

# Rajalakshmi Engineering College

Name: karthieswaran L

Email: 241801117@rajalakshmi.edu.in

Roll no: 241801117

Phone: 9840695531

Branch: REC

Department: I AI & DS FB

Batch: 2028

Degree: B.E - AI & DS

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

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

Attempt : 2

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***

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
typedef struct Char {
    char value;
    struct Char*next;
}Node;
Node*newnode(char value){
    Node*new_node = (Node*) malloc(sizeof(Node));
    new_node->value = value;
    new_node->next=NULL;
    return new_node;
```

```

}
void insertNode(Node** head, char value) {
    Node* temp = *head;
    if (temp == NULL)
    {
        *head = newnode(value);
        return;
    }
    while (temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = newnode(value);
}
int length(Node* head) {
    int len = 0;
    while (head != NULL) {
        head = head->next;
        len++;
    }
    return len;
}
void traverse(Node* head) {
    while (head != NULL) {
        printf("%c", head->value);
        head = head->next;
    }
    printf("\n");
}
void insert(Node** head, int pos, char value) {
    if (pos >= length(*head)) {
        printf("Invalid index\n");
        return;
    }
    Node* temp = *head;
    for (int i = 0; i < pos; i++)
    {
        temp = temp->next;
    }
    Node* new_node = newnode(value);
    new_node->next = temp->next;
    temp->next = new_node;
}

```

```
int main()
{
    int n;
    char value;
    Node*head=NULL;
    scanf("%d",&n);
    for(int i=0;i<=n;i++){
        scanf("%c",&value);
        if(value == ' ' || value == '\n') {
            continue;
        }
        insertNode(&head,value);
    }
    scanf("%d %c",&n,&value);
    insert(&head,n,value);
    printf("Updated list: ");
    traverse(head);
}
```

**Status :** Correct

**Marks :** 10/10