図書館（ 

）

問題概要

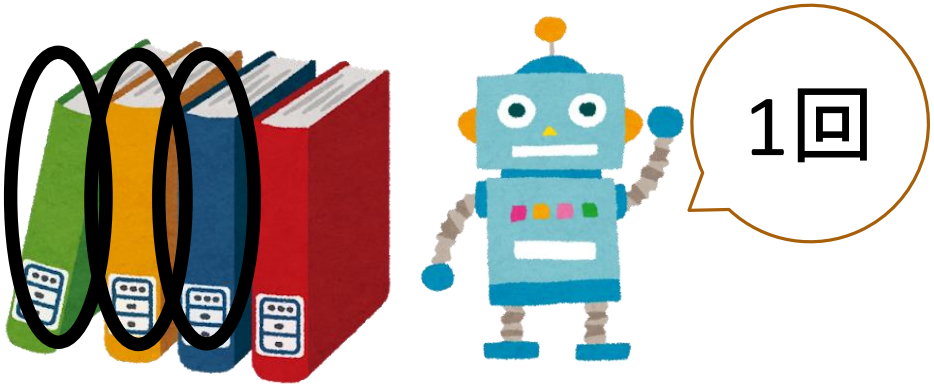
reactiveな問題

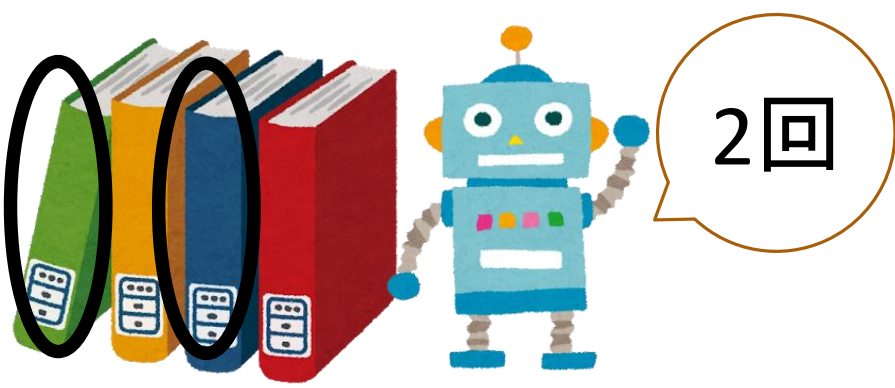
- 番号1から番号NまでのN冊の🕮がある。(1<=N<=1000)
- 1列の本棚にそれらの🕮が無作為に並んでいる。
- クエリを投げて🕮の並び順を特定したい。（左右は逆でもよい）

問題概要

reactiveな問題

- 本棚からは連続して並んでいる1冊以上の🕮を同時に取り出せる。
- **空でない**🕮の集合Mとともにクエリを投げると、クエリで指定した🕮を取り出すために必要な操作の回数を教えてもらえる。
- 20000回以下のクエリで🕮の並び順を特定したい





例

(□□1,□□5,□□2,□□3,□□4)

Query(1,1,1,0,0) -> 2
Query(1,1,1,0,1) -> 1
Query(0,0,1,1,0) -> 1
Query(0,0,0,0,0) -> Wrong Answer

クエリの読み替え

Query(M)は指定した本の集合Mを

すべて取り出すための最小の作業回数が返ってくる

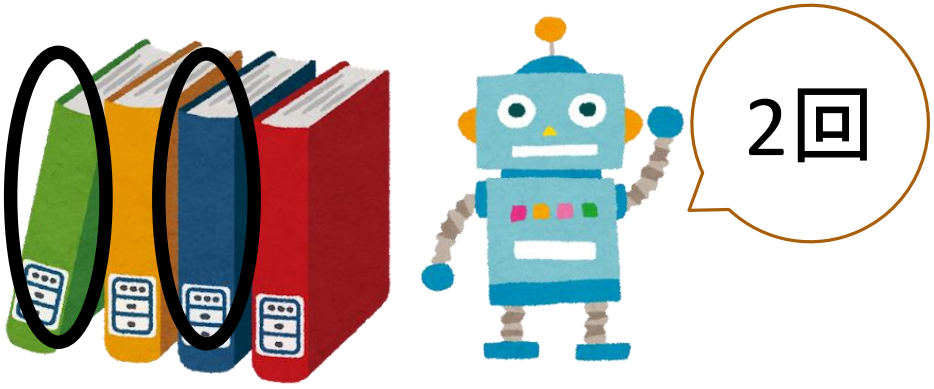
このクエリは状態を変えるわけではないので情報を得ることに意味がある

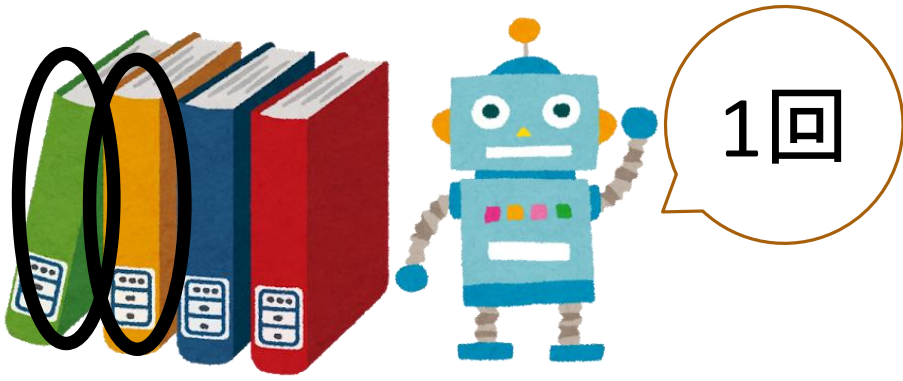
しかしこれは一見何がわかるのかわかりずらい

クエリの読み替え

作業回数は孤立した本の数が増えると大きくなる。

また、同じ📖の数でも、作業回数は隣接する📖が1組あるたびに1回減ることを意識する





クエリの読み替え

つまり、

Query(M)=

Mで指定した📖の数-Mで指定した📖同士で隣接する📖の組の数

であることがわかる

クエリの読み替え

つまり、クエリから隣接する🕮の組の数がわかる。

なので、クエリを投げるとMの中で、隣接する組の数が返ってくると考える(これをAdjacent(M)とする。)

(つまり、Adjacent(M)=|M|-query(M))

例

(□□1,□□5,□□2,□□3,□□4)

Adjacent(1,1,1,0,0) -> 1(Query=2)

Adjacent(1,1,1,0,1) -> 3(Query=1)

Adjacent(0,0,1,1,0) -> 1(Query=1)

基本事項・自明な考察

- 質問できる回数は20000回
- 隣接する📖の組はN-1組
- 指定した📖の集合Mが1冊なら自明にAdjacent(M)=0
- 1組logN回で求められればできそう
- どの📖が端にあるかは簡単にわかる(全集合Aと任意の📖xに対しAdjacent(A-x)はxが端にあるならN-2,そうでないならN-3となり、異なるため）

小課題1(N<=200)

全探索

（全ての2つの📖の組み合わせを投げる）

すると各📖に隣接する📖がわかるので並び順が一意に定まる

小課題1(N<=200)

全探索

（全ての2つの📖の組み合わせを投げる）

${}_nC_2$回クエリを投げればいいので、たかだか19900回でできる

考察

指定する本の集合M(xを含まない)へ要素を追加したときの挙動を考えると

Adjacent(M∪ x)-Adjacent(M)

=📖xに隣接する📖の数

となる

考察

例

Adjacent(1,1,0,0,1)=3

Adjacent(1,1,1,0,1)=4

このとき{□1,□2,□5}のうち

□3に隣接する□の数は上2つの式の差から1つであることがわかる。

考察

これを応用して二分探索すれば、🕮xに隣接する🕮の1つがO(logN)回のクエリでわかる

1. まずxに隣接する🕮の候補をすべて列挙し、その集合をMとする。
2. MをAとBにおよそ均等に分割し、Queryを投げてC=Adjacent(x ∪ A)-Adjacent(x)を求める。
3. Cが1以上ならAを、そうでないならBをMとして2を再び行う。

Mのサイズが1になったとき、そのMに含まれる🕮がxに隣接することがわかる。

満点解法

二分探索

まず端にある📖a_1 を1つ求める

その📖a_1に隣接する📖a_2を二分探索で求める。

その📖a_iに隣接する📖a_{i+1}を二分探索で求める…

を繰り返せばおわり

満点解法

操作回数は

a_1を求めるのにN回

2logN回の二分探索をN回で2NlogN回

合計でN+2NlogN回（ん？）

注意点

雑にやるとN+2NlogN回で怪しいが、適当に節約すれば大丈夫

隣接する📖の候補が1つになったとき、2回ではなく1回で識別するとか

すでに並び順がわかっている📖を省いて探索とか

そもそも、1つの📖を選んでその📖の両隣を見つけて同様のことをすれば2NlogN回でよい

得点分布