

解説・講評

writer / tester : tatyam, noshi91, potato167, simasima, ponjuice,
noya2, shobonvip, Rice_tawara459, Series_205, hotman78, Nzt3,
zer0star, otoshigo, kenken714, ebi_fly, tassei903, Astral__,
cho57020, x0214sh7
test-runner(Div. 1) : shiomusubi496, kaichou243, QgQ

目次

- 統計情報
- 運営想定難易度
- 解説(共通問題 7 問のみ)
- TTPC2024 の作り方 by noya2 (LT)



統計情報 (Div. 2)

番号	問題名	全体AC数	公式サイトFA
A	Don't Detect Cycle	1	
B	Self Checkout	39	のうねこにゃー
C	Segment Tree	1	CTRL
D	Odd Xor	69	NHK
E	ReTravel	2	
F	I prefer ISCT	176	のうねこにゃー
G	Coloring Tree	105	のうねこにゃー
H	TTPC Never Ends	122	Monochromatic Path
I	Near pair	1	
J	Grid Construction	4	VRChatはSteamで無料です
K	01 Abs Sum	1	
L	Long Sequence Inversion 2	7	Monochromatic Path
M	Colorful Stone Sorting	5	森
N	Adjacent Game	89	VRChatはSteamで無料です
O	Marunomi for All Prefixes	2	Monochromatic Path

運営想定難易度 (Div. 2)

運営想定難易度順 (Div. 2)

FH<GDN<ALBEOC<KMJ<I (by noya2)

FH<GD<BL<NKO<AECM<JI (by tassei903)

HF<GD<NM<ALO<CEK<B<JI (by Rice_tawara459)

実際の難易度順

FGH < NDB < LMJEO < ACIK



オンサイト順位発表 (Div. 2)



3位

のうねこにゃー



2位

VRChatはSteamで
無料です



1位

Monochromatic
Path

統計情報 (Div. 1)

※ AtCoder, Universal Cup を合わせたもの

番号	問題名	全体AC数	オンサイトFA
A	Don't Detect Cycle	21	japan406364961 (3:16)
B	Self Checkout	73	MCTUDSYTP (0:57)
C	Segment Tree	51	tritr (1:04)
D	A xor B plus C	3	
E	ReTravel	51	anmisted (0:23)
F	Origami Warp	9	japan406364961 (4:42)
G	Diverge and Converge	1	
H	Two Convex Set	2	
I	Near pair	7	
J	Grid Construction	17	anmisted (2:15)
K	Sum is One	10	
L	Long Sequence Inversion 2	80	SRM (0:36)
M	Cartesian Trees	26	japan406364961 (2:43)

運営想定難易度 (Div. 1)

運営想定難易度順 (Div. 1)

ALB < ECF < JKM < HGID (by noya2)

BL < AEC < FK < JM < GHID (by tassei903)

AL < CE < BFK < JIM < G < HD (by Rice_tawara459)

実際の難易度順

BEL < C < J < AK < I << DFGHM



オンサイト順位発表 (Div. 1)

- ・得点は部分点を削除したもの
- ・ペナルティは 0 分
- ・タイブレークは最終AC



3位 tritr

540点 LastAC : 3:45



2位 livia

730点 Last AC : 4:46



1位

japan406364961

7完 1107分

Last AC : 4:42

[参考情報] UCup 順位表



3位 fyyfbcbq

(apiad)

8 完 901 分



2位 HoMaMa0v0

(maroon, hos_lyric, maspy)

10 完 1423 分



1位 USA1

(tourist, ksun48, ecnerwala)

13 完 2041 分



解説(共通7問のみ)

A - Don't Detect Cycle

提案:Astral__

準備:Astral__

tester:Rice_tawara459

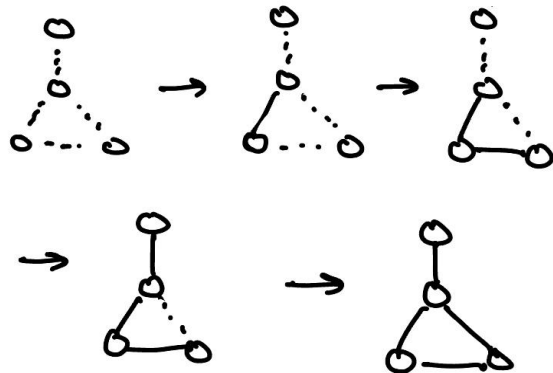
問題概要

Input

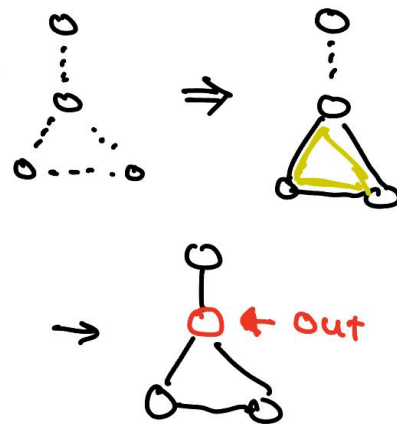


Output

• ok



• NG

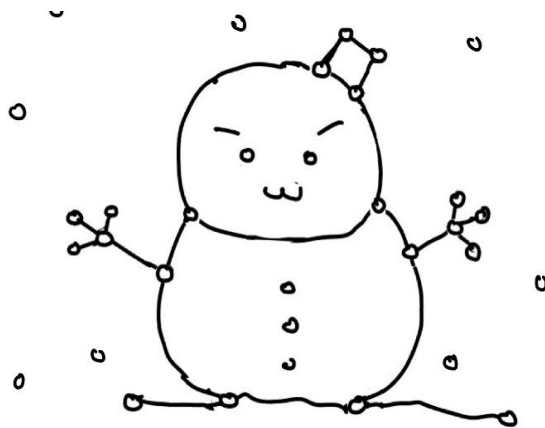


入力: 形が決まってるけど、まだ辺が無いグラフ

出力: 辺の追加順(順列)であって、それぞれの辺を追加する時に、端点がサイクルに含まれないもの (if exists)

制約

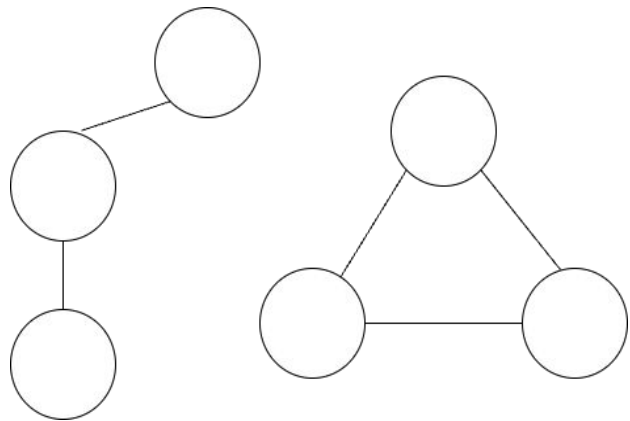
- $2 \leq N \leq 4000$ (部分点: $N \leq 100$)
- $1 \leq M \leq 4000$ (部分点: $M \leq 100$)
- 辺を全て追加したグラフは単純
- マルチテストケース



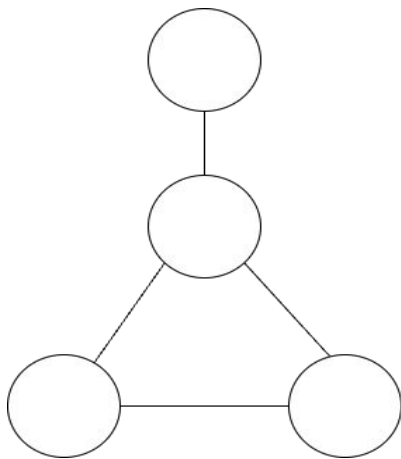
<- たぶん存在する

観察1

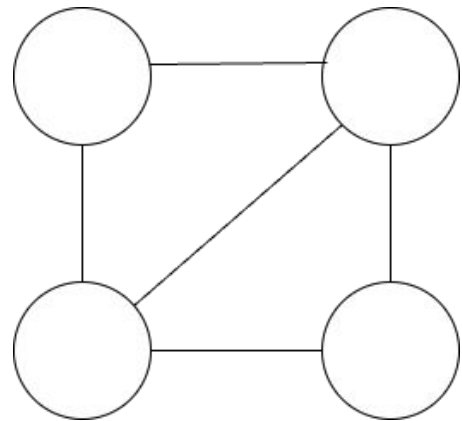
簡単に出来るグラフ



順番に依るが出来るグラフ



無理そうなグラフ



部分点

- ・最後の「辺の追加」に注目します
- ・条件を見たす追加順が存在するのなら当然、「たとえ最後に追加したとしても、その追加では条件を違反しない辺」が必ず存在する
- ・かつ、もしvalidな辺の追加順が存在するならば、その様な辺を最後に追加するような辺の追加順であってvalidな物も存在する
 - > 最後に追加すると決定してみる
- ・Q.最後に追加する辺を決定すると、問題はどうか変わる？

部分点

- ・A.最後に追加すると決定した辺はもう考慮しなくて良い（前の辺に影響を与えないので）
 - > 考慮すべき辺の本数が減る
 - > M が1小さくなっただけの、全く同じ問題に帰着される
 - > つまり、最後に追加しても条件を違反しない辺をある程度高速に取得できれば、それで解法になる
- ・とりあえず、辺を全探索して探します

部分点・まとめ

判定の手順:

- ・最後に追加する辺を固定し、それ以外の辺を追加したグラフを用意します
- ・そのグラフ上で、実際に最後の追加で条件が違反されないか見れば良いです
 - ・頂点 v を含むサイクルが存在する $\Leftrightarrow$ 頂点 v に直接繋がる辺であって、橋でない物が存在

例えばlowlinkを使うと計算量 $O(M^2(M + N))$ で動作します <- 部分点獲得

満点・考察1

部分点では、「最後に追加しても、その追加では制約が違反されない辺」を全探索で探していた

-> もしこれが高速に取得出来れば、それだけで解法が完成する

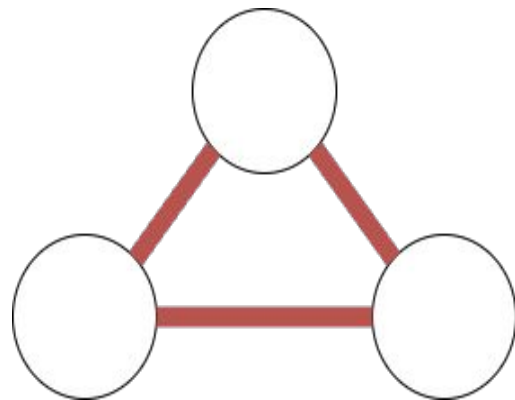
- ・ シミュレーションの高速化...たぶん難しいです

-> Q.最後に追加しても、その追加では制約が違反されない辺とはどのような辺？



満点・考察1

- ・A.辺であって、両端の頂点の次数が共に2以下の辺に注目
- ・この時、この辺は最後に追加しても制約違反しない
 - この辺を取り除いた時、両端の頂点の次数が ≤ 1 だから、どうしてもサイクルに含まれてない
- ・よって、「最後に追加しても、その追加では制約を違反しない辺」の一部が手に入った



満点・考察1

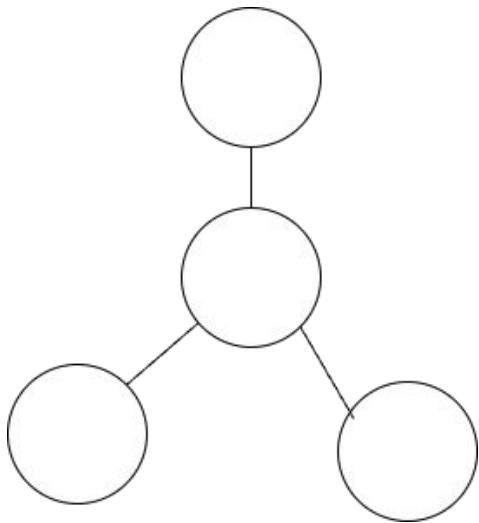
- ・ ⚠️ しかし、これは必要十分では無いので、まだ解けてない

-> Q. **この辺**を無視した上で、解くべきグラフとして残っているものは何？

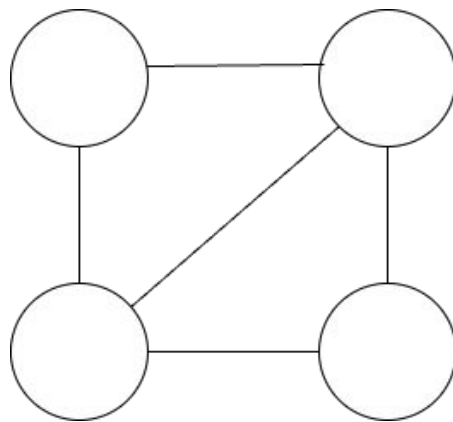
-> このあたりでもう1回観察を挟むと良いと思います

観察2: 残っているグラフ

未処理だが出来るグラフ



未処理で、無理そうなグラフ



満点・考察2

・ところで: “飛び出てる辺”は最初の方に追加した方が良い？

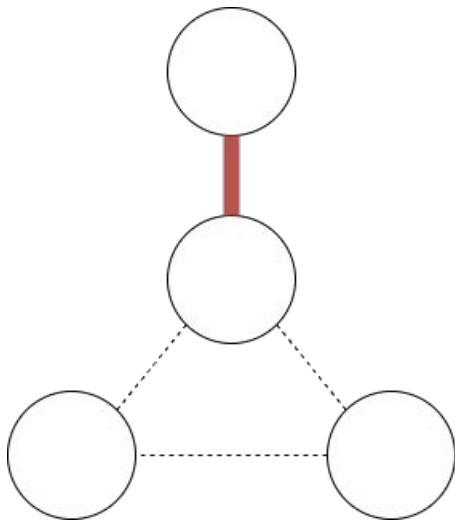
-> 良さそう

・なぜ？

-> 他の辺によって妨害される事はあっても、**この辺**が他の辺を妨害することはないから

・“他の辺を妨害することはない辺”とは？

<- サイクルを形成しない・決してサイクルに含まれない辺<-> **橋**



満点・考察2:

前ページのまとめ: 最終的なグラフの **橋** は、最初に追加するのが良い
もう1つ、しておくの良い観察があります:

- ・さっき決めた辺は、サイクルの形成に関わらない辺
- ・かつ、最初に追加する事にした

-> もう忘れちゃっても良くない?

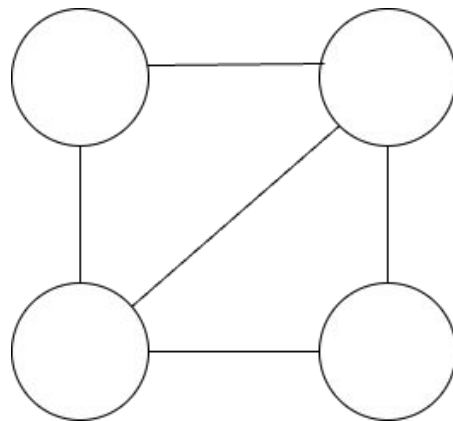
-> **橋** は、最初に追加すると決めたら、元々最初から追加のリストに存在しなかったと考えて良い

-> よって、これも最後に追加する辺と同様に、もう考慮しなくて良い

まとめ

考察 1・2 の辺の削除を繰り返しても残るグラフ:

- ・実はもう無理なグラフしか残っていない？
 - > Yes (証明はAtCoderの解説で)
- ・グラフの橋の列挙さえ(高速に)できれば終わり
 - > lowlinkというアルゴリズムがある
- ・実装で辺の順番をバグらせやすいかも(1敗)
 - > 気をつける



B - Self Checkout

提案: ebi_fly

準備: ebi_fly

tester: Nzt3

問題概要

人間は 1 または 2 の値を持っており、1 列に並んでいる。

1 の値を処理できるレジ **X** と 1 と 2 の値を処理できるレジ **Y** があります。

毎ターン、レジXは 1 の値を持つ人のうち最も先頭に近い人を処理し、その後

レジYは先頭の人を処理する。処理した値の和をそのターンの利益とする。

利益の列が入力された S と一致するような値の列はいくつありますか？



例: $S = (3, 2)$ のとき

S を生成する数列を A とします。

A が $(1, 2, 2), (2, 1, 2), (2, 1, 2), (2, 2, 1), (2, 2, 1), (2, 1, 1, 1), (2, 1, 1, 1), (1, 2, 1, 1), (1, 2, 1, 1)$ のとき $S = (3, 2)$ となります。

$A = (2, 1, 1, 1)$ のときは以下のように処理が進んでいきます。

$A: (2, 1, 1, 1) \longrightarrow (1, 1) \longrightarrow ()$
 $\begin{matrix} \uparrow & \uparrow \\ \text{Y} & \text{X} \end{matrix} \qquad \begin{matrix} \uparrow & \uparrow \\ \text{X} & \text{Y} \end{matrix}$

$S: () \longrightarrow (3) \longrightarrow (3, 2)$

問題の帰着

答えが非零となる数列 S は正規表現で表すと以下のようになります。

$$(3 * 2) * 3 * 2 * 1 ?$$

ここで $*$ は直前のものを 0 回以上繰り返すことを表します。

各段階に分けて解説していきます。

1. $3 *$
2. $3 * 2 * 1 ?$
3. $(3 * 2) 3 * 2 * 1 ?$
4. $(3 * 2) * 3 * 2 * 1 ?$

$S = 3^*$ のとき

S に含まれる 3 の個数を p とすると S を生成する 1, 2 からなる数列は以下を満たします。

- 1 が p 個、2 が p 個現れる
- 1 の prefix sum が 2 の prefix sum の個数 $+1$ 以下である

このような数列の個数は $(0, 0) \rightarrow (p, p) \wedge y = x + 1$ より上に行かないような最短経路の個数なので $\text{binom}(2p, p) - \text{binom}(2p, p + 2)$ となります。



$S = 3 * 2 * 1 ?$ のとき

2 は 1 を 2 つ取り除いて生成する場合と、2 を 1 つ取り除いて生成場合があります。ここで、2 を 1 つ取り除いて 2 を生成した後に 1 が取り除かれることはありません。



$S = 3 * 2 * 1 ?$ のとき

S の末尾に 1 が含まれる場合

$2 *$ は全て 1 を 2 つ取り除いて生成されます。

この場合は $3 *$ を生成する個数と等しいです。



$S = 3 * 2 * 1 ?$ のとき

S の末尾に 1 が含まれない場合

$3 \ 3 \ \dots \ 3 \ 2 \ 2 \ \dots \ 2$
p個 q個

2 のうち先頭 i 個を 1 を 2 つ取り除いて生成されるとします。

$i > 0$ のときの生成する数列の個数は $3 *$ を生成する数列の個数と等しいです。

$S = 3 * 2 * 1 ?$ のとき

S の末尾に 1 が含まれない場合

$3 \ 3 \ \dots \ 3 \ 2 \ 2 \ \dots \ 2$
 └───┘ └───┘
 p個 q個

2 のうち先頭 i 個を 1 を 2 つ取り除いて生成されるとします。

$i = 0$ のときは $(0, 0) \rightarrow (p + q, p) \curvearrowright y = x + 1$ より上に行かないような最短路の個数

となり $\text{binom}(2p+q, p) - \text{binom}(2p + q, p + q + 2)$ となります。

$S = (3 * 2) 3 * 2 * 1 ?$ のとき

$S = 3 * 2 * 1 ?$ で表現できない場合を考えます。

$(3 * 2)$ の部分を生成する部分列は $3 *$ を生成する数列の末尾に 1 を 2 つ追加したものです。

末尾で 1 が 2 つ連続しているため $S = (3 * 2) 3 * 2 * 1 ?$ を生成する数列 A で

$(3 * 2)$ と $3 * 2 * 1 ?$ を生成する部分は連続部分列となり重なりません

よって独立に数え上げて積を取ればよいです



$S = (3 * 2) * 3 * 2 * 1 ?$ のとき

S を生成する数列の中で、各 $(3 * 2)$ と $3 * 2 * 1 ?$ を生成する部分列はそれぞれ連続部分列であり重なってないです。

よってそれぞれ数え上げて積を取ればよいです

これまで見たように、各部分は二項係数を用いて計算できます。

よって全体で $O(N)$ でこの問題が解けました。



C - Segment Tree

提案: shobonvip

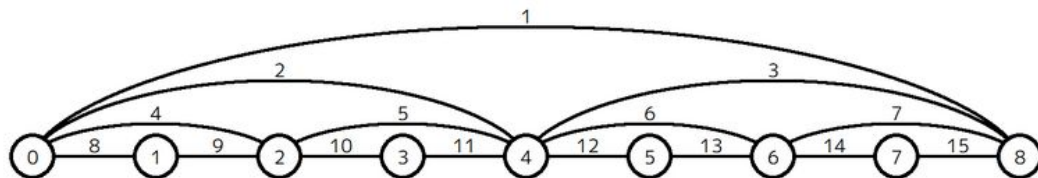
準備: shobonvip

tester: ponjuice, Rice_tawara459

問題概要

セグメント木の区間の両端に辺がひかれたような $2^N + 1$ 頂点 $2^{N+1} - 1$ 辺無向グラフが与えられる。次の Q 個のクエリに答える:

- 2 点間の最短路の長さを出力
- 辺の長さを変更



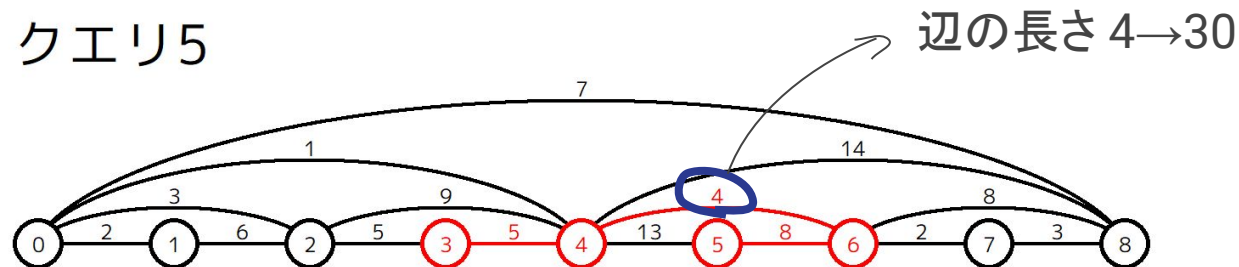
$N \leq 18, Q \leq 2 \times 10^5$, 各辺の長さは 1 以上 10^7 以下

1								
2				3				
4		5		6		7		
8	9	10	11	12	13	14	15	
0	1	2	3	4	5	6	7	8

サンプルを見る

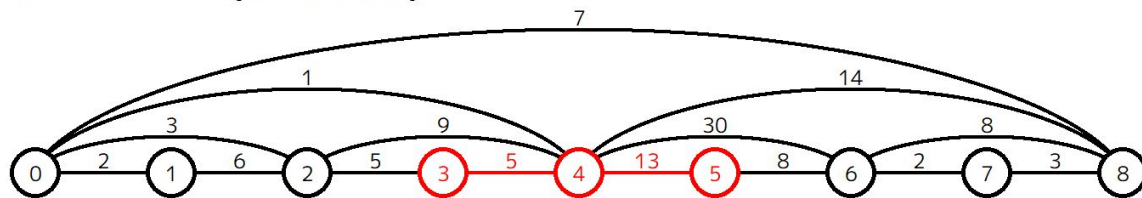
答え:17

クエリ5



答え:18

クエリ7 (更新後)



考察

- 一般のグラフでは難しそう
- 今回、グラフの見た目がきれいなのでこれを利用できないか？

図のように、上の辺を通るより下の辺を連続して通る方が安いなら、上の辺を下の辺の重みの合計としてみる



辺の重みを調整

→下の辺を連続して使う代わりに
上の辺を使えばよい

式で表すと

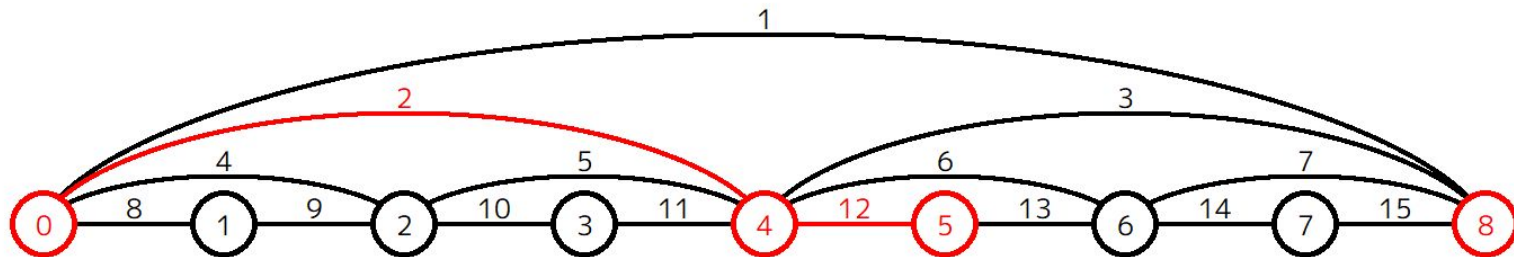
調整された重み W_i はこのように表せる

$$W_i = \begin{cases} C_i & (2^N \leq i < 2^{N+1}) \\ \min(C_i, W_{2i} + W_{2i+1}) & (1 \leq i < 2^N) \end{cases}$$

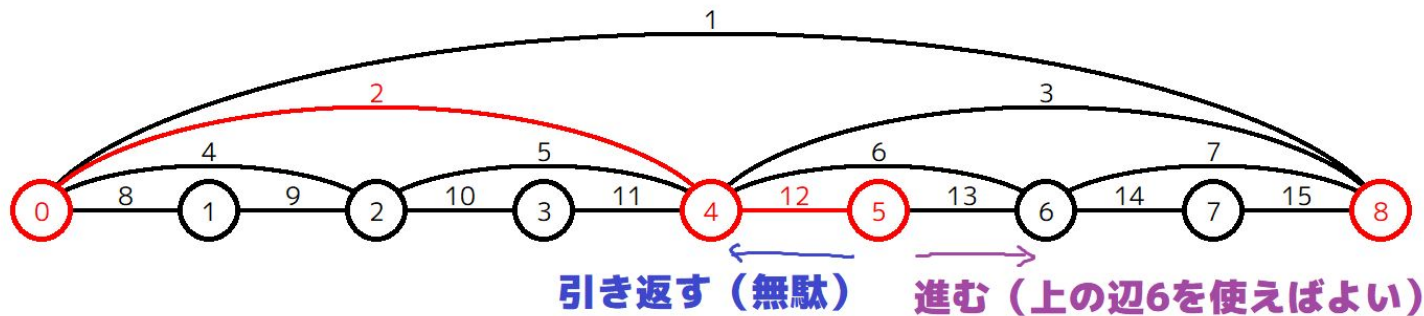
このとき、連続する下の辺を使う代わりに上の辺を使えばよかった

→「最適解のうち、同じタイプの辺を連続で使わないものが必ずある」

下の段に迷ってしまう場合



(重み調整後)たとえば0 から8 に向かおうとして、 $0 \rightarrow 4 \rightarrow 5$ と下の段に迷い込んだとする → 引き返しても進んでも駄目がいえる！

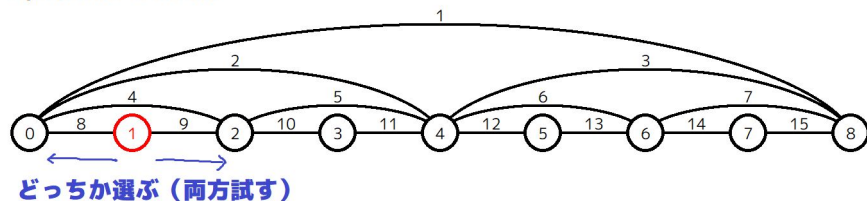


もう少し詳しく.....

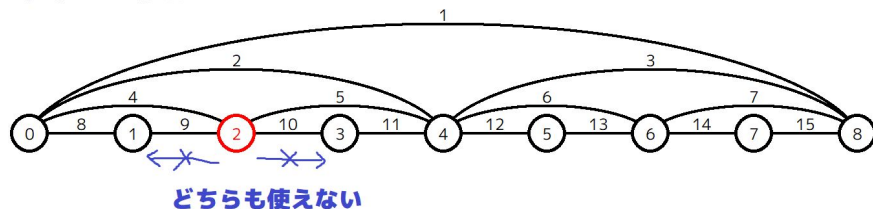
番号が奇数の頂点を通る可能性があるのは、最初と最後だけ

- 奇数スタートの場合、最初が一番下の辺を使う必要がある
 - 最初 2 通りの進み方が考えられるので、2パターン調べる
 - 以降は一番下の辺を使ってはいけない
- 偶数スタートの場合、一番下の辺は使ってはいけない

最初が奇数



最初が偶数



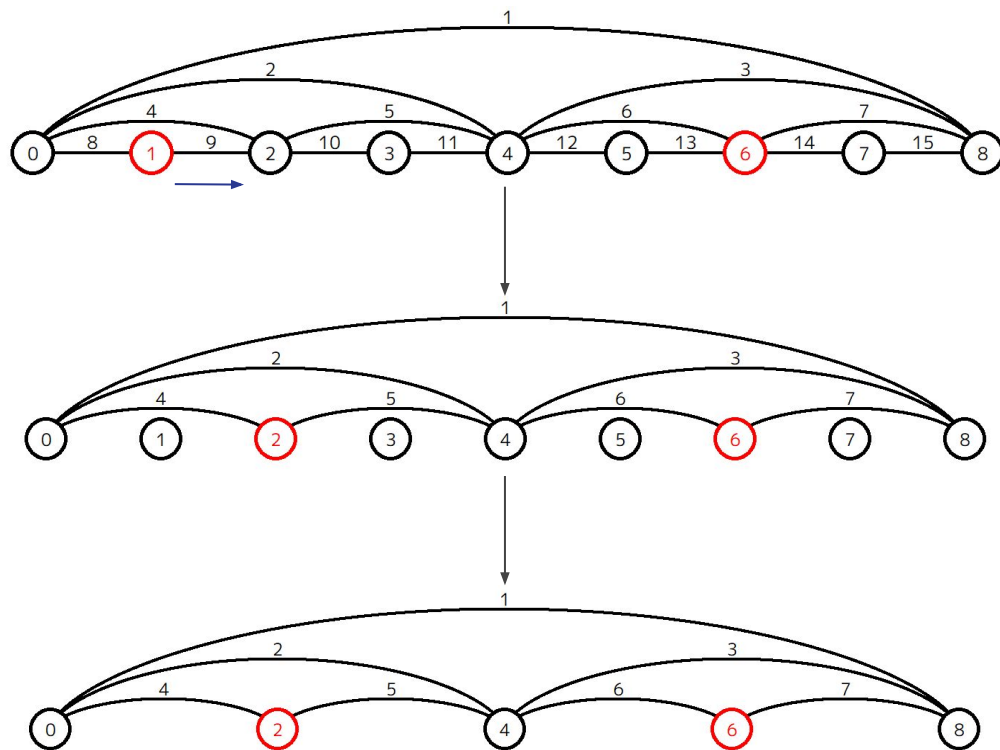
1手先

同時に、終点も一番下の辺を考える

その1手先

- もう一番下の辺は使えない
- 頂点も偶数しか考えなくて良い

→ より小さいグラフに帰着される！



これを繰り返すが...

奇数を引くたびに 2 通りに分かれてしまい、とんでもない時間がかかるのでは？

→ メモ化再帰を使うと、実は辺の段ごとに**定数個(高々 $2 \times 2 = 4$ 個)のパターン**しかないことが証明できる！

→ 「2 s t」のクエリは $O(N)$ 時間で出来る

「1 j x」のクエリも、更新ごとに W_i は高々 $O(N)$ 個しか変わらない！

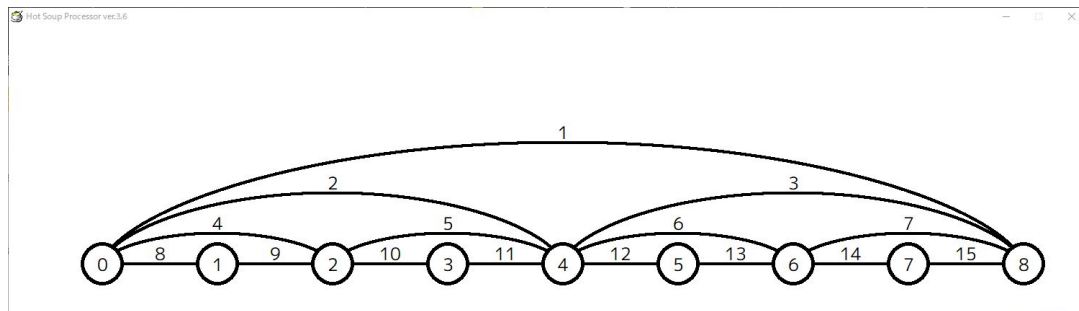
- W_i はデータ構造の **Segment Tree** のように更新できる
- 変える値は W_i ではなく C_i であることに注意
- このクエリも $O(N)$ 時間

空間計算量 $O(2^N)$ 、時間計算量 $O(2^N + QN)$

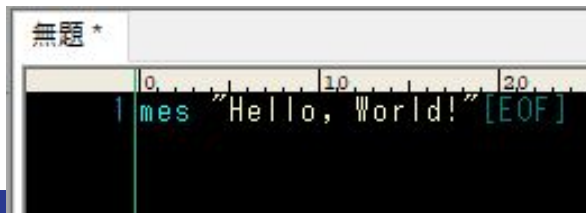


裏話

この図は Hot Soup Processor という思い出の言語で作った
(AtCoder では使えない)



GUI が超簡単に作れるので便利かも



Div.1 D - A xor B plus C

提案:noya2

準備:noya2

tester:potato167

証明査読:tatyam, noshi91

Div.2 D - Odd Xor

提案:Nzt3

準備:Nzt3

tester:tassei903

E - ReTravel

提案:tassei903

準備:tassei903

tester:ponjuice

問題概要

平面上の N 点を与えられる。

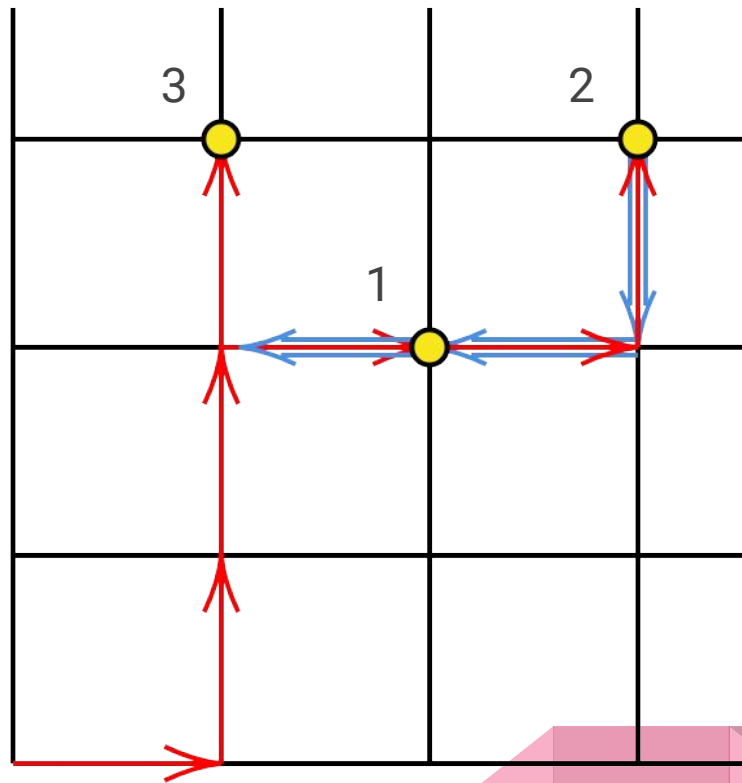
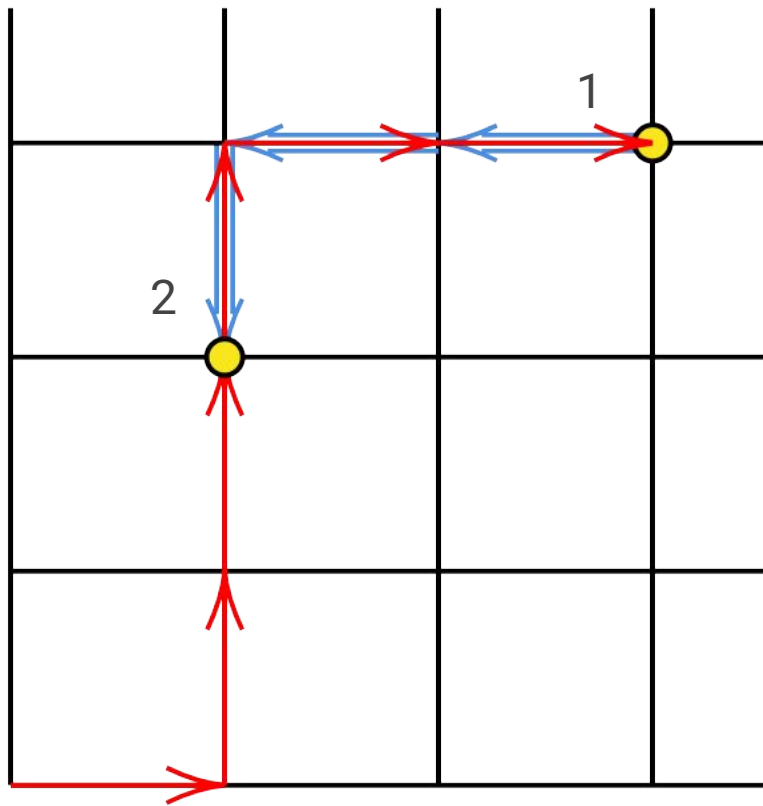
→ or $\uparrow$ に進むか(コスト1) 操作の逆戻り(コスト0) ができるとき

N 点を順番に訪れる最小コストをもとめる。

制約

- $1 \leq N \leq 5000 \leq X_i, Y_i \leq 10^9$





区間DP

$$MX[l, r] = \min(X_l, X_{l+1}, \dots, X_r)$$

$$MY[l, r] = \min(Y_l, Y_{l+1}, \dots, Y_r)$$

$dp(l, r) = (MX[l, r], MY[l, r])$ からスタートして $l, \dots, r$ を訪れる最小コスト

$$dp(l, r) = \min(dp(l, k) + dp(k+1, r) + MX[l, k] + MX[k+1, r] - 2MX[l, r] + MY[l, k] + MY[k+1, r] - 2MY[l, r])$$

($MX[l, r], MY[l, r]$)に戻ってくるタイミングを全探索

Div.1 F - Origami Warp

提案: Rice_tawara459

準備: Rice_tawara459

tester: noya2

Div.2 F - I prefer ISCT

提案:0214sh7

準備:0214sh7

tester:otoshigo

Div.1 G - Diverge and Converge

提案: noya2

解法提案: noshi91

準備: noya2

tester: noshi91

Div.2 G - Coloring Tree

提案:tassei903

誤読:tatyam

準備:kenken714

tester:cho57020

Div.1 H - Two Convex Sets

提案: noya2

解法提案: tatyam

準備: noya2

tester: tatyam

Div.2 H - TTPC never ends

提案:0214sh7

準備:noya2

tester:tassei903

I - Near Pair

提案:simasima

準備:simasima

tester:Rice_tawara459, tatyam

問題概要

$(1, 2, \dots, N)$ の順列の端の要素を当てたい

クエリを Q 回以下送って当てよう

クエリ: ある部分集合に差が K 以下の2数の組が何組あるか？

制約

- $N = 20000$
- $1 \leq K \leq 10$ (部分点: $K=10$)
- $Q = 30(K+1)$
- 順列は固定(adaptiveでない)

クエリが少ないよ

一つのクエリで得られる情報は高々NK

$(NK)^Q < N!$ より数列の全ての要素を当てるのは不可能

->端だけを当てる必要がある

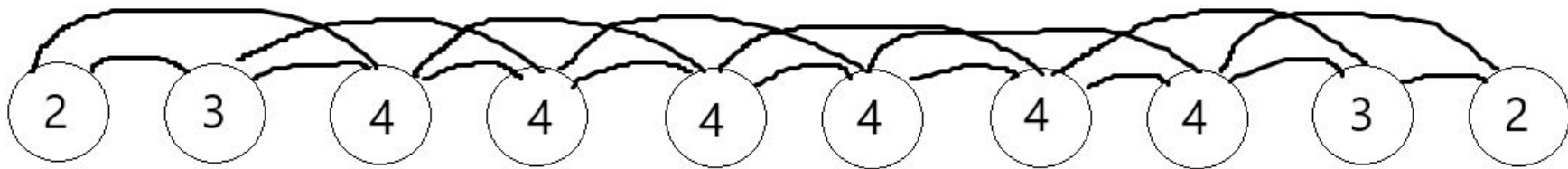


グラフにする

カウントされるペアに辺を張ったグラフを考える

この時、クエリは誘導グラフの辺の数を聞いていると解釈できる

K=2の場合

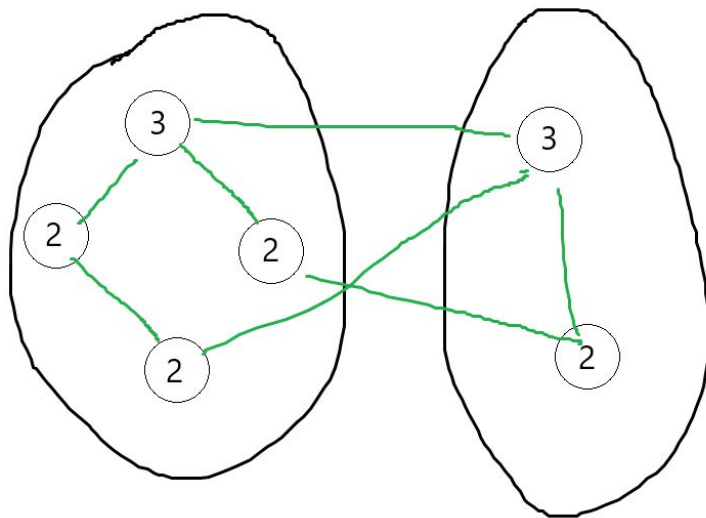


端の次数だけが少ない！

解法

ある部分集合とその補集合についてクエリを投げる

この時、二つの誘導グラフの辺の数の差は次数の差の二倍になっている

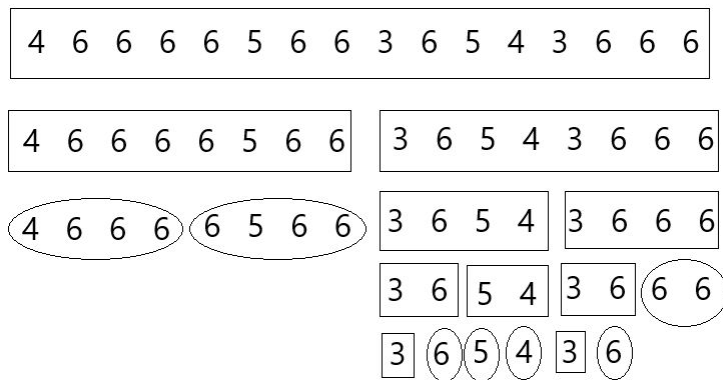


解法

さっきの方法を使って二分探索をする

この時、 $(2K-1)$ の和が K 未満であるような集合は探索を打ち切ってよい

$15 \times (K+1) \times 2$ で確実に間に合う



J - Grid Construction

提案:noya2

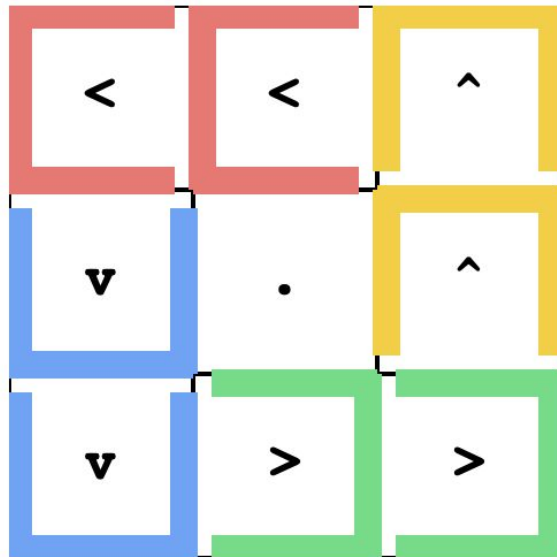
準備:noya2

tester:shobonvip

問題概要

H 行 W 列のグリッドを「コの字」を並べて作れ。

(H,W)=(3,3) の例:



部分点

ペンシルパズルをすると $(H,W)=(2,2),(3,3),(5,5)$ ができるとわかる。

$H = 5, W = 5, \text{コ} = 20, \text{accepted!}$



観察

「コの字」の長さは 3 。H 行 W 列のグリッドに必要な長さは $H(W+1)+W(H+1)$ 。

外側から埋めていくと良さそうで、外側から 2 マス目を埋める際に H,W が奇数でないと困る。

-> $(H,W)=(3,3),(5,5) \bmod 6$ が必要！

外側から 3 マス目まで綺麗に埋めると $(H-6,W-6)$ の問題に帰着できる！



満点

H=W しかできない？ -> WA

外側から 3 マス目を埋める方法に自由度があり、

実は $H, W \geq 9$ では $(H, W) = (3, 3), (5, 5) \pmod{6}$ が構築可能な条件として必要十分。

-6 の帰着で $(H, W) = (9, 9+6w), (11, 11+6w)$ を構築できれば OK !

構築は繰り返し単位を作ると良い。

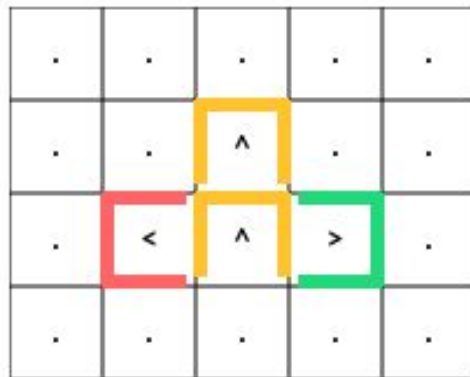


満点の構築

繰り返し単位を探したい。雑に全探索しても時間はかかるが出てくる(大変)。

ペンシルパズルをするなら、次のパーツが役立つ：

4つのコで3つの口を作っている



公式解説の構築 $(H,W) = (9,39)$

H = 9, W = 39, $\square$ = 250, accepted!

[illegible]

裏話

証明コンテスト(後述)で出題したところ simasima に人間解法を出してもらった(4つのコで3つの口を作る発想をベースにしたもの)。

tatyam に visualizer を作ってもらった。綺麗。楽しんでいただけましたか？



Div.1 K - Sum is One

提案: otoshigo

解法提案: tatyam

準備: otoshigo

tester: tassei903, noya2

Div.2 K - 01 Abs Sum

提案: otoshigo

解法提案: tatyam

準備: otoshigo

tester: Astral__

L - Long Sequence Inversion 2

提案: potato167

準備: potato167

tester: ebi_fly

ARC 178 F

本番 1 AC

ごちゃつしてる

短くなって復活！

yuki にも

問題文

正の整数 N, M, K と、長さ M の非負整数列 $A = (A_0, A_1, \dots, A_{M-1})$ が与えられます。ここで、 $2^{N-1} \leq K < 2^N$ が成り立ちます。

入力において、 K は 2 進法表記で N 桁の数として与えられ、その他の整数は 10 進法表記で与えられます。

また、 A は入力では直接与えられず、 $i = 0, 1, \dots, M - 1$ について、 $A_i = \sum_{j=0}^{L_i-1} 2^{X_{i,j}}$ となるような、長さ L_i の整数列 $X_i = (X_{i,0}, X_{i,1}, \dots, X_{i,L_i-1})$ が与えられます。ここで、 $0 \leq X_{i,0} < X_{i,1} < \dots < X_{i,L_i-1} < N$ が成り立ちます。

以下のように定義される、長さ MK の数列 $B = (B_0, B_1, \dots, B_{MK-1})$ の転倒数を 998244353 で割ったあまりを求めてください。

- 任意の 0 以上 K 未満の整数 a と 任意の 0 以上 M 未満の整数 b に対して以下が成り立つ。
 - B_{aM+b} は $\text{popcount}(a \text{ AND } A_b)$ を 2 で割ったあまりに等しい。

1848	Long Prefixes							
1845	Long Substrings	★★★	文字列	 potato167	81	4		
1841	Long Long	★		 potato167	331	0		

問題文

長さ L の順列 P と、 L 個の長さ B の順列 $V_0, V_1, \dots, V_{L-1}$ が与えられます。
長さ B^L の数列 A を以下のように定義します。

各 $0 \leq n < B^L$ に対し、 n を B 進 L 桁で表した際の B^i の位 ($0 \leq i < L$) の値を $N[i]$ とすると、

$$A[n] = \sum_{i=0}^{L-1} V_i[N[i]] \cdot B^{P[i]}$$

である。

数列 A の転倒数を 998244353 で割ったあまりを求めてください。

何をしているのか

短いだけでごちゃついてる

問題文を見てもイメージがわからない...

手を動かそう！



何をしているのか

$$B = 2, L = 3$$

サンプル 1 の例を見てみる まずは B 進法に直す

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1

何をしているのか

$$B = 2, L = 3$$

Vに合わせて変換する

0	0	0	0	1	1	1	1	$V0 = (0, 1)$
0	0	1	1	0	0	1	1	$V0 = (1, 0)$
1	0	1	0	1	0	1	0	$V0 = (1, 0)$

何をしているのか

$$B = 2, L = 3$$

Vに合わせて変換する

0	0	0	0	1	1	1	1	$V0 = (0, 1)$
1	1	0	0	1	1	0	0	$V0 = (1, 0)$
1	0	1	0	1	0	1	0	$V0 = (1, 0)$

何をしているのか

$$B = 2, L = 3$$

Vに合わせて変換する

0	0	0	0	1	1	1	1	$V0 = (0, 1)$
1	1	0	0	1	1	0	0	$V0 = (1, 0)$
1	0	1	0	1	0	1	0	$V0 = (1, 0)$

何をしているのか

$$B = 2, L = 3$$

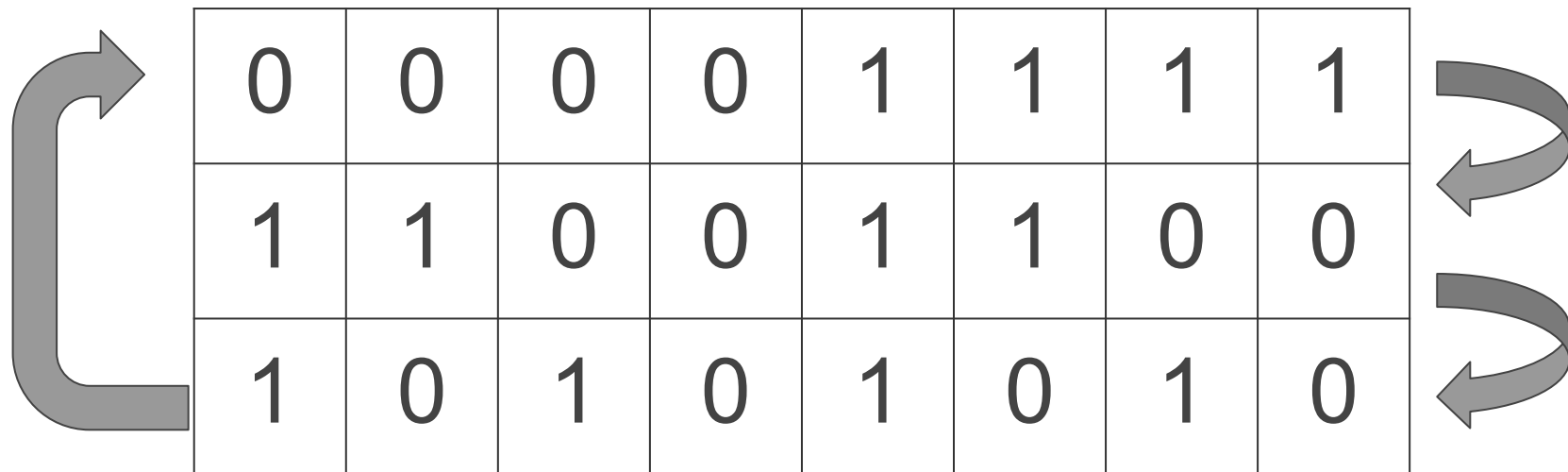
Pに合わせて変換する

0	0	0	0	1	1	1	1	$P[2] = 1$
1	1	0	0	1	1	0	0	$P[1] = 0$
1	0	1	0	1	0	1	0	$P[0] = 2$

何をしているのか

$$B = 2, L = 3$$

Pに合わせて変換する



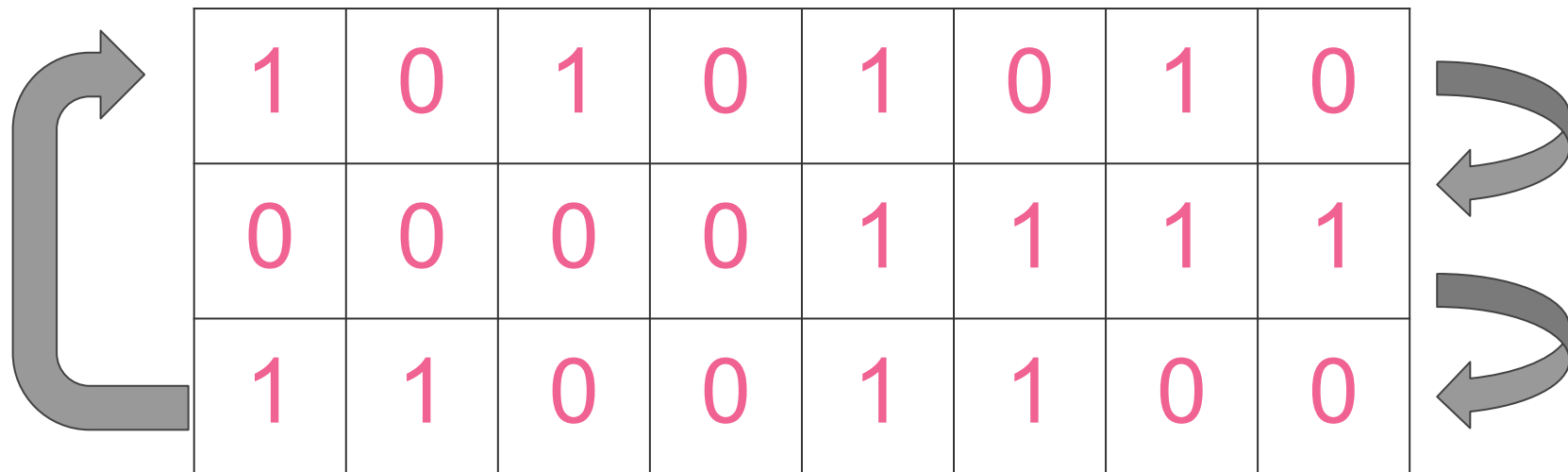
The diagram shows a 3x8 grid of binary digits. A large gray arrow on the left points from the first column to the second column. On the right, two gray arrows point from the first and second columns to the third column, indicating a transformation or shift operation.

0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0

何をしているのか

$$B = 2, L = 3$$

Pに合わせて変換する



The diagram shows a 3x8 grid of binary digits (0s and 1s) in pink. To the left of the grid is a large grey arrow pointing upwards, indicating a vertical shift or wrap-around. To the right of the grid are two grey arrows pointing downwards, indicating a vertical shift or wrap-around. The grid contains the following values:

1	0	1	0	1	0	1	0
0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	0

何をしているのか

戻す サンプル完成！

1	0	1	0	1	0	1	0
0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	0
5	1	4	0	7	3	6	2



答えを求めよう

ans = 0

欲しいのは転倒数 上の桁からソートしてみよう

1	0	1	0	1	0	1	0
0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	0
5	1	4	0	7	3	6	2

答えを求めよう

ans = 0

10101010 の転倒数は 10 。これを安定ソートする

1	0	1	0	1	0	1	0
0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	0
5	1	4	0	7	3	6	2

答えを求めよう

ans = 10

10101010 の転倒数は 10 。これを安定ソートした

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6



答えを求めよう

ans = 10

太枠で囲んだ部分は同じ 片方だけ考え B 倍する

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6

答えを求めよう

ans = 10

0011 の転倒数は 0 これの B 倍を足す

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6

答えを求めよう

ans = 10

変化なし

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6

答えを求めよう

ans = 10

同様に $4 = (B^2)$ つ同じものがある 1 つだけ考える

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6

答えを求めよう

ans = 10

10 転倒数の 1 の 4 倍を足す 桁ごとに安定ソート

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0
1	0	3	2	5	4	7	6

答えを求めよう

ans = 14

ソート終了！ 答えは 14

0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
0	1	2	3	4	5	6	7



何をしてきたか？

$V1 = (1, 0)$ を 1 倍したやつを 4 つ繋げたもの
 $(1, 0, 1, 0, 1, 0, 1, 0)$ の転倒数の 1 倍



何をしてきたか？

$V_2 = (0, 1)$ を 2 倍したやつを 1 つ繋げたもの
 $(0, 0, 1, 1)$ の転倒数の 2 倍

最初の 2 倍というのは、数字 n を 2 個の n に置き換えているという解釈で



何をしてきたか？

$V0 = (1, 0)$ を 1 倍したやつを 1 つ繋げたもの
 $(1, 0)$ の転倒数の 4 倍



一般化

V_i を x_i 倍したやつを y_i つ繋げたものの
の転倒数の z_i 倍

上記の x_i, y_i, z_i が分かれば解けそう



一般化

V_i を $B^{\{a_i\}}$ 倍したやつを $B^{\{b_i\}}$ つ繋げたものの
の転倒数の $B^{\{c_i\}}$ 倍

上記の a_i, b_i, c_i が分かれば解けそう



一般化

$$P = (2, 0, 1)$$

V_i を $B^{\{a_i\}}$ 倍したやつを $B^{\{b_i\}}$ つ繋げたものの
の転倒数の $B^{\{c_i\}}$ 倍

$a_i : j < i$ かつ、 $P[j] < P[i]$ を満たす整数の数

これは、自分より下の桁で、移動後も下の桁である
ような桁の数



一般化

$$P = (2, 0, 1)$$

V_i を B^{a_i} 倍したやつを B^{b_i} つ繋げたものの
の転倒数の B^{c_i} 倍

$$b_i : P[i] - a_i$$

長さは $B^{P[i] + 1}$ だった



一般化

$$P = (2, 0, 1)$$

V_i を $B^{\{a_i\}}$ 倍したやつを $B^{\{b_i\}}$ つ繋げたものの
の転倒数の $B^{\{c_i\}}$ 倍

$$c_i : L - 1 - P[i]$$

どの桁に行くか、つまり P にのみ依存する

計算しよう

V_i を B^{a_i} 倍したやつを B^{b_i} つ繋げたものの
の転倒数の B^{c_i} 倍

それぞれの i について、 a_i, b_i, c_i が求まった！
あとはそれぞれ計算するだけ

計算しよう

V_i を x_i 倍したやつを y_i つ繋げたものの
の転倒数の z_i 倍

それぞれの i について、 x_i, y_i, z_i が求まった！
あとはそれぞれ計算するだけ



計算しよう

$V_i = (1, 0)$ のとき

$$x_i^2 * y_i * (y_i - 1) / 2$$

$V_i = (0, 1)$ のとき

$$x_i^2 * y_i * (y_i + 1) / 2$$



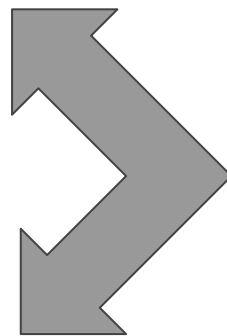
計算しよう

V_i の転倒数を X とする

$$X * xi^{2} * yi * (yi - 1) / 2$$

$$Y := L * (L - 1) / 2 - X$$

$$Y * xi^{2} * yi * (yi + 1) / 2$$



総和を z_i 倍

統計情報

Div. 2 : 7 AC

Div. 1: 18 AC

ucup : 62 AC

足したら 87 AC

多分オンサイトFA : ✨🥳✨🥳✨(1 : 11)

Div.1 M - Cartesian Trees

提案:tassei903

解法提案:tatyam

準備:tatyam

tester:shobonvip

Div.2 M - Colorful Stone Sorting

提案:Rice_tawara459

準備:Rice_tawara459

tester:cho57020

Div.2 N - Adjacent Game

提案: noya2

準備: noya2

tester: hotman78

Div.2 M - Marunomi for All Prefixes

提案: shobonvip

改題: noya2

準備: hotman78

tester: shobonvip

TTPC2024 の作り方

by noya2

自己紹介

AtCoder ID : **noya2**

学年 : 学部 3 年 (入学の年に第 3 期 TTPC 開始)

作問経験 : TTPC 2022/2023/2024

ARC179, yukicoder, etc...



TTPC での作問経験

採用された原案の数

TTPC 2022 : 6

TTPC 2023 : 7

TTPC 2024 : 5



TTPC での作問経験

採用された原案の数

TTPC 2022 : 6

TTPC 2023 : 7

TTPC 2024 : 5

自力で解決した数

TTPC 2022 : 3

TTPC 2023 : 4

TTPC 2024 : 2



ほとんど他人に
解いてもらっている



他力本願(?)作問のメリット・デメリット

メリット

- ・強い人が見てくれる
- ・強い問題ができるかも
- ・~~自分で解く必要はない~~

デメリット

- ・強い人に迷惑がかかる
- ・結局誰も解けないがち
- ・解けるかわからない問題を取り組むモチベがない
- ・強い人が得るものがない

難しい問題の作り方



難しい問題の作り方

1. 見た目が面白そう・すぐには解けない問題をいくつか用意する



難しい問題の作り方

1. 見た目が面白そう・すぐには解けない問題をいくつか用意する
2. 自力でそのうちいくつか解く



難しい問題の作り方

1. 見た目が面白そう・すぐには解けない問題をいくつか用意する
2. 自力でそのうちいくつか解く
3. 赤コーダーを集める



難しい問題の作り方

1. 見た目が面白そう・すぐには解けない問題をいくつか用意する
2. 自力でそのうちいくつか解く
3. 赤コーダーを集める
4. 「いくつかは解けているので、解けるだけ解いてね」と言って、
解けていない問題も含めて出題する

難しい問題の作り方

1. 見た目が面白そう・すぐには解けない問題をいくつか用意する
2. 自力でそのうちいくつか解く
3. 赤コーダーを集める
4. 「いくつかは解けているので、解けるだけ解いてね」と言って、
解けていない問題も含めて出題する
5. 全部解かれる(！？)

デメリットを解消

- ・モチベがない

- コンテスト形式にすることで対戦相手との勝負になる

- 解けている問題も混ぜることで出題者との勝負になる

- ・得るものがない

- 解けている問題もあるので、OK



解法証明コンテストを開催した話

東工大の赤コーダー4人(noshi91, potato167, simasima, tatyam)を集めて開催

運営・出題は noya2, shobonvip

6問の出題のうち、4問がTTPC行き

- Segment Tree (noshi91 による簡潔な解法)
- $A \text{ xor } B \text{ plus } C$ (証明内容は最大値の存在判定だったが、noya2 が求値に強化)
- Diverge and Converge (未解決で出題したところ noshi91 に解かれた)
- Grid Construction (simasima による「繰り返し単位」のまともな発見方法)

解法証明コンテストを開催した話

“競プロの問題” 6 問を 1 日+αで解く(解法を提示し、正当性を証明する)形式

Diverge and Converge のみ出題陣も解けていなかったが、noshi91 に解かれてうれしい、すごい。

A xor B plus C は列 X の要素が最大値を持つ条件についての証明を要求する問題だったが、逆にこれが誰にも解かれなかった。



強い人に頼ると良いことが起こる

Two Convex Set は条件を満たす S の存在判定(+復元)だったが、愚直+枝刈りが強すぎて断念

→ tatyam が数え上げに強化、解法もかなりシンプルに

simasima を解法証明コンテストに招待し、TTPC運営に~~(強引に)~~取り込む

→ 難しい問題を出してくれる(Near pair)



まとめ

チーム作問をするときは
コンテストを開こう

