Rajalakshmi Engineering College

Name: Jhanani shree
Email: 240701215@rajalakshmi.edu.in
Roll no: 240701215
Phone: 7373333511
Branch: REC
Department: I CSE AH
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bob is tasked with developing a company's employee record management system. The system needs to maintain a list of employee records using a doubly linked list. Each employee is represented by a unique integer ID.

Help Bob to complete a program that adds employee records at the front, traverses the list, and prints the same for each addition of employees to the list.

Input Format

The first line of input consists of an integer N, representing the number of employees.

The second line consists of N space-separated integers, representing the employee IDs.

Output Format

For each employee ID, the program prints "Node Inserted" followed by the current state of the doubly linked list in the next line, with the data values of each node separated by spaces.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
101 102 103 104
Output: Node Inserted
101
Node Inserted
102 101
Node Inserted
103 102 101
Node Inserted
104 103 102 101

Answer

```
#include <iostream>
using namespace std;

struct node {
    int info;
    struct node* prev, * next;
};

struct node* start = NULL;
```

```
# You are using Python
class Node:
    def __init__(self, emp_id):
        self.emp_id = emp_id  # Employee ID
        self.next = None      # Pointer to the next node
        self.prev = None      # Pointer to the previous node


class EmployeeList:
```

```
    def __init__(self):
        self.head = None      # Initialize the head of the list

    def insert_at_front(self, emp_id):
        new_node = Node(emp_id)  # Create a new node
        new_node.next = self.head  # Point new node's next to current head

        if self.head is not None:
            self.head.prev = new_node  # Update current head's previous pointer

        self.head = new_node  # Update head to new node

        # Print the current state of the list
        self.print_list()

    def print_list(self):
        current = self.head
        print("Node Inserted")

        # Traverse the list and collect data for printing
        ids = []
        while current is not None:
            ids.append(str(current.emp_id))
            current = current.next

        print(" ".join(ids))  # Print the collected employee IDs


def main():
    N = int(input())  # Read the number of employees
    employee_list = EmployeeList()  # Create an employee list

    emp_ids = list(map(int, input().split()))  # Read employee IDs

    for emp_id in emp_ids:
        employee_list.insert_at_front(emp_id)  # Insert each ID at the front


if __name__ == "__main__":
    main()
int main() {
    int n, data;
```

```
    cin >> n;
    for (int i = 0; i < n; ++i) {
        cin >> data;
        insertAtFront(data);
        traverse();
    }
    return 0;
}
```

Status : Correct ***Marks : 10/10***