

# Rajalakshmi Engineering College

Name: Sowmyalakshmi N  
Email: 240701524@rajalakshmi.edu.in  
Roll no: 240701524  
Phone: 9003767185  
Branch: REC  
Department: I CSE FE  
Batch: 2028  
Degree: B.E - CSE

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>
```

```
// Structure for each node
```

```
struct node {
```

```
    int id;
```

```
    struct node* prev;
```

```
    struct node* next;
```

```
};
```

```
// Function to append a new ID at the end
```

```
void append(struct node** head, struct node** tail, int id) {
```

```
    struct node* newNode = (struct node*)malloc(sizeof(struct node));
```

```
    newNode->id = id;
```

```
    newNode->prev = *tail;
```

```
    newNode->next = NULL;
```

```
    if (*tail)
```

```
        (*tail)->next = newNode;
```

```
    else
```

```
        *head = newNode;
```

```
    *tail = newNode;
```

```
}
```

```
// Function to find and print the maximum ID
void printMaxID(struct node* head) {
```

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

```
    int maxID = head->id;
    struct node* temp = head->next;
```

```
    while (temp != NULL) {
        if (temp->id > maxID)
            maxID = temp->id;
        temp = temp->next;
    }
```

```
    printf("%d\n", maxID);
}
```

```
// Function to free memory
void freeList(struct node* head) {
```

```
    struct node* temp;
    while (head) {
        temp = head;
        head = head->next;
        free(temp);
    }
}
```

```
// Main function
```

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

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

```
    struct node* head = NULL;
    struct node* tail = NULL;
```

```
for (int i = 0; i < n; i++) {  
    int id;  
    scanf("%d", &id);  
    append(&head, &tail, id);  
}  
  
printMaxID(head);  
freeList(head);  
return 0;  
}
```

**Status :** Correct

**Marks :** 10/10