Assignment 4

U21CS089
Garvit Shah

1. There are two linked lists A and B containing the following data:

A: 3, 7,10,15,16,9,22,17,32

B: 16,2,9,13,47,8,10,1,28

WAP to create a linked list C that contains only those elements that are common in linked list A and B and also create a linked list D which contains all elements of A as well as B ensuring that there is no repetition of elements..

Code -

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

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

int count[100];

struct Node* create(struct Node* start){
    int n;
    struct Node* curr = start;
    printf("\nEnter the no. of elements - ");
    scanf("%d", &n);
    for(int i=0;i<n;i++){
        struct Node* temp;
        temp = (struct Node*)malloc(sizeof(struct Node));
        printf("\n- ");
        scanf("%d", &(temp->data));
        if(count[temp->data] == 0){
            count[temp->data] = 1;
        }
        else{
            count[temp->data] += 1;
        }
        temp->next = NULL;
        curr->next = temp;
        curr = temp;
    }
    return curr;
}
```

```c
void llcomm(struct Node* start1, struct Node* start2){
    int x=1, i=0;
    printf("\nCommon Elements are - ");
    while(x){
        if(count[i]>1){
            printf("%d | ", i);
        }
        else if(i==  100){
            x=0;
        }
        i++;
    }
}

void llunion(struct Node* start1, struct Node* start2){
    int x=1, i=0;
    printf("\nUnion of Elements is - ");
    while(x){
        if(count[i]>0){
            printf("%d | ", i);
        }
        else if(i==  100){
            x=0;
        }
        i++;
    }
}

void display(struct Node* start){
    struct Node* temp;
    temp = start;
    while(temp != NULL){
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
}

int main(){
    struct Node* start1;
    struct Node* curr1;
    struct Node* start2;
    struct Node *curr2;
    start1 = (struct Node*)malloc(sizeof(struct Node));
    start2 = (struct Node*)malloc(sizeof(struct Node));
    for(int i=0;i<100;i++){
        count[i] = 0;
    }
    printf("Enter the elements for linked lists : ");
    curr1 = create(start1);
    curr2 = create(start2);
    printf("\nFirst List: ");
```

```
    display(start1);
    printf("\nSecond List: ");
    display(start2);
    llcomm(start1, start2);
    llunion(start1, start2);
}
```

```
Enter the no. of elements - 9

- 3

- 7

- 10

- 15

- 16

- 9

- 22

- 17

- 32

Enter the no. of elements - 9

- 16

- 2

- 9

- 13

- 47

- 8

- 10

- 1

- 28

First List: 0 -> 3 -> 7 -> 10 -> 15 -> 16 -> 9 -> 22 -> 17 -> 32 ->
Second List: 0 -> 16 -> 2 -> 9 -> 13 -> 47 -> 8 -> 10 -> 1 -> 28 ->
Common Elements are - 9 | 10 | 16 |
Union of Elements is - 1 | 2 | 3 | 7 | 8 | 9 | 10 | 13 | 15 | 16 | 17 | 22 | 28 | 32 | 47 | ⏎
```

2. Split a linked list into two lists where each list contains alternate elements from it Given a linked list of integers, split it into two lists containing alternating elements from the original list.

For example, if the original list is {1, 2, 3, 4, 5}, then one sublist should be {1, 3, 5} and the other should be {2, 4}. The elements in the output lists may be in any order. i.e., the sublists can be {5, 3, 1} and {4, 2} for input list {1, 2, 3, 4, 5}.

Code –

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

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

struct Node* create(struct Node* start){
    int n;
    struct Node* curr = start;
    printf("\nEnter the no. of elements - ");
    scanf("%d", &n);
    for(int i=0;i<n;i++){
        struct Node* temp;
        temp = (struct Node*)malloc(sizeof(struct Node));
        printf("\n- ");
        scanf("%d", &(temp->data));
        temp->next = NULL;
        curr->next = temp;
        curr = temp;
    }
    return curr;
}

void split(struct Node* start0, struct Node* start1, struct Node*
start2){
    struct Node * temp;
    struct Node * curr1;
    struct Node* curr2;
    curr1 = start1;
    curr2 = start2;
    start0 = start0->next;
    for(int i =0;start0!=NULL;i++){
        temp = (struct Node *)malloc(sizeof(struct Node));
        temp->next = NULL;
        if(i%2 != 0){
            temp->data = start0->data;
            curr1->next = temp;
```

```
            curr1 = temp;
        }
        else{
            temp->data = start0->data;
            curr2->next = temp;
            curr2 = temp;
        }
        start0 = start0->next;
    }

}

void display(struct Node* start){
    struct Node* temp;
    temp = start->next;
    printf("\nThe Linked List: ");
    while(temp != NULL){
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
}

int main(){
    struct Node* start0;
    struct Node *curr0;
    struct Node* start1;
    struct Node* curr1;
    struct Node* start2;
    struct Node *curr2;
    start0 = (struct Node*)malloc(sizeof(struct Node));
    start1 = (struct Node*)malloc(sizeof(struct Node));
    start2 = (struct Node*)malloc(sizeof(struct Node));
    printf("Enter the elements for linked lists - ");
    curr0 = create(start0);
    display(start0);
    split(start0, start1, start2);
    display(start1);
    display(start2);
}
```

```
Enter the elements for linked lists -
Enter the no. of elements - 6

- 1

- 2

- 3

- 4

- 5

- 7

The Linked List: 1 -> 2 -> 3 -> 4 -> 5 -> 7 ->
The Linked List: 2 -> 4 -> 7 ->
The Linked List: 1 -> 3 -> 5 ->
```

3. Implemet Arithmetic expression solving equations using linklist. For Example: (ax^2 + bx + c)

Code -

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
struct Node
{
    int coeff;
    int power;
    struct Node *next;
};
typedef struct Node node;
node *createExp(node *ele, int n)
{
    int i, coeff, power;
    node *head = ele;
    for (i = 0; i < n; i++)
    {
        node *temp = (node *)malloc(sizeof(node));
        printf("Enter power of x for term %d: ", i + 1);
        scanf("%d", &power);
        printf("Enter coefficient of x for term %d: ", i + 1);
```

```
        scanf("%d", &coeff);
        temp->coeff = coeff;
        temp->power = power;
        temp->next = NULL;
        ele->next = temp;
        ele = temp;
    }
    return head;
}
void displayExp(node *curr)
{
    while (curr != NULL)
    {
        printf(" %dx^%d ", curr->coeff, curr->power);
        if (curr->next != NULL)
        {
            printf("+");
        }
        curr = curr->next;
    }
    printf("\n");
}
int result(node *ele, int x)
{
    int sum = 0, term_val;
    while (ele != NULL)
    {
        term_val = (ele->coeff) * (pow(x, ele->power));
        sum += term_val;
        ele = ele->next;
    }
    return sum;
}
int main()
{
    int res, x, n;
    node *head = (node *)malloc(sizeof(node));
    head->next = NULL;
    node *curr = head;
    printf("Enter no. of terms: ");
    scanf("%d", &n);
    head = createExp(curr, n);
    printf("\nThe expression is: ");
    displayExp(head->next);
    printf("Enter value of x: ");
    scanf("%d", &x);
    res = result(head->next, x);
    printf("\nThe result of the expression is %d", res);
    return 0;
}
```

```
Enter no. of terms: 5
Enter power of x for term 1: 7
Enter coefficient of x for term 1: 2
Enter power of x for term 2: 6
Enter coefficient of x for term 2: 1
Enter power of x for term 3: 5
Enter coefficient of x for term 3: 1
Enter power of x for term 4: 4
Enter coefficient of x for term 4: 1
Enter power of x for term 5: 3
Enter coefficient of x for term 5: -1

The expression is:  2x^7 + 1x^6 + 1x^5 + 1x^4 + -1x^3
Enter value of x: 1

The result of the expression is 4
```