LeetCode Problem: Number of Students Unable to Eat Lunch

Problem Statement

School cafeteria me **circular (0)** aur **square (1)** sandwiches hain. Students ek queue me khade hain, aur unka apna preference hai (0 ya 1). Jo sandwich stack me top pe hai, agar queue ke front wale student ki pasand ka hai to wo leke chala jayega. Agar nahi hai, to wo queue ke end me chala jayega. Ye process tab tak chalega jab tak ya to sab students sandwich leke chale na jayein ya phir koi bhi student bachi hui sandwich na lena chahe.

Mujhe do integer arrays diye gaye hain:

- **students**: har student ki preference (0 ya 1)
- **sandwiches**: stack me sandwiches ka order

Mujhe batana hai ki kitne students **sandwich nahi le payenge**.

Meri Approach

Maine ye logic socha:

1. **Queue ka use karna**: Students ko ek queue me daal diya taki unki **original order** maintain rahe.
2. **Ek ek sandwich process karna**:
    - Agar front student ki pasand top sandwich se match karti hai to wo sandwich leke chala jayega.
    - Agar nahi karti, to wo queue ke end me chala jayega.
3. **Break condition**: Agar **poori queue ghoom gayi** (i.e., koi bhi student sandwich nahi le raha) to process rok denge.

Code Implementation

```
#include <bits/stdc++.h>

using namespace std;


class Solution {

public:

   int countStudents(vector<int>& students, vector<int>& sandwiches) {

      queue <int> q; // Students ki ek queue banai


      int i = 0;
```

```
        while(i <= students.size()-1){ // Sab students ko queue me daala
          q.push(students[i]);
          i++;
        }

        int j = 0; // Sandwich stack ka pointer
        int count = 0; // Track karega ki kitne students ne sandwich accept nahi ki
        while(j <= sandwiches.size()-1 && count <= q.size()){ // Jab tak sandwiches khatam na ho ya koi accept na kare
            if( q.front() == sandwiches[j] ){ // Agar queue ka front sandwich ke top se match karta hai
                count = 0; // Reset counter kyunki ek student ne sandwich le li
                q.pop(); // Student ko queue se hata diya
                j++; // Next sandwich dekhenge
            }
            else{
                count++; // Agar student sandwich nahi le raha to count badh gaya
                int x = q.front(); // Student ko queue ke end me bhej diya
                q.pop();
                q.push(x);
            }
        }
        return q.size(); // Jo students bache wo return karenge
    }
};
```

Dry Run Example

Input:

students = [1,1,0,0]

sandwiches = [0,1,0,1]

Step by Step Execution:

1. **Queue:** [1,1,0,0], **Stack:** [0,1,0,1]

- Front student 1 hai, lekin top sandwich 0 hai → 1 queue ke end me chala gaya.

2. **Queue:** [1,0,0,1], **Stack:** [0,1,0,1]
    - Front student 1 hai, lekin top sandwich 0 hai → 1 queue ke end me chala gaya.
3. **Queue:** [0,0,1,1], **Stack:** [0,1,0,1]
    - Front student 0 hai, aur top sandwich 0 hai → Student ne le li.
4. **Queue:** [0,1,1], **Stack:** [1,0,1]
    - Front student 0 hai, lekin top sandwich 1 hai → 0 queue ke end me chala gaya.
5. **Queue:** [1,1,0], **Stack:** [1,0,1]
    - Front student 1 hai, aur top sandwich 1 hai → Student ne le li.
6. **Queue:** [1,0], **Stack:** [0,1]
    - Front student 1 hai, lekin top sandwich 0 hai → 1 queue ke end me chala gaya.
7. **Queue:** [0,1], **Stack:** [0,1]
    - Front student 0 hai, aur top sandwich 0 hai → Student ne le li.
8. **Queue:** [1], **Stack:** [1]
    - Front student 1 hai, aur top sandwich 1 hai → Student ne le li.

Output:

0 // Sab students ne sandwich le li

Complexity Analysis

- Queue me **O(n)** time laga har student ko push karne me.
- Har student ko **O(n)** time me queue se nikal ke last me dalne ka worst case scenario ho sakta hai.
- **Total complexity: O(n)**

Edge Cases

1. **Jab har student apni pasand ki sandwich mil jaye:**
    - Output: 0
2. **Jab koi bhi student sandwich na le sake:**
    - Example: students = [1,1,1,1], sandwiches = [0,0,0,0]
    - Output: 4
3. **Jab ek hi type ki sandwiches ho aur kuch students different type ki pasand kare:**

- Example: students = [0,0,0,1], sandwiches = [0,0,0,0]
- Output: 1

Conclusion

Meri approach ek simple queue-based approach hai jo **FIFO (First In First Out)** logic follow karti hai. Maine **greedy approach** nahi use ki kyunki mujhe exact student movement track karna tha. Agar koi optimized solution chahiye to hum **0 aur 1 ka count** lekar bhi solve kar sakte hain, jo **O(n) time** me chalega bina queue banaye.

Lekin yeh approach samajhne aur visualize karne me zyada aasaan hai! 🚀