# Offthentic Feed

Problem ID: feed

---

## Problem Statement

You are building a new social media platform called **Offthentic**. Users post content continuously, and the platform must display a live feed of posts.

There are $n$ posts. Each post $i$ has three attributes:

- $u_i$ — the user ID of the author,
- $t_i$ — the timestamp when the post was created (seconds since the start),
- $l_i$ — the number of likes the post has.

The feed has the following rules:

- At each second, all posts created up to that time are available.
- The feed displays at most $k$ posts at a time.
- Posts are ranked by:
  1. Higher likes ($l_i$),
  2. Later timestamp ($t_i$),
  3. Smaller user ID ($u_i$).
- When a new post enters the feed, it may push out an older one. Once a post leaves the feed, it does not reappear later.

It is guaranteed that no two posts have identical $(u_i, t_i)$ pairs, so ties are always resolvable.

Your task is to simulate the feed and output the order in which posts first appeared. **Note:** Some posts may never appear if they are always out-ranked.

## Input

- The first line contains two integers $n$ and $k$ ($1 \leq n, k \leq 10^5$).
- The next $n$ lines each contain three integers $u_i$, $t_i$, and $l_i$ ($1 \leq u_i \leq 10^9$, $0 \leq t_i \leq 10^9$, $0 \leq l_i \leq 10^5$).

## Output

Print the indices of the posts (1-based, in input order) in the order they **first appear** in the feed. If a post never appears, it should not be printed.

## Subtasks

| Subtask | Constraints | Points |
|---|---|---|
| 1 | $n \leq 100$, $k = 1$ | 20 |
| 2 | $n \leq 2000$, $k \leq 10$ | 20 |
| 3 | $n \leq 10^5$, $k \leq 100$ | 30 |
| 4 | No additional constraints | 30 |

## Sample 1

### Input

```
5 2
10 1 5
7 2 5
3 2 8
5 3 5
2 5 10
```

### Output

```
1 3 2 4 5
```

### Explanation

- At $t = 1$, post 1 appears.
- At $t = 2$, posts 2 and 3 are available. Post 3 has more likes, so it enters; post 2 also enters while post 1 is pushed out.
- At $t = 3$, post 4 joins.
- At $t = 5$, post 5 (10 likes) pushes out post 2.

The order of first appearance is: 1, 3, 2, 4, 5.

## Sample 2

### Input

```
4 2
1 1 100
2 2 50
3 3 100
4 4 200
```

### Output

```
1 2 3 4
```

**Explanation**

- At $t = 1$, post 1 appears.
- At $t = 2$, post 2 enters with lower likes but fills the empty slot.
- At $t = 3$, post 3 (likes = 100) replaces post 2 (likes = 50).
- At $t = 4$, post 4 (likes = 200) pushes out post 1.

Final order: 1, 2, 3, 4. Post 2 never reappears after being pushed out, even though newer posts arrive.

---

*End of Problem 3*