T194 Roulette

Problem by Alex Tung

Prepared by Charlie Li

Problem

- Given a circular array with C cells (C = sum of A_i) consists of N groups. Every group is some consecutive cells with the same value.
- For every query, find a position x such that the maximum from the next *LOW* cells to the next *UP* cells is minimized

Observation 1

- Actually, only the range of *LOW* and *UP* is important, ie. *LOW*-*UP*+1
- We can use only range for the query and after finding the answer, we can do some simple calculation to evaluate the desired output.
  - E.g. ((ans - low) % c + c) % c + 1
  - Be careful that x % y may be negative if x is negative

- So from now on, we only consider the range instead of *LOW* and *UP*.

Subtask 1

- *LOW* = *UP*
- So the range is always one.
- Just find the minimum among all *N* groups

Observation 2

- If range >= C then no matter where we start, the maximum value must be the same
- So we may just output any number between 1 to C which is O(1)

- So for now on, we may only consider the case that range < C

Subtask 2

- 1 <= N, M, C <= 100
- For every query, try every starting position and use naïve calculation to find maximum
- O(MC^2)

Subtask 3

- 1 <= N, M, C <= 1000
- We can precompute the maximum for every starting position and every range using naïve calculation
- This part is O(C^2)

- For every query, we just look up the table and calculate the output in O(1).

- So this runs in O(M + C^2)

Observation 3

- We may consider some kind of reverse problem:
  - If we know that a certain element is the maximum element in some continuous cells, what is the largest range of these cells?

- If we can precompute that for every element, we can then use binary search for every query to find the answer in O(lg n)

- So the remaining problem is how to precompute those values

Subtask 5

- We can use some data structure to deal with it.
  - e.g. segment tree or std::map

Subtask 5

- Suppose we use segment tree.
- First, we can sort the groups according to the B_i in descending order.
- Then for the group with largest B, the range is C.

- We then update those groups with largest B in the segment tree (mark those group id to be true).
- Then for the next group, the range is the largest id on the left that are marked true to the smallest id on the right that are marked true.

Subtask 5

- Trick: To simulate a circular array, we can simply put two copy of the array to gather
- When query, query on both cells.
- When update, update both cells.

- Beware: There may be some groups with the same B, you should do the query for all of them before updating any of them.

Subtask 5

- There are also some detail of the algorithm such as calculating the start of the range
- Sometimes, there maybe some smaller B with larger range, remember to handle this case after sorting.

- Subtask 4 requires less detail.

Example

5

1 2 3 4 5

5 4 5 1 1

=>

i	1	2	3	4	5
B[i]	5	4	5	1	1
Max range	inf	2	inf	9	9
Start at	wherever	2	wherever	7	7

=> Binary search on this table: (max range non-decreasing)

B[i]	1	1	4	5	5
Max range	9	9	9	inf	Inf
Start at	7	7	7	wherever	wherever