1)

#include <stdlib.h>

#include <stdio.h>

struct Node

{

    struct Node \*prev;

    struct Node \*next;

    int data[4];

} \*head1, \*head2, \*temp;

struct Node \*createNode()

{

    struct Node \*newnode = (struct Node \*)malloc(sizeof(struct Node));

    newnode->prev = NULL;

    newnode->next = NULL;

    return newnode;

}

void insert(struct Node \*\*head, struct Node \*node)

{

    if (!(\*head))

    {

        \*head = node;

    }

    else

    {

        struct Node \*temp = \*head;

        while (temp->next != NULL)

        {

            temp = temp->next;

        }

        temp->next = node;

        node->next = NULL;

        node->prev = temp;

    }

}

struct Node \*reverse\_list()

{

    struct Node \*head = NULL;

    long long int n = 0;

    printf("Enter the data\n");

    scanf("%lld", &n);

    struct Node \*node = createNode();

    int count = 0;

    while (n > 0) // stores the num in reverse order in 4 digit blocks

    {

        if (!node)

        {

            printf("node not created\n");

        }

        node->data[count++] = n % 10;

        n = n / 10;

        if (count == 4)

        {

            insert(&head, node);

            count = 0;

            node = createNode();

        }

    }

    if (count != 4)

    {

        while (count <= 4)

        {

            if (!node)

                printf("not\n");

            node->data[count] = 0;

            if (count == 4)

                insert(&head, node);

            count++;

        }

    }

    else

    {

        free(node);

    }

    return head;

}

void display(struct Node \*head)

{

    if (!head)

        printf("head is null\n");

    temp = head;

    while (temp != NULL)

    {

        for (int i = 0; i < 4; i++)

        {

            printf("%d", temp->data[i]);

        }

        printf("->");

        temp = temp->next;

    }

}

struct Node \*addnumbers(struct Node \*head1, struct Node \*head2)

{

    if (!head1)

        printf("head1 not present\n");

    if (!head2)

        printf("head2 not present\n");

    int carry = 0;

    struct Node \*temp1, \*temp2, \*result, \*node;

    temp1 = head1;

    temp2 = head2;

    node = createNode();

    result = NULL;

    while ((temp1 != NULL) && (temp2 != NULL))

    {

        for (int i = 0; i < 4; i++)

        {

            int sum = temp1->data[i] + temp2->data[i] + carry;

            carry = sum / 10;

            sum = sum % 10;

            node->data[i] = sum;

            // printf("sum = %d\n",node->data[i]);

        }

        insert(&result, node);

        node = createNode();

        temp1 = temp1->next;

        temp2 = temp2->next;

    }

    if (!temp1)

    {

        temp = temp1;

        while (temp != NULL)

        {

            for (int i = 0; i < 4; i++)

            {

                int sum = temp->data[i] + carry;

                carry = sum / 10;

                sum = sum % 10;

                temp->data[i] = sum;

            }

            insert(&result, temp);

            temp = temp->next;

        }

    }

    if (!temp2)

    {

        temp = temp2;

        while (temp != NULL)

        {

            for (int i = 0; i < 4; i++)

            {

                int sum = temp->data[i] + carry;

                carry = sum / 10;

                sum = sum % 10;

                temp->data[i] = sum;

            }

            insert(&result, temp);

            temp = temp->next;

        }

    }

    if (carry)

    {

        node->data[0] = carry;

        node->data[1] = 0;

        node->data[2] = 0;

        node->data[3] = 0;

    }

    else

    {

        free(node);

    }

    return result;

}

void reverse(struct Node \*head)

{

    struct Node \*temp = head;

    while (temp->next != NULL)

        temp = temp->next;

    while (temp != NULL)

    {

        for (int i = 3; i >= 0; i--)

        {

            printf("%d", temp->data[i]);

        }

        printf("->");

        temp = temp->prev;

    }

}

int main()

{

    long long int n = 0;

    head1 = NULL;

    head2 = NULL;

    head1 = reverse\_list();

    head2 = reverse\_list();

    display(head1);

    printf("\n");

    display(head2);

    printf("\n");

    struct Node \*result = addnumbers(head1, head2);

    if (!result)

        printf("result not\n");

    // prints the reversed list in the correct order.

    display(result);

    printf("\n resultant = ");

    reverse(result);

}

Output:

