QUIZ 1 - The Smart Task Manager

Balancing multiple deadlines can be overwhelming. Whether it's assignments, projects, or daily tasks, keeping track of everything manually often leads to missed deadlines and unnecessary stress. To solve this, you are given a challenge: **design and implement a Task Manager application using C programming and a doubly linked list**.

Application Features:

Your application should be able to:

- **Store tasks** with an ID, name, and deadline.
- **Sort tasks** automatically by deadline in ascending order.
- **Allow task removal** by ID or by finishing the nearest deadline task.
- **Display the current list of tasks** after all operations.

Task Requirements:

Your program must follow these specifications:

1. Input Structure

- The first line of input contains two integers:
  - **N** → Number of tasks to be added.
  - **M** → Number of operations to be performed on the list.
- The next **N** lines contain:
  - **D** → Integer representing the deadline of the task.
  - **S** → String representing the task name (single line).
  - Each task should be stored in a **doubly linked list, sorted by deadline**.
  - Each task has an **auto-incremented ID**, starting from 1.
- After inserting all tasks, the next **M** lines represent operations, which can be:
  - **finishTask** → Removes the task with the earliest deadline.
  - **removeId X** → Removes the task with ID **X**, if it exists.

2. Expected Output

- After performing all operations, the program should print the **remaining tasks** in the following format:
- deadline: D
- task name: S
- id: X

- If an operation tries to remove a non-existent task, print: id X does not exist!

Example Input & Expected Output:

Example 1:

Input:	**Output:**
3 0 3 do homework 1 self study 0 sleep	deadline: 0 task name: sleep id: 3 deadline: 1 task name: self study id: 2 deadline: 3 task name: do homework id: 1

Explanation:

- **Sleep** has the earliest deadline, followed by **self study**, then **do homework**.
- Tasks are displayed in **ascending order of deadlines**.

Example 2:

Input:	**Output:**
3 1 3 do homework 1 self study 0 sleep removeId 3	deadline: 1 task name: self study id: 2 deadline: 3 task name: do homework id: 1

Explanation:

- Task **sleep (ID: 3)** had the earliest deadline but was removed.
- Remaining tasks are displayed **sorted by deadline**.

Example 3:

Input:	**Output:**
5 2 2 do homework 1 self study 3 review 3 quiz finishTask removeId 2	id 2 does not exist! deadline: 2 task name: do homework id: 1 deadline: 3 task name: review id: 3 deadline: 3 task name: quiz id: 4 deadline: 3 task name: quiz id: 5

Explanation:

- Initially, the tasks were sorted by deadline.
- **finishTask** removed **self study (ID: 1, deadline: 1)**.
- **removeId 2** failed because **task ID 2 was already removed**.
- The remaining tasks were displayed **sorted by deadline**.

Submission Format:

Submit your code using this file format: **Class_NIM_NAME.C**

Example: LA05_2802200001_Dature.C

Happy coding! 🎯💻