

# Rajalakshmi Engineering College

Name: eswar rs  
Email: 240801074@rajalakshmi.edu.in  
Roll no: 240801074  
Phone: 7845648127  
Branch: REC  
Department: I ECE AE  
Batch: 2028  
Degree: B.E - ECE

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**

```
// You are using GCC
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
struct Node {  
    int id;  
    struct Node* next;  
    struct Node* prev;  
};
```

```
struct Node* createNode(int id) {  
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));  
    newNode->id = id;  
    newNode->next = NULL;  
    newNode->prev = NULL;  
    return newNode;  
}
```

```
void append(struct Node** head, int id) {  
    struct Node* newNode = createNode(id);  
    if (*head == NULL) {  
        *head = newNode;  
    } else {  
        struct Node* temp = *head;  
        while (temp->next != NULL) {
```

```
        temp = temp->next;
    }
    temp->next = newNode;
    newNode->prev = temp;
}
}
```

```
int findMaxID(struct Node* head) {
    if (head == NULL) {
        return -1;
    }
    int maxID = head->id;
    struct Node* temp = head->next;
    while (temp != NULL) {
        if (temp->id > maxID) {
            maxID = temp->id;
        }
        temp = temp->next;
    }
    return maxID;
}
```

```
void freeList(struct Node* head) {
    struct Node* temp;
    while (head != NULL) {
        temp = head;
        head = head->next;
        free(temp);
    }
}
```

```
int main() {
    int n;
    scanf("%d", &n);

    struct Node* head = NULL;

    if (n == 0) {
        printf("Empty list!\n");
        return 0;
    }
}
```

```
for (int i = 0; i < n; i++) {  
    int id;  
    scanf("%d", &id);  
    append(&head, id);  
}  
  
int maxID = findMaxID(head);  
if (maxID == -1) {  
    printf("Empty list!\n");  
} else {  
    printf("%d\n", maxID);  
}  
  
freeList(head);  
return 0;  
}
```

**Status :** Correct

**Marks :** 10/10