Lab Session 5

Question # 1

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

struct Contact {
    char name[50];
    char email[50];
    char phone[20];
};

void addContact(struct Contact **contacts, int *count){
    *count += 1;
    *contacts = (struct Contact *)realloc(*contacts, (*count) *sizeof(struct Contact));

    if (*contacts == NULL){
        fprintf(stderr, "Memory Allocation failed");
        exit(1);
    }

    printf("\nEnter name: ");
    scanf(" %[^\n]", (*contacts)[*count - 1].name);
    printf("\nEnter email: ");
    scanf(" %[^\n]", (*contacts)[*count - 1].email);
    printf("\nEnter phone: ");
    scanf(" %[^\n]", (*contacts)[*count - 1].phone);

    printf("Contact added successfully ✅\n");
}
void displayContact(struct Contact *contacts, int count){
    if (count == 0){
        printf("\nNo contacts found\n");
        return;
    }

    printf("Address Book:\n");
    for (int i=0; i <count; i++){
        printf("Contact Info #%d:\n", i + 1);
        printf("Name: %s\n", contacts[i].name);
        printf("Email: %s\n", contacts[i].email);
        printf("Phone: %s\n", contacts[i].phone);
    }

}
```

Lab Session 5

Question # 1

```c
void deleteContact(struct Contact **contacts, int *count){
    if (*count == 0){
        printf("\nNo counts to be deleted.\n");
        return;
    }

    int index;
    printf("Enter contact to be deleted: ");
    scanf("%d", &index);

    if (index<1 || index > *count){
        printf("Invalid contact number\n");
        return;
    }

    for(int i= index-1; i < *count-1; i++){
        (*contacts)[i] = (*contacts)[i + 1];
    }

    *count -= 1;

    *contacts = (struct Contact*)realloc(*contacts, (*count)*sizeof(struct Contact));

    if(*count >0 && *contacts == NULL){
        fprintf(stderr,"Memory Allocation failed\n");
        exit(1);
    }

    printf("\nContact Deleted Successfully\n");
}
int main(){
    struct Contact *contacts = NULL;
    int count = 0;
    int choice;

    while(1){
        printf("---Address Book----\n");
        printf("1. Add Contact\n");
        printf("2. Display Contacts\n");
        printf("3. Delete Contact\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice){
```

Lab Session 5

Question # 1

```
            case 1:
                addContact(&contacts, &count);
                break;

            case 2:
                displayContact(contacts, count);
                break;

            case 3:
                deleteContact(&contacts, &count);
                break;

            case 4:
                free(contacts);
                printf("Exiting... Memory Freed Successfully!\n");
                return 0;

            default:
                printf("Invalid choice. Please try again.");
        }
    }
}
```

```
---Address Book----
1. Add Contact
2. Display Contacts
3. Delete Contact
4. Exit
Enter your choice: 1

Enter name: Usman

Enter email: usman@email.com

Enter phone: 03348728312
Contact added successfully ✅
```

```
Enter your choice: 2

Address Book:
Contact Info #1:
Name: Usman
Email: usman@email.com
Phone: 03348728312

Enter your choice: 3
Enter contact to be deleted: 1

Contact Deleted Successfully
```

Question # 2

```c
#include <stdio.h>
#include "LinkedList.h"


struct Node *MergeLL(struct Node *head1, struct Node *head2){

    struct Node *current1 = head1;
    struct Node *current2 = head2;

    if(current1 == NULL){
        return current2;
    }

    if(current2 == NULL){
        return current1;
    }

    while(current1->next !=NULL){
        current1 = current1->next;
    }

    current1->next = head2;

    return head1;
}
```

```c
int main() {
    struct Node *head1 = NULL;
    struct Node *head2 = NULL;
    struct Node *merged = NULL;

    head1 = insertAtEnd(head1, 1);
    head1 = insertAtEnd(head1, 2);
    head1 = insertAtEnd(head1, 3);

    head2 = insertAtEnd(head2, 4);
    head2 = insertAtEnd(head2, 5);
    head2 = insertAtEnd(head2, 6);

    printf("List 1: ");
    printList(head1);

    printf("List 2: ");
    printList(head2);

    merged = MergeLL(head1, head2);

    printf("Merged List: ");
    printList(merged);

    freeList(merged);
    return 0;
}
```

```
List 1: 1 -> 2 -> 3 -> NULL
List 2: 4 -> 5 -> 6 -> NULL
Merged List: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> NULL
```

Question # 3

```c
#include <stdio.h>
#include "LinkedList.h"
#include <stdlib.h>

int *ConvertToArray(struct Node *head, int size){

    int *arr = (int *)malloc(size *sizeof(int));
    if(arr == NULL){
        fprintf(stderr, "Memory Allocation Failed");
        exit(1);
    }

    struct Node *current = head;
    int i=0;
    while(current != NULL){
        arr[i] = current->data;
        current = current->next;
        i ++;
    }
    return arr;
}
```

```c
int main(){
    struct Node *head = NULL;
    head = insertAtEnd(head, 1);
    head = insertAtEnd(head, 2);
    head = insertAtEnd(head, 3);

    struct Node *current = head;
    int size = 0;
    while(current != NULL){
        size++;
        current = current->next;
    }

    int *arr = ConvertToArray(head, size);
    printf("Elements: [");
    for(int i=0; i <size; i++){
        printf(" %d ", arr[i]);
    }
    printf("]");
}
```

```
Elements: [ 1  2  3 ]
```

Question # 4

```c
#include <stdio.h>
#include "LinkedList.h"
#include <stdlib.h>

struct Node *RemoveOddIndices(struct Node *head){

    int i=1;
    struct Node *prev = head;
    struct Node *current = head->next;
    while(current != NULL){
        if(i % 2 != 0){
            prev->next = current->next;
            free(current);
            current = current->next;
        }
        else{
            prev = current;
            current = current->next;
        }
        i++;
    }

    return head;
}
```

```c
int main(){
    struct Node *head = NULL;
    head = insertAtEnd(head, 1);
    head = insertAtEnd(head, 2);
    head = insertAtEnd(head, 3);
    head = insertAtBeginning(head, 6);
    head = insertAtBeginning(head, 5);
    head = insertAtBeginning(head, 4);

    printf("Original list:\n");
    printList(head);

    head = RemoveOddIndices(head);
    printf("\nList after removing odd indices:\n");
    printList(head);

    return 0;
}
```

```
Original list:
4 -> 5 -> 6 -> 1 -> 2 -> 3 -> NULL

List after removing odd indices:
4 -> 6 -> 2 -> NULL
```

Lab Session 5

Question # 1

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

int main(){
    FILE *fptr;
    fptr = fopen("Lab5.txt", "w");
    if(fptr != NULL){
        fprintf(fptr, "This is a test sentence written to a file using C programming.");
    }
    else{
        printf("Unable to open the file");
    }
    fclose(fptr);

    fptr = fopen("Lab5.txt", "a");
    if(fptr != NULL){
        fprintf(fptr, "\nThis line was added later to demonstrate file appending in C.");
    }
    else{
        printf("Unable to open the file");
    }
    fclose(fptr);

    fptr = fopen("Lab5.txt", "r");
    if(fptr != NULL){
        char myString[100];
        while(fgets(myString, 100, fptr)){
        printf("%s", myString);
        }
    }
    else{
        printf("Unable to open the file");
    }
    fclose(fptr);
}
```

```
This is a test sentence written to a file using C programming.
This line was added later to demonstrate file appending in C.
```

Question # 2

```c
#include <stdio.h>

int main(){
    FILE *fptr;
    int ch;
    int count = 0;
    fptr = fopen("Lab5.txt", "r");
    if (fptr != NULL){
        printf("Text from file:\n");
        while((ch = getc(fptr)) != EOF){
            printf("%c", ch);
            if (ch == ' ' || ch == '\n' || ch=='\t'){
                count++;
            }
        }
    }
    else{
        printf("Unable to open the file");
    }
    fclose(fptr);
    count++;

    printf("\nNumber of words: %d", count);

    return 0;
}
```

```
Text from file:
This is a test sentence written to a file using C programming.
This line was added later to demonstrate file appending in C.
Number of words: 23
```

Question # 3

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

void Add(){
    FILE *fptr = fopen("Student_Database.txt", "a+");
    if(fptr == NULL){

        printf("Unable to open the file");
        exit(1);
    }
    char name[30], dept[30], cgpa[5], roll[10];
    printf("\nEnter student's name: ");
    scanf(" %[^\n]",name);
    printf("\nEnter student's department: ");
    scanf(" %[^\n]",dept);
    printf("\nEnter student's roll number: ");
    scanf(" %[^\n]",roll);
    printf("\nEnter student's CPGA: ");
    scanf(" %[^\n]",cgpa);

    fprintf(fptr,"%-10s | %-25s | %-30s | %-5s\n", roll, name, dept, cgpa);

    printf("\nStudent record added successfully ✅\n");
    fclose(fptr);
}

void Modify(){
    FILE *fptr = fopen("Student_Database.txt", "r");
    FILE *temp = fopen("temp.txt", "w");
    if(fptr == NULL || temp == NULL){

        printf("Unable to open the file");
        exit(1);
    }
```

Question # 3

```
    char roll[10], searchRoll[10];
    char name[30], dept[30], cgpa[5];
    char line[200];
    int found = 0;

    printf("Enter roll number of student to modify: ");
    scanf(" %[^\n]",searchRoll);

    while(fgets(line, sizeof(line), fptr)){

        sscanf(line, "%s", roll);

        if(strcmp(roll, searchRoll) == 0){
            found = 1;
            printf("Enter new name: ");
            scanf(" %[^\n]",name);
            printf("Enter new department: ");
            scanf(" %[^\n]",dept);
            printf("Enter new CGPA: ");
            scanf(" %[^\n]",cgpa);

            fprintf(temp, "%-10s | %-25s | %-30s | %-5s\n", roll, name, dept, cgpa);
        }
        else{
            fputs(line, temp);
        }
    }
    fclose(fptr);
    fclose(temp);

    remove("Student_Database.txt");
    rename("temp.txt", "Student_Database.txt");

    if (found)
        printf("\nRecord modified successfully!\n");
    else
        printf("\nRecord not found!\n");
```

Question # 3

```c
void Delete(){
    FILE *fptr = fopen("Student_Database.txt", "r");
    FILE *temp = fopen("temp.txt", "w");

    if(fptr == NULL || temp == NULL){
        printf("Unable to open the file");
        exit(1);
    }
    char roll[10], name[30], dept[30], cgpa[5];
    char searchRoll[10];
    char line[200];
    int found = 0;

    printf("Enter the roll number of student to be delete: ");
    scanf(" %[^\n]",searchRoll);

    while(fgets(line, sizeof(line), fptr)){

        sscanf(line, "%s", roll);

        if(strcmp(searchRoll, roll) != 0){
            fputs(line, temp);
        }
        else{
            found = 1;
        }
    }
    fclose(fptr);
    fclose(temp);

    remove("Student_Database.txt");
    rename("temp.txt", "Student_Database.txt");

    if (found)
        printf("\nRecord deleted successfully!\n");
    else
        printf("\nRecord not found!\n");

}
```

Question # 3

```
    remove("Student_Database.txt");
    rename("temp.txt", "Student_Database.txt");

    if (found)
        printf("\nRecord deleted successfully!\n");
    else
        printf("\nRecord not found!\n");

}

void Display(){
    FILE *fptr = fopen("Student_Database.txt", "r");

    if(fptr == NULL){
        printf("Unable to open the file");
        exit(1);
    }

    char line[200];
    int found = 0;


    printf("\n%-10s | %-25s | %-30s | %-5s\n", "ROLL NO.", "NAME", "DEPARTMENT", "CGPA");
    printf("--------------------------------------------------------------------------------\n");
    while(fgets(line, sizeof(line), fptr)){
        printf("%s", line);
        }

    fclose(fptr);
}
int main(){

    FILE *fptr = fopen("Student_Database.txt", "a+");
    if(fptr == NULL){
        printf("Unable to open the file");
        exit(1);
    }
    fclose(fptr);

    int choice;
```

Lab Session 5

Question # 3

```c
while(1){
    printf("\n---Student's Database Menu----\n");
    printf("1. Add Student's Record\n");
    printf("2. Display Record\n");
    printf("3. Delete Student's Record\n");
    printf("4. Edit Student's Record\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    switch (choice){
        case 1:
            Add();
            break;

        case 2:
            Display();
            break;

        case 3:
            Delete();
            break;

        case 4:
            Modify();
            break;

        case 5:
            fclose(fptr);
            printf("Exiting...\n");
            return 0;

        default:
            printf("Invalid choice. Please try again.");
    }
  }
}
```

```
---Student's Database Menu----
1. Add Student's Record
2. Display Record
3. Delete Student's Record
4. Edit Student's Record
5. Exit
Enter your choice: 1

Enter student's name: Usman Rasheed Siddiqui

Enter student's department: CIS

Enter student's roll number: CS-24038

Enter student's CPGA: 3.98

Student record added successfully ✅
```

```
Enter your choice: 4
Enter roll number of student to modify: CS-24037
Enter new name: Abdur Rehman Abdul Hafeez
Enter new department: CIS
Enter new CGPA: 3.6

Record modified successfully!


Enter your choice: 2

ROLL NO.   | NAME                    | DEPARTMENT                    | CGPA
-------------------------------------------------------------------------------
CS-24037   | Abdur Rehman Abdul Hafeez | CIS                           | 3.6
CS-24038   | Usman Rasheed Siddiqui    | CIS                           | 3.98

Enter your choice: 3
Enter the roll number of student to be delete: CS-24037

Record deleted successfully!

Enter your choice: 2

ROLL NO.   | NAME                    | DEPARTMENT                    | CGPA
-------------------------------------------------------------------------------
CS-24038   | Usman Rasheed Siddiqui    | CIS                           | 3.98
```

Lab Session 5

Question # 3

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

int main(){
    int stop = 1;
    int element[100];
    int size = 0;
    int *stopptr = &stop;

    while(stop != 0){

        printf("\nEnter element for array: ");
        scanf("%d", element + size);
        size += 1;

        printf("Do you want to stop (0 to stop): ");
        scanf("%d", stopptr);
    }

    printf("\nElements of list: [ ");
    for(int i=0; i<size; i++){
        printf("%d ", *(element + i));
    }
    printf("]");

}
```

```
Enter element for array: 1
Do you want to stop (0 to stop): 1

Enter element for array: 2
Do you want to stop (0 to stop): 1

Enter element for array: 3
Do you want to stop (0 to stop): 1

Enter element for array: 4
Do you want to stop (0 to stop): 0

Elements of list: [ 1 2 3 4 ]
```

Question # 4

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

char search(int *element, int *list, int *size){

    for(int i=0; i< *size; i++){
        if (*(list + i) == *element){
            return 1;
        }

    }
    return 0;
}
```

```c
int main(){

    int element;
    int list[5] = {1, 2, 3, 4, 5};
    int size = sizeof(list)/sizeof(*(list+ 0));
    printf("Enter element to search in the list: ");
    scanf("%d", &element);

    int found = search(&element, list, &size);

    if(found == 1){
        printf("Element found in the list");
    }
    else{
        printf("Element not found");
    }
    return 0;
}
```

```
Enter element to search in the list: 2
Element found in the list

Enter element to search in the list: 6
Element not found
```