CSE 4304-Data Structures Lab. Winter 23-24
Batch: CSE 22
Date: March 12, 2025
Target Group: All
Topic: Prefix Sum, Sparse Table

Instructions:

- Regardless of how you finish the lab tasks, you must submit the solutions in Google Classroom. In case I forget to upload the tasks there, CR should contact me. The deadline will always be 11:59 PM on the day the lab took place.
- Task naming format: fullID_T01L01_2A.c/cpp
- If you find any issues in the problem description/test cases, comment in the Google Classroom.
- If you find any tricky test cases that I didn't include but that others might forget to handle, please comment! I'll be happy to add them.
- Use appropriate comments in your code. This will help you recall the solution easily in the future.
- Obtained marks will vary based on the efficiency of the solution.
- Do not use <bits/stdc++.h> library.
- Modified sections will be marked with BLUE color.
- You can use the STL stack unless it's specifically mentioned that you should use manual functions.

Group	**Tasks**
2A	1 2 3
1B	1 2 3
1A	4 5 6
2B	4 5 6
Assignments	2A/1B: 1A/2B:

Task 1: (10 points) Find the Most Common Character in a Substring

Given a string S of length N and Q queries each consisting of two integers L and R (1-indexed), representing a substring S[L, R].

Input:
The first line provides the string. The second line carries the value of Q. The following Q lines provide the values of L and R representing the ranges.

For each query, output the most common character in the substring. If there is a tie, return the lexicographically smallest character.

Sample Input	**Sample output**	**Explanation**
abcccabaaabb		
3		
1 4	c	
2 5	c	
6 12	a	

Task 2: (10 points) Implement the basic operations of a Sparse Table

Given an array of N integers, find the minimum value of a range [L,R] using Sparse Table.

You have to print the following information:

1. The ranges to be stored in the Sparse table
2. Print the Sparse table after construction
3. Answer the result of each query and how the answer was extracted from Sparse Table

Input:
The first line contains the value of N. The Second line provides the N integers.
The third line will contain the Q value, representing the number of queries. The following Q lines will provide the range definitions.

Sample Input	**Sample output**

8
2 3 4 5 3 1 1 3
5
2 6
1 3
1 7
2 7
3 4
	Ranges to be stored in Sparse Table:
	(0,0) (0,1) (0,3) (0,7)
	(1,1) (1,2) (1,4)
	(2,2) (2,3) (2,5)
	(3,3) (3,4) (3,6)
	(4,4) (4,5) (4,7)
	(5,5) (5,6)
	(6,6) (6,7)
	(7,7)
	
	Status of Sparse Table:
	2 2 2 1
	3 3 3
	4 4 1
	5 3 1
	3 1 1
	1 1
	1 1
	3
	
	Query-1: Min=1 min([2,5],[3,6])
	Query-2: Min=3 min([1,2],[2,3])
	Query-3: Min=1 min([1,4],[4,7])
	Query-4: Min=1 min([2,5],[4,7])
	Query-5: Min=3 min([3,3],[4,4])

Task 3: (10 points) THRBL - Catapult that ball

Bob has an unusual problem. In Byteland, there are many hills and cities. The king of Byteland ordered Bob to deliver magic balls from one city to another. Unfortunately, Bob has to deliver many magic balls, so walking with them would take too much time for him. Bob came up with a great idea—catapulting them.

Byteland is divided into intervals. Each interval contains cities and hills.

Bob can catapult a magic ball accurately from city A to city B if there isn't a higher hill between them than A's hill.

Input

- Every test case contains N and M (N ≤ 50000, M ≤ 50000), representing the number of intervals, and number of balls.
- In the next line, there are N numbers H (H ≤ 109) separated by one space.
- In the next M lines numbers A and B (1 ≤ A, B ≤ N), the city from which we want to catapult the ball and the city to which we want to catapult the ball.

Output

Write one number - the number of magic balls that Bob can catapult successfully.

Sample Input	**Sample output**	**Explanation**

7 3
2 3 5 4 2 1 6
3 5
2 5
4 6
	2	Bob can catapult balls numbered 1 and 3.
		[2 5] not possible as there is a hill of height 5 which is larger

6 4
4 2 6 1 3 5
1 3
2 4
5 6
1 6
	2	[2 4] and [1 6] not possible as there is a bigger hill in the middle

4 2
5 5 5 5
1 4
2 3
	2	

Note: We are using 1-based indexing. So [5 4 2] is working with value 4 2 1, meaning index 3 to 5 of the original array.

Task 4: (10 points) Bit Flipper

What is the minimum number of bits you need to flip (change 0 to 1 or 1 to 0) to make the subarray contain an equal number of 0s and 1s?

Input:
You are given a binary array A of length n (containing only 0s and 1s). You need to answer Q queries, where each query is a subarray [L,R] (1 indexed) and you must determine:

Output:
For each query, the number of flips required if possible. (-1 if impossible)

Sample Input	**Sample output**	**Explanation**

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

5 2
0 0 0 0 0
1 5
2 4
	-1
	-1	

4 2
0 1 0 1
1 4
2 3
	0
	0	

1 1
0
1 1
	-1	

7 2
1 1 1 1 1 0 0
1 4
3 7
	2
	-1	

20 10
1 0 1 1 0 0 1 1 0 0 1 0 1 1 0 1 0 0 1 0
1 10
5 15
10 20
1 20
3 12
6 17
2 14
4 13
5 10
	0
	-1
	-1
	0
	0
	0
	-1
	0
	1	

Task 5: (10 points) Implement the basic operations of a Sparse Table

Given an array of N integers, find the **GCD** value of a range [L,R] using Sparse Table.

You have to print the following information:

1. The ranges to be stored in the Sparse table
2. Print the Sparse table after construction
3. Answer the result of each query and how the answer was extracted from Sparse Table

Input:
The first line contains the value of N. The Second line provides the N integers.
The third line will contain the Q value, representing the number of queries. The following Q lines will provide the range definitions.
(for determining GCD use the function __gcd(x,y))

Sample Input	**Sample output**

10
30 10 50 100 20 25 5 15 40 60
5
0 9
2 6
3 7
1 3
4 4
	Ranges to be stored in Sparse Table:
	(0,0) (0,1) (0,3) (0,7)
	(1,1) (1,2) (1,4) (1,8)
	(2,2) (2,3) (2,5) (2,9)
	(3,3) (3,4) (3,6)
	(4,4) (4,5) (4,7)
	(5,5) (5,6) (5,8)
	(6,6) (6,7) (6,9)
	(7,7) (7,8)
	(8,8) (8,9)
	(9,9)
	
	Status of Sparse Table:
	30 10 10 5
	10 10 10 5
	50 50 5 5
	100 20 5
	20 5 5
	25 5 5
	5 5 5
	15 5
	40 20
	60
	
	Query-1: GCD=5 gcd([0,7],[2,9])
	Query-2: GCD=5 gcd([2,5],[3,6])
	Query-3: GCD=5 gcd([3,6],[4,7])
	Query-4: GCD=10 gcd([1,2],[2,3])
	Query-5: GCD=20 gcd([4,4],[4,4])

Task 6: (10 points) Highly Secured Lab

The security system in a mobile lab records **N** security log entries, each represented as an integer. Due to the sensitivity of the logs, each entry contains bitwise information about detected anomalies. The **bitwise OR** operation between a range of logs gives the combined anomaly pattern for that time period.

To verify the integrity of the system, you are given **Q** verification queries. Each query checks whether the **bitwise OR** of a given range matches the **i-th security log entry**. If it matches, print "YES", otherwise, print "NO".

Input:

1. An integer N ($1 \le N \le 10^5$) - the number of security log entries.
2. N space-separated integers A_1, A_2, ..., A_n ($0 \le A_i \le 10^9$) - the security log entries.
3. An integer Q ($1 \le Q \le 10^5$) - the number of verification queries.
4. Q lines follow, each containing two integers L, R and i ($1 \le L \le R \le N$)

Output:

For each query, print "YES" if the bitwise OR of the range **[L, R]** equals **X**, otherwise print "NO".

Sample Input	**Sample output**

8
3 5 7 10 15 8 12 15
4
1 3 4
2 5 5
0 6 7
3 7 6
	YES
	NO
	YES
	NO

8
1 0 1 0 1 0 1 0
5
0 2 2
1 3 3
2 5 5
3 6 6
4 7 7
	YES
	NO
	NO
	YES
	NO