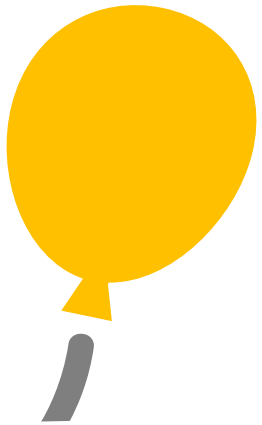
- ここまでで各交換クエリにおいて更新される値が分かっているので，交換クエリを更新クエリとして処理できるようになる.
- あとはDFSしながら区間和クエリと更新クエリを処理すればよい.

```
dfs2(u):
    for query in queries[u]:
        {com, x, y, i} ← query
        if com = 3:
            res[i] ← findSum(x, y)
        else if com = 2:
            tmp ← {x, x番目の要素の値}
            update(x, y)
    for v in adj[u]:
        dfs2(v)

    for e in tmp:
        update(e.first, e.second)
```

問題１２　三角形パズル（１４点）

正答数:　2チーム

最初の正解:　渋谷教育学園幕張高等学校
P03 貪欲なGreedy
1時間35分3秒 (14:50:03)



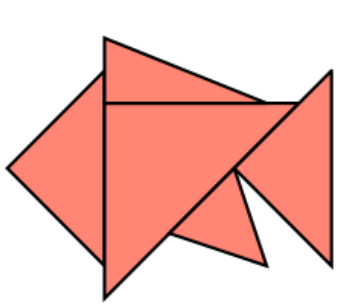
問題１２　三角形パズル

概要

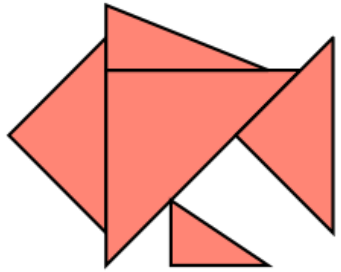
- N個の三角形が以下の条件で与えられる.
  - ✓互いに重なることはない.
  - ✓各三角形は少なくとも他の一つの三角形と辺の一部を共有して連結している.
  - ✓三角形の集合が形成する多角形の内部は，三角形によりすべて埋め尽くされている.
- これらの三角形で形成される多角形のすべての頂点の座標を求めよ.
- $2 \leq N \leq 10^5$，　$-10^9 \leq x_i, y_i \leq 10^9$.

問題１２　三角形パズル

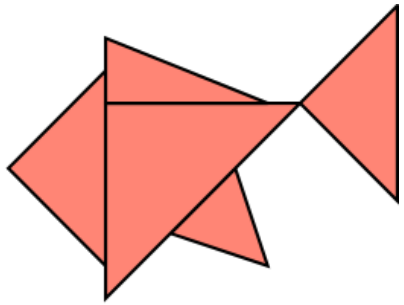
概要



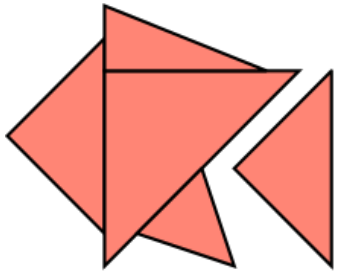
(a)



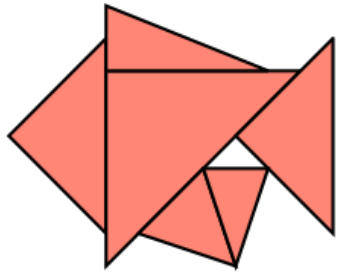
(b)



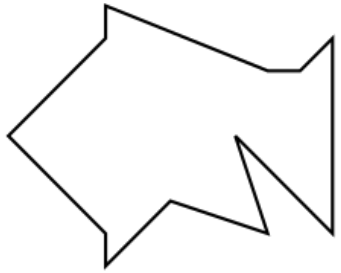
(c)



(d)



(e)

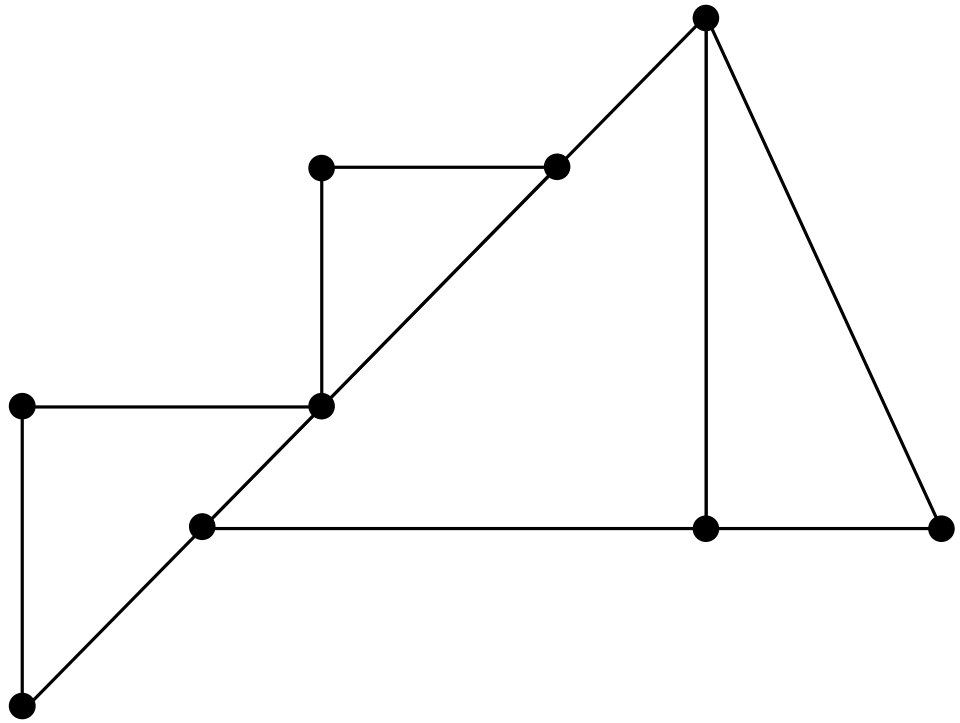


(f)

問題１２　三角形パズル

解法：概要

1. 直線に属する点のリストを作成する.
2. 多角形の辺となり得る辺を抽出する.
3. 多角形を生成する.

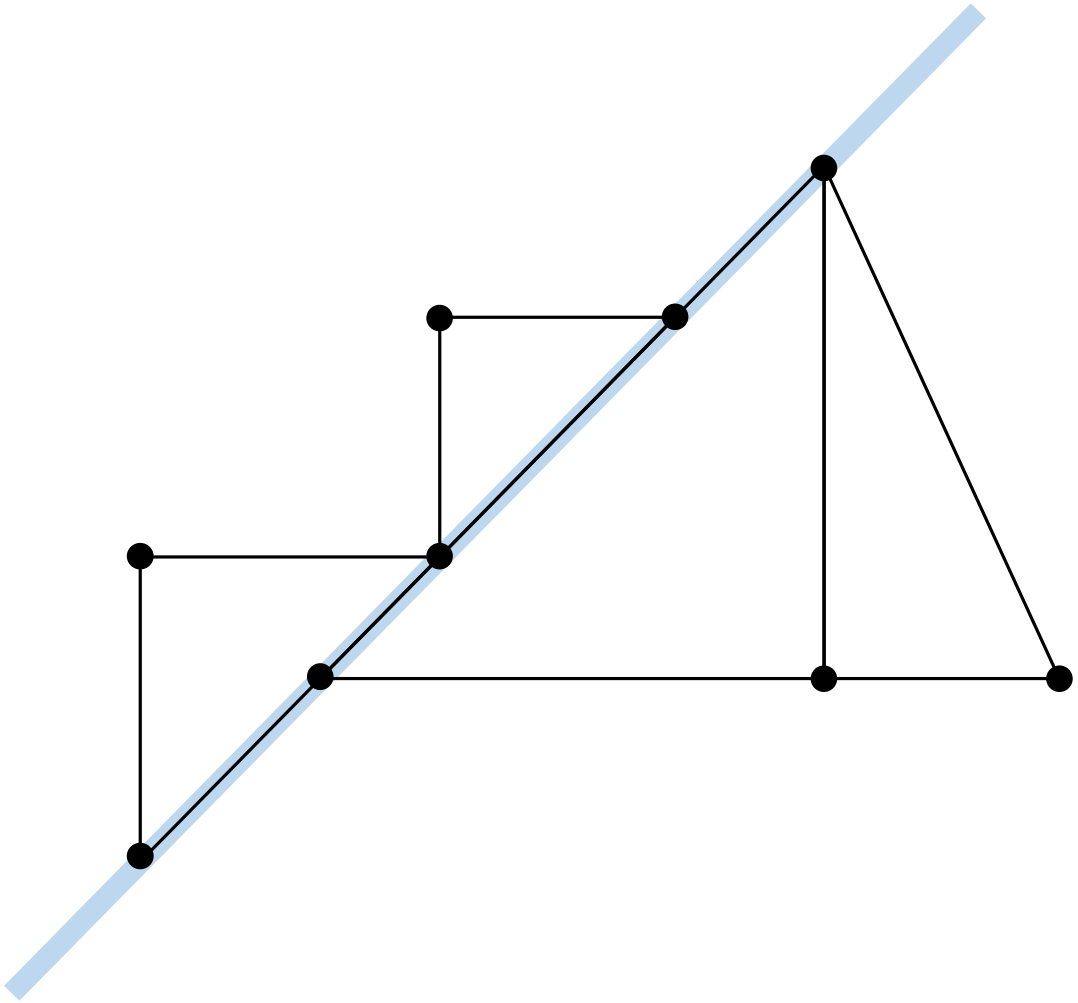


問題１２　三角形パズル

解法：直線に属する点のリストを作成する

- 直線をパラメタで表す.
  - $y = ax + b$　　（傾きと切片）
  - $ax + by + c = 0$　（一般形）
- 直線が軸に平行な場合に注意.
- 直線をパラメタによって一意に特定できればよい.

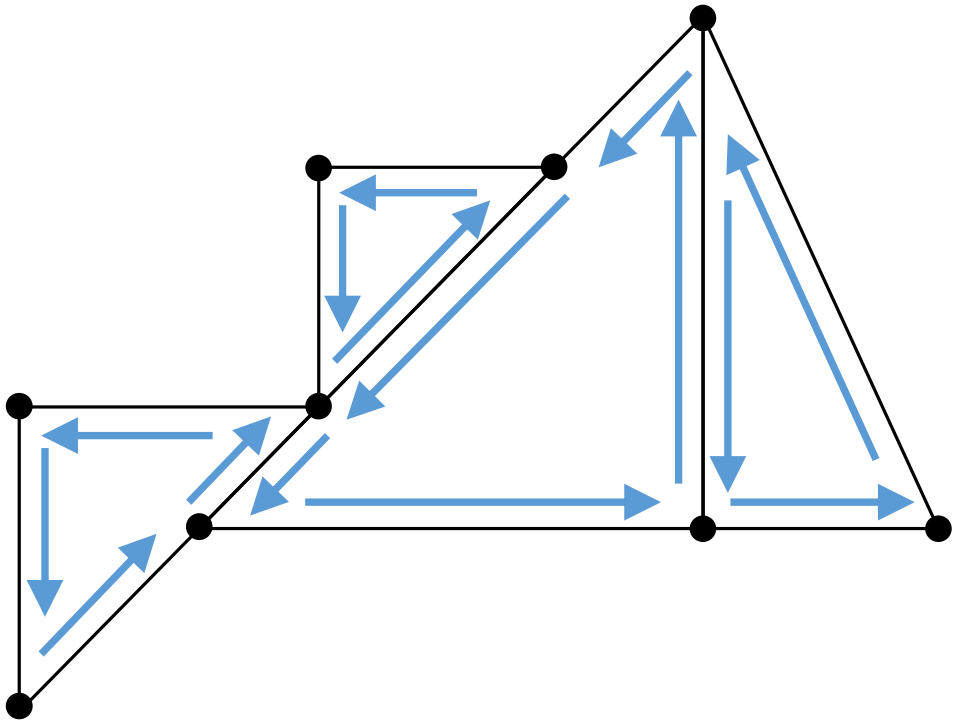
- ある点が属する直線をmapで管理できるようになる.
- 各直線上の点は整列しユニークをかけておく.
- 整列する処理に$O(N \log N)$.



問題１２　三角形パズル

解法：多角形の辺となり得る辺を抽出する

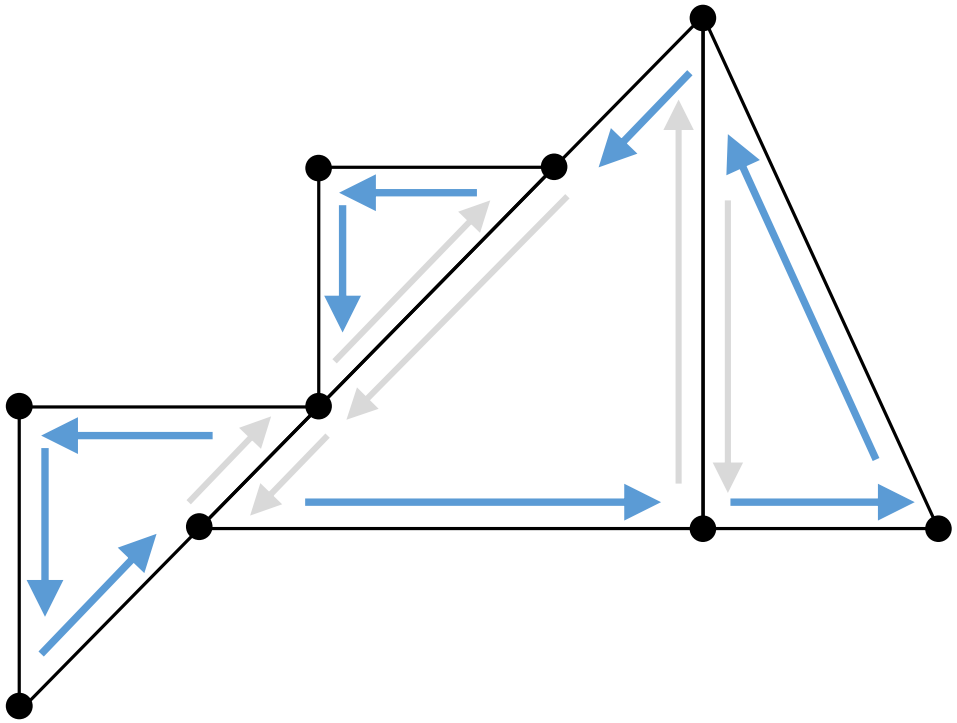
- 各三角形を反時計回りに辿り，各線分を訪問する．
  - 三角形の２頂点を始点・終点として該当する直線上の線分を訪問する．
  - 始点・終点は二分探索(lower_bound等）で取得できる．
- 全ての線分を辿る処理に$O(N)$
- 始点・終点を特定する処理に$O(N \log N)$



問題１２　三角形パズル

解法：多角形の辺となり得る辺を抽出する

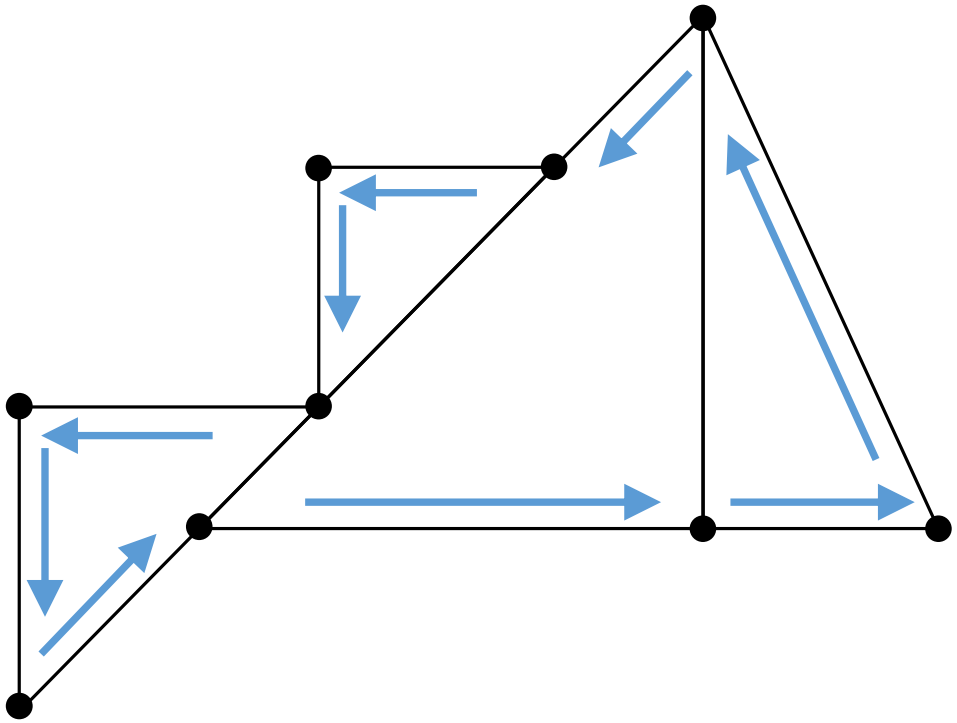
- 異なる訪問方向で相殺される線分を排除し，多角形の辺を抽出する
- 各点ごとに隣接する点を保持する
  - グラフを生成しておく



問題１２　三角形パズル

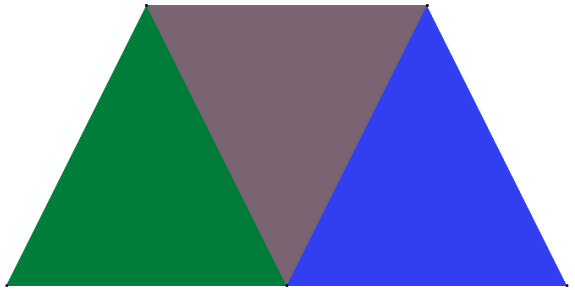
解法：多角形を生成する

- 視点と２番目の点を特定して，未訪問の隣接する点を順番に追加していく



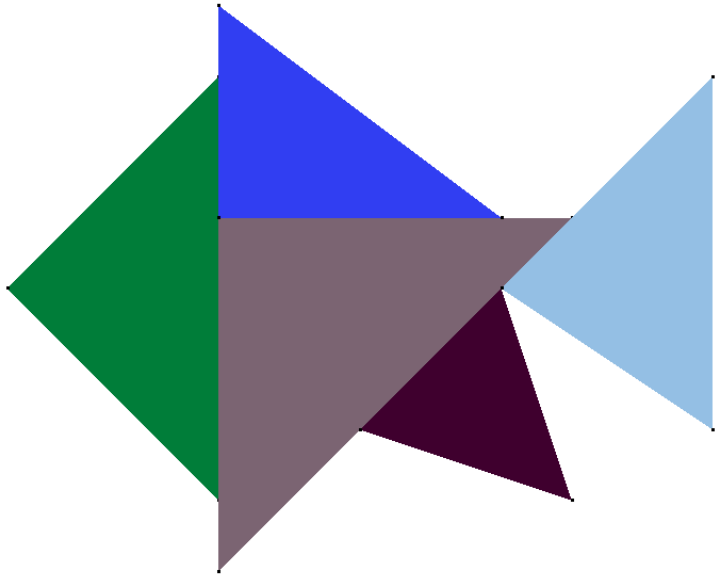
問題１２　三角形パズル

テストケース　00_sample_XX.in



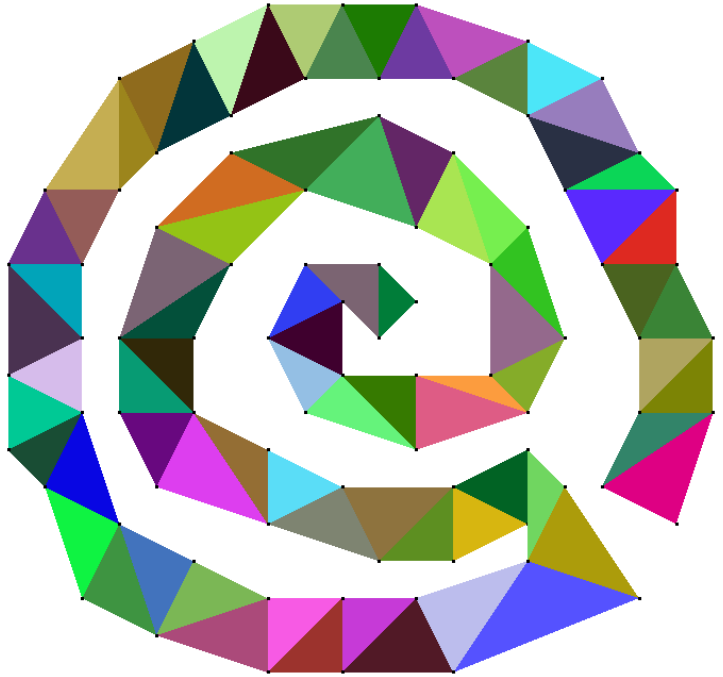
問題１２　三角形パズル

テストケース　01_small_XX.in



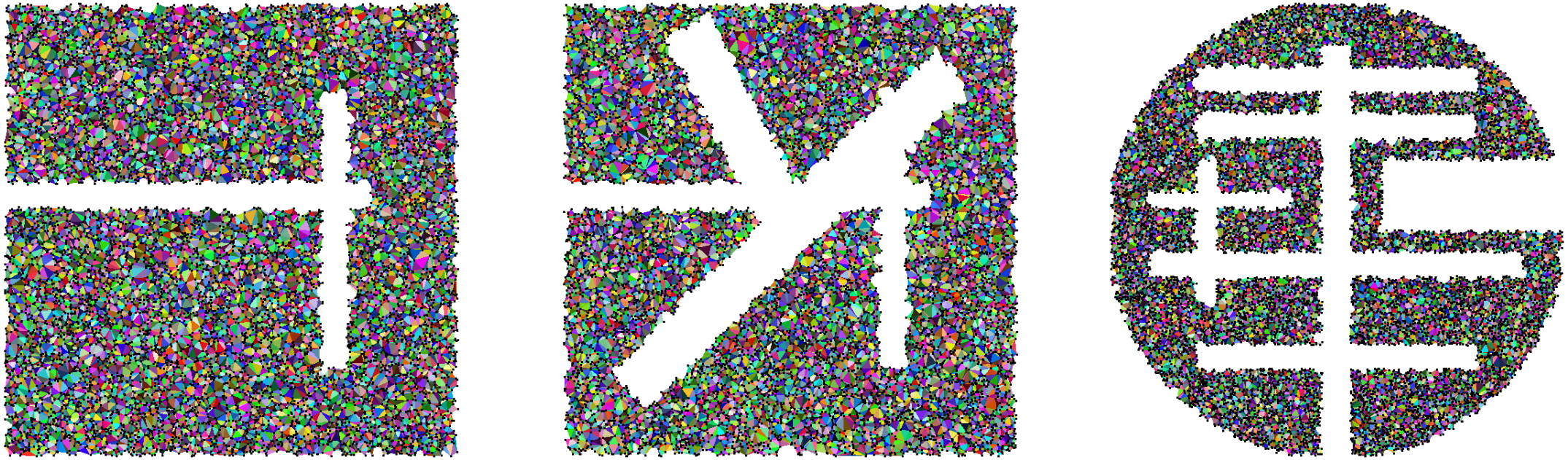
問題１２　三角形パズル

テストケース　03_general_XX.in



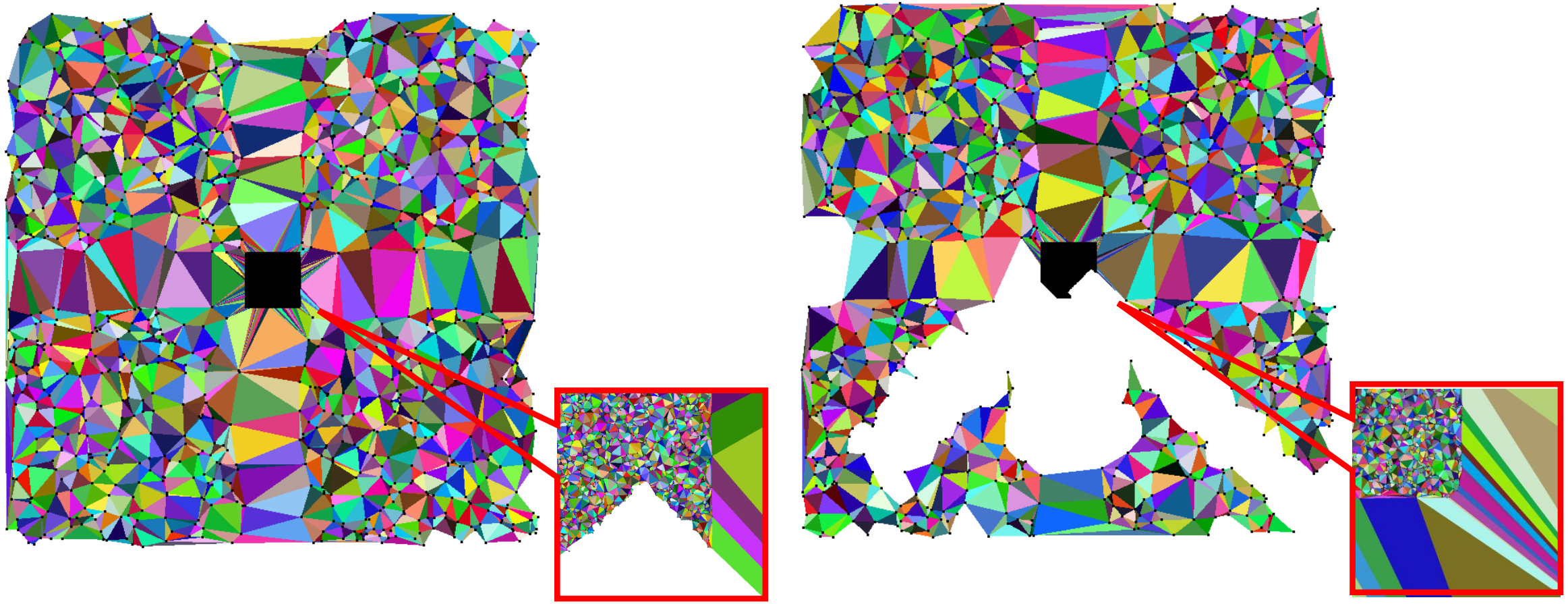
問題１２　三角形パズル

テストケース　05_large_XX.in



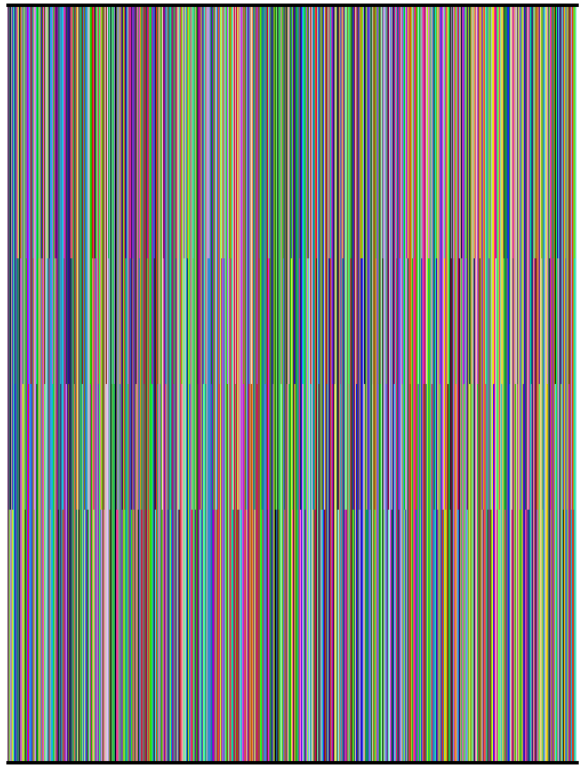
問題１２　三角形パズル

テストケース　06_corner_maximum_XX.in



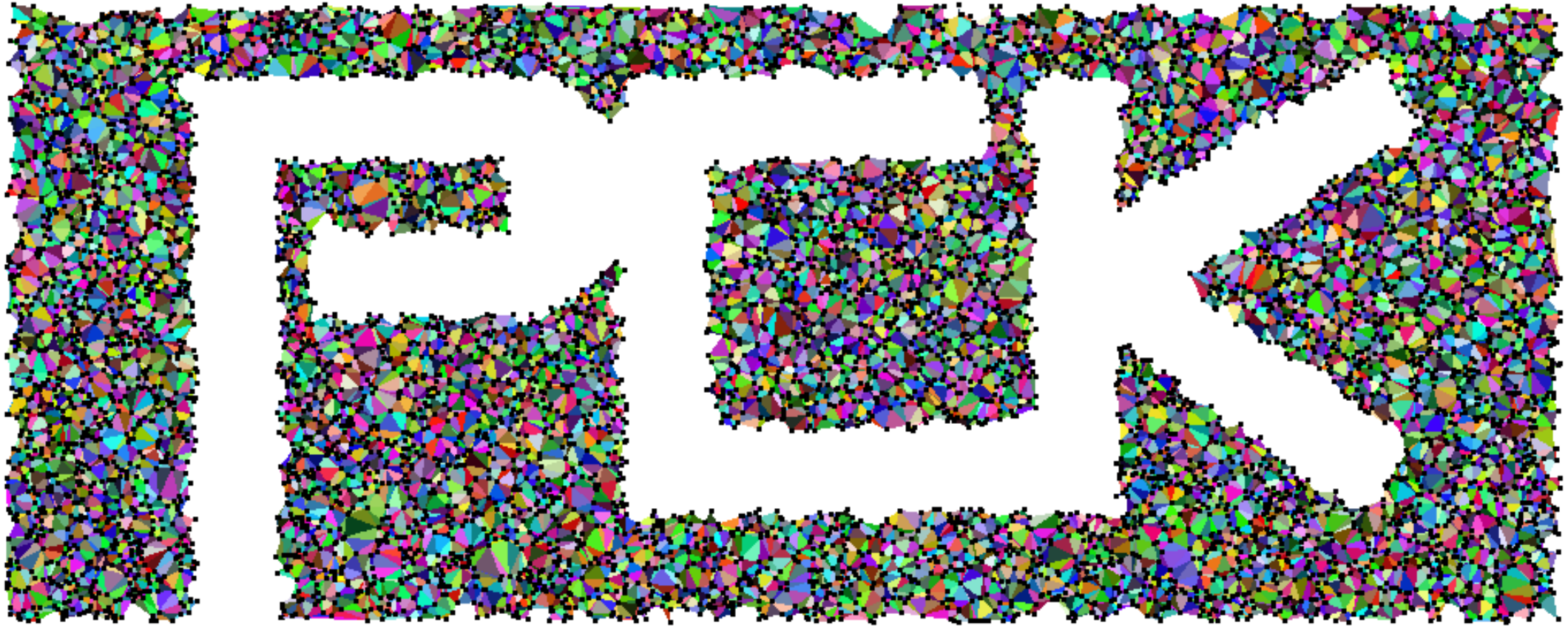
問題１２　三角形パズル

テストケース　06_corner_maximum_XX.in



問題１２　三角形パズル

テストケース　09_beautiful_XX.in



パソコン甲子園2022

ご視聴ありがとうございました。
また来年2023で!!