

# Rajalakshmi Engineering College

Name: SANJAY V  
Email: 241801247@rajalakshmi.edu.in  
Roll no: 241801247  
Phone: 7397492247  
Branch: REC  
Department: I AI & DS FD  
Batch: 2028  
Degree: B.E - AI & DS

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 2\_COD\_Question 2

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Moniksha, a chess coach organizing a tournament, needs a program to manage participant IDs efficiently. The program maintains a doubly linked list of IDs and offers two functions: Append to add IDs as students register, and Print Maximum ID to identify the highest ID for administrative tasks.

This tool streamlines tournament organization, allowing Moniksha to focus on coaching her students effectively.

##### ***Input Format***

The first line consists of an integer  $n$ , representing the number of participant IDs to be added.

The second line consists of  $n$  space-separated integers representing the participant IDs.

### **Output Format**

The output displays a single integer, representing the maximum participant ID.

If the list is empty, the output prints "Empty list!".

Refer to the sample output for the formatting specifications.

### **Sample Test Case**

Input: 3

163 137 155

Output: 163

### **Answer**

```
#include <stdio.h>
#include <stdlib.h>
typedef struct Node {
    int data;
    struct Node* prev;
    struct Node* next;
} Node;
Node* createNode(int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->prev = NULL;
    newNode->next = NULL;
    return newNode;
}
void append(Node** head, int data) {
    Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }

    Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;
```

```

    temp->next = newNode;
    newNode->prev = temp;
}
void printMaxID(Node* head) {
    if (head == NULL) {
        printf("Empty list!\n");
        return;
    }

    int max = head->data;
    Node* temp = head->next;
    while (temp != NULL) {
        if (temp->data > max)
            max = temp->data;
        temp = temp->next;
    }
    printf("%d\n", max);
}

int main() {
    int n, value;
    Node* head = NULL;
    scanf("%d", &n);

    if (n == 0) {
        printf("Empty list!\n");
        return 0;
    }
    for (int i = 0; i < n; i++) {
        scanf("%d", &value);
        append(&head, value);
    }
    printMaxID(head);

    return 0;
}

```

**Status :** Correct

**Marks :** 10/10