**Case: Multi-Level Linkedlist**

You are tasked with implementing a data structure to represent the organizational hierarchy of a company. Each employee is represented by a node in a multi-level linked list, with each node having a name and pointers to its direct subordinates and the next employee at the same level.

**Given Code:**

The provided C code defines a Node structure to represent employees, along with functions to create a new node, add subordinates to a manager, print the organizational hierarchy, and free all nodes in the hierarchy.

**Explanation:**

The Node structure has two pointers:

**next**: Points to the next employee at the same level (i.e., the next employee reporting to the same manager).

**subordinate**: Points to a linked list of subordinates (i.e., employees reporting directly to the current employee).

The **createNode** function creates a new node with the given name.

The **addSubordinate** function adds a subordinate node to a given manager. It creates a new subordinate node and adds it to the beginning of the manager's subordinate list.

The **printList** function prints the organizational hierarchy. It starts from the CEO and prints each manager along with their direct subordinates.

The **fireAll** function frees all nodes in the organizational hierarchy. It uses recursion to free subordinates first and then moves to the next employee at the same level.

A screenshot of a computer code

Description automatically generated

The main function, which could look like this, demonstrates the usage by creating a CEO node ("Frieren") and adding direct subordinates ("Eisen" and "Heiter"). It then adds subordinates to "Eisen" and "Heiter" as well, forming a multi-level hierarchy.

**Example Output:**

The output of the provided code will be:

A black screen with white text

Description automatically generated

This represents the organizational hierarchy with "Frieren" as the CEO, "Heiter" and "Eisen" as direct subordinates, and "Stark" as a subordinate of "Eisen."

After printing the hierarchy, the **fireAll** function is called to free all allocated memory.

Tips: you could utilize recursive for printList all subordinates.