

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 1

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Janani is a tech enthusiast who loves working with polynomials. She wants to create a program that can add polynomial coefficients and provide the sum of their coefficients.

The polynomials will be represented as a linked list, where each node of the linked list contains a coefficient and an exponent. The polynomial is represented in the standard form with descending order of exponents.

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

The first line of input consists of an integer  $n$ , representing the number of terms in the first polynomial.

The following  $n$  lines of input consist of two integers each: the coefficient and the exponent of the term in the first polynomial.

The next line of input consists of an integer m, representing the number of terms in the second polynomial.

The following m lines of input consist of two integers each: the coefficient and the exponent of the term in the second polynomial.

### **Output Format**

The output prints the sum of the coefficients of the polynomials.

### **Sample Test Case**

Input: 3

2 2

3 1

4 0

3

2 2

3 1

4 0

Output: 18

### **Answer**

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int coef;
    int exp;
    struct node*next;
};
struct node*createnode(int coef,int exp){
    struct node*newnode=(struct node*)malloc(sizeof(struct node));
    newnode->coef=coef;
    newnode->exp=exp;
    newnode->next=NULL;
    return newnode;
}
void insertnode(struct node**head,int coef,int exp){
    struct node*newnode=createnode(coef,exp);
    if(*head==NULL){
        *head=newnode;
    }
    else{
```

```

    struct node*temp=*head;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    temp->next=newnode;
}
}

struct node*addpolynomials(struct node*poly1,struct node*poly2){
    struct node*result=NULL;
    struct node*p1=poly1,*p2=poly2;
    while(p1!=NULL||p2!=NULL){
        if(p1!=NULL && (p2==NULL || p1->exp>p2->exp)){
            insertnode(&result,p1->coef,p1->exp);
            p1=p1->next;
        }
        else if(p2!=NULL && (p1==NULL||p2->exp>p1->exp)){
            insertnode(&result,p2->coef,p2->exp);
            p2=p2->next;
        }
        else{
            int sumcoef=p1->coef+p2->coef;
            if(sumcoef!=0){
                insertnode(&result,sumcoef,p1->exp);
            }
            p1=p1->next;
            p2=p2->next;
        }
    }
    return result;
}

int sumcoefficients(struct node*head){
    int sum=0;
    struct node*temp=head;
    while(temp!=NULL){
        sum+=temp->coef;
        temp=temp->next;
    }
    return sum;
}

void freepolynomial(struct node*head){
    struct node*temp;
    while(head!=NULL){

```

```

    temp=head;
    head=head->next;
    free(temp);
}
}
int main(){
    int n,m;
    struct node*poly1=NULL,*poly2=NULL;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        int coef,exp;
        scanf("%d %d",&coef,&exp);
        insertnode(&poly1,coef,exp);
    }
    scanf("%d",&m);
    for(int i=0;i<m;i++){
        int coef,exp;
        scanf("%d %d",&coef,&exp);
        insertnode(&poly2,coef,exp);
    }
    struct node*result=addpolynomials(poly1,poly2);
    printf("%d\n",sumcoefficients(result));
    freepolynomial(poly1);
    freepolynomial(poly2);
    freepolynomial(result);
    return 0;
}

```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 2

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Arun is learning about data structures and algorithms. He needs your help in solving a specific problem related to a singly linked list.

Your task is to implement a program to delete a node at a given position. If the position is valid, the program should perform the deletion; otherwise, it should display an appropriate message.

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

The first line of input consists of an integer N, representing the number of elements in the linked list.

The second line consists of N space-separated elements of the linked list.

The third line consists of an integer x, representing the position to delete.

Position starts from 1.

### **Output Format**

The output prints space-separated integers, representing the updated linked list after deleting the element at the given position.

If the position is not valid, print "Invalid position. Deletion not possible."

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

8 2 3 1 7

2

Output: 8 3 1 7

### **Answer**

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

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

```
void insert(int);
```

```
void display_List();
```

```
void deleteNode(int);
```

```
struct node {
```

```
    int data;
```

```
    struct node* next;
```

```
} *head = NULL, *tail = NULL;
```

```
void insert(int value){
```

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

```
    newnode->data=value;
```

```
    newnode->next=NULL;
```

```
    if(head==NULL){
```

```
        head=newnode;
```

```
        tail=newnode;
```

```
    }
```

```
    else{
```

```
        tail->next=newnode;
```

```

        tail=newnode;
    }
}

void display_List(){
    struct node*temp=head;
    while(temp!=NULL){
        printf("%d",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

void deleteNode(int position){
    if(head==NULL){
        printf("Invalid position. Deletion not possible.\n");
        return;
    }
    struct node*temp=head;

    if(position==1){
        head=head->next;
        free(temp);
        display_List();
        return;
    }
    struct node*prev=NULL;
    for(int i=1;temp!=NULL && i<position;i++){
        prev=temp;
        temp=temp->next;
    }
    if(temp==NULL){
        printf("Invalid position.Deletion not possible.\n");
        return;
    }
    prev->next=temp->next;
    if(temp==tail){
        tail=prev;
    }
    free(temp);
    display_List();
}

```

```
int main() {
    int num_elements, element, pos_to_delete;

    scanf("%d", &num_elements);

    for (int i = 0; i < num_elements; i++) {
        scanf("%d", &element);
        insert(element);
    }

    scanf("%d", &pos_to_delete);

    deleteNode(pos_to_delete);

    return 0;
}
```

**Status :** Correct

**Marks :** 10/10



# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 3

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Imagine you are working on a text processing tool and need to implement a feature that allows users to insert characters at a specific position.

Implement a program that takes user inputs to create a singly linked list of characters and inserts a new character after a given index in the list.

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

The first line of input consists of an integer N, representing the number of characters in the linked list.

The second line consists of a sequence of N characters, representing the linked list.

The third line consists of an integer index, representing the index(0-based) after

which the new character node needs to be inserted.

The fourth line consists of a character value representing the character to be inserted after the given index.

### ***Output Format***

If the provided index is out of bounds (larger than the list size):

1. The first line of output prints "Invalid index".
2. The second line prints "Updated list: " followed by the unchanged linked list values.

Otherwise, the output prints "Updated list: " followed by the updated linked list after inserting the new character after the given index.

Refer to the sample output for formatting specifications.

### ***Sample Test Case***

Input: 5

a b c d e

2

X

Output: Updated list: a b c X d e

### ***Answer***

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

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

```
struct Node{  
    char data[10];  
    struct Node* next;  
};
```

```
struct Node* insert(struct Node* head, char value, int pos){  
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));  
    newNode->data[0] = value;  
    newNode->data[1] = '\0';
```

```
if (pos == 1){
    newNode->next = head;
    head = newNode;
    return head;
}
```

```
newNode->next = NULL;
int i = 1;
struct Node* temp = head;
while (i < pos - 1 && temp != NULL){
    temp = temp->next;
    i++;
}
```

```
if (temp == NULL){
    printf("Invalid index\n");
    free(newNode);
    return head;
}
```

```
newNode->next = temp->next;
temp->next = newNode;
return head;
}
```

```
void printList(struct Node* head){
    struct Node* temp = head;
    while (temp != NULL){
        printf("%c ", temp->data[0]);
        temp = temp->next;
    }
    printf("\n");
}
```

```
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;
```

```
    for (int i = 1; i <= n; i++){
```

```
        char c;
```

```
        scanf(" %c", &c);
```

```
        head = insert(head, c, i);
```

```
    }
```

```
    int pos;
```

```
    scanf("%d", &pos);
```

```
    if (pos > 0){
```

```
        pos += 2;
```

```
    }
```

```
    char c2;
```

```
    scanf(" %c", &c2);
```

```
    head = insert(head, c2, pos);
```

```
    printf("Updated list: ");
```

```
    printList(head);
```

```
    freeList(head);
```

```
    return 0;
```

```
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 4

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

As part of a programming assignment in a data structures course, students are required to create a program to construct a singly linked list by inserting elements at the beginning.

You are an evaluator of the course and guide the students to complete the task.

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

The first line of input consists of an integer N, which is the number of elements.

The second line consists of N space-separated integers.

##### ***Output Format***

The output prints the singly linked list elements, after inserting them at the beginning.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

78 89 34 51 67

Output: 67 51 34 89 78

### **Answer**

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

```
struct Node {
    int data;
    struct Node* next;
};
```

```
void insertAtFront(struct Node**head,int data){
    struct Node*newNode=(struct Node*)malloc(sizeof(struct Node));
    if(newNode==NULL){
        printf("Memory allocation failed\n");
        return;
    }
    newNode->data=data;
    newNode->next=*head;
    *head=newNode;
}
```

```
void printList(struct Node*head){
    struct Node*current=head;
    while(current!=NULL){
        printf("%d",current->data);
        current=current->next;
    }
    printf("\n");
}
```

```
int main(){
```

```
struct Node* head = NULL;

int n;
scanf("%d", &n);

for (int i = 0; i < n; i++) {
    int activity;
    scanf("%d", &activity);
    insertAtFront(&head, activity);
}

printList(head);
struct Node* current = head;
while (current != NULL) {
    struct Node* temp = current;
    current = current->next;
    free(temp);
}

return 0;
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 5

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Imagine you are tasked with developing a simple GPA management system using a singly linked list. The system allows users to input student GPA values, insertion should happen at the front of the linked list, delete record by position, and display the updated list of student GPAs.

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

The first line of input contains an integer  $n$ , representing the number of students.

The next  $n$  lines contain a single floating-point value representing the GPA of each student.

The last line contains an integer position, indicating the position at which a student record should be deleted. Position starts from 1.



### **Output Format**

After deleting the data in the given position, display the output in the format "GPA: " followed by the GPA value, rounded off to one decimal place.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 4

3.8

3.2

3.5

4.1

2

Output: GPA: 4.1

GPA: 3.2

GPA: 3.8

### **Answer**

```
#include<stdio.h>
#include<stdlib.h>
typedef struct node{
    float gpa;
    struct node*next;
}node;
void insertfront(node**head,float gpa){
    node*newnode=(node*)malloc(sizeof(node));
    newnode->gpa=gpa;
    newnode->next=*head;
    *head=newnode;
}
void deleteatposition(node**head,int position){
    if(*head==NULL) return;
    node*temp=*head;
    if(position==1){
        *head=temp->next;
        free(temp);
        return;
    }
```

```

    for(int i=1;temp!=NULL && i<position-1;i++){
        temp=temp->next;
    }
    if(temp==NULL || temp->next==NULL) return;
    node*next=temp->next->next;
    free(temp->next);
    temp->next=next;
}
void displaylist(node*head){
    while(head!=NULL){
        printf("GPA: %.1f\n",head->gpa);
        head=head->next;
    }
}
int main(){
    int n,position;
    float gpa;
    node*head=NULL;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%f",&gpa);
        insertfront(&head,gpa);
    }
    scanf("%d",&position);
    deleteatposition(&head,position);
    displaylist(head);
    return 0;
}

```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 6

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

John is tasked with creating a program to manage student roll numbers using a singly linked list.

Write a program for John that accepts students' roll numbers, inserts them at the end of the linked list, and displays the numbers.

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

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

The second line consists of N space-separated integers, representing the roll numbers of students.

##### ***Output Format***

The output prints the space-separated integers singly linked list, after inserting the roll numbers of students at the end.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

23 85 47 62 31

Output: 23 85 47 62 31

### **Answer**

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node*next;
};
void insertatend(struct node**head,int data){
    struct node*newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=data;
    newnode->next=NULL;
    if(*head==NULL){
        *head=newnode;
        return;
    }
    struct node*temp=*head;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    temp->next=newnode;
}
void display(struct node*head){
    struct node*temp=head;
    while(temp!=NULL){
        printf("%d",temp->data);
        temp=temp->next;
    }
    printf("\n");
}
```

```
int main(){
    int n,value;
    struct node*head=NULL;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&value);
        insertatend(&head,value);
    }
    display(head);
    return 0;
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Sanjai k  
Email: 241501178@rajalakshmi.edu.in  
Roll no: 241501178  
Phone: 7904386392  
Branch: REC  
Department: I AIML AD  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 1\_COD\_Question 7

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Dev is tasked with creating a program that efficiently finds the middle element of a linked list. The program should take user input to populate the linked list by inserting each element into the front of the list and then determining the middle element.

Assist Dev, as he needs to ensure that the middle element is accurately identified from the constructed singly linked list:

If it's an odd-length linked list, return the middle element. If it's an even-length linked list, return the second middle element of the two elements.

##### **Input Format**

The first line of input consists of an integer n, representing the number of elements in the linked list.

The second line consists of n space-separated integers, representing the elements of the list.

### ***Output Format***

The first line of output displays the linked list after inserting elements at the front.

The second line displays "Middle Element: " followed by the middle element of the linked list.

Refer to the sample output for formatting specifications.

### ***Sample Test Case***

Input: 5

10 20 30 40 50

Output: 50 40 30 20 10

Middle Element: 30

### ***Answer***

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

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

```
struct Node {
```

```
    int data;
```

```
    struct Node* next;
```

```
};
```

```
struct Node*push(struct Node*head,int data){
```

```
    struct Node*new_node=(struct Node*)malloc(sizeof(struct Node));
```

```
    new_node->data=data;
```

```
    new_node->next=head;
```

```
    return new_node;
```

```
}
```

```
int printMiddle(struct Node*head){
```

```
    struct Node*slow=head,*fast=head;
```

```
    while(fast!=NULL && fast->next!=NULL){
```

```
        slow=slow->next;
```

```
        fast=fast->next->next;
```

```

    }
    return slow->data;
}

int main() {
    struct Node* head = NULL;
    int n;

    scanf("%d", &n);
    int value;

    for (int i = 0; i < n; i++) {
        scanf("%d", &value);
        head = push(head, value);
    }

    struct Node* current = head;
    while (current != NULL) {
        printf("%d ", current->data);
        current = current->next;
    }
    printf("\n");

    int middle_element = printMiddle(head);
    printf("Middle Element: %d\n", middle_element);

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

    return 0;
}

```

**Status :** Correct

**Marks : 10/10**