Fibonacci Cubes in Boxes — Problem Description and Code Explanation

Problem Statement

There are n Fibonacci cubes, where the side of the i-th cube is equal to f_i, where f_i is the i-th Fibonacci number.

The Fibonacci numbers are defined as follows:

$$f_1 = 1, \quad f_2 = 2, \quad f_i = f_{i-1} + f_{i-2} \text{ for } i > 2$$

There are also m empty boxes, where the i-th box has a width w_i, a length l_i, and a height h_i.

Goal

For each of the m boxes, you need to determine whether all the cubes can fit inside that box. The cubes must be placed in the box following these rules:

- Cubes can only be stacked such that their sides are parallel to the sides of the box.
- Cubes can be placed either on the bottom of the box or on top of other cubes, filling space from bottom up.
- A larger cube cannot be placed on top of a smaller cube.

Input

- The first line contains a single integer t ($1 \leq t \leq 10^3$) — number of test cases.
- For each test case:
  - One line with two integers n and m ($2 \leq n \leq 10$, $1 \leq m \leq 2 \cdot 10^5$).
  - Then m lines, each with 3 integers w_i, l_i, h_i ($1 \leq w_i, l_i, h_i \leq 150$) — box dimensions.
- The sum of all m across all test cases will not exceed $2 \cdot 10^5$.

Output

For each test case, output a string of length m, where the i-th character is:

- ‘”1”‘ if all n cubes can fit into the i-th box
- ‘”0”‘ otherwise

Example

Input:

```
2
5 4
3 1 2
10 10 10
9 8 13
14 7 20
2 6
3 3 3
1 2 1
2 1 2
3 2 2
2 3 1
3 2 4
```

Output:

```
0010
100101
```

C Code Implementation

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {
    int t;
    scanf("%d", &t);

    while(t--) {
        int n, m;
        scanf("%d %d", &n, &m);

        // Generate the (n+1)-th Fibonacci number (f and g hold f_n and f_{n+1})
        int f = 1, g = 1;
        for(int p = 2; p <= n + 1; p++) {
            int x = f;
            f = g;
            g += x;
        }

        // Allocate output string of length m
        char *s = (char *)malloc((m + 1) * sizeof(char));
        for(int i = 0; i < m; i++) {
            int a, b, c;
            scanf("%d %d %d", &a, &b, &c);

            // Compute the largest dimension
            int d = a;
            if(b > d) d = b;
            if(c > d) d = c;

            s[i] = ’0’;
            // Check if all dimensions are at least f and tallest stack can support g
            if(a >= f && b >= f && c >= f && d >= g) {
                s[i] += 1;  // Make ’0’ into ’1’
            }
        }

        s[m] = ’\0’;
        printf("%s\n", s);
        free(s);
    }
}
```

Code Explanation

- The Fibonacci cubes use $f_1 = 1$, $f_2 = 1$, and so on. This code computes the $(n+1)$-th Fibonacci number since we need:
  - the side length of the n-th cube (for width, length, and height checks),
  - the sum of all heights is upper bounded by the $(n+1)$-th Fibonacci number.
- For each box, we:
  1. Read the box dimensions a, b, c
  2. Determine the largest among them (used to check vertical stacking)
  3. If all dimensions $\geq f_n$ and the tallest side $\geq$ total stacked height (f_{n+1}), we mark the output as ‘”1”‘, else ‘”0”‘