```
#include <stdio.h>
#include <stdlib.h>
struct node
{
    struct node *prev;
    int info;
    struct node *next;
};

struct node *addatbeg(struct node *start, int n)
{
    struct node *tmp;
    if ((tmp = (struct node *)malloc(sizeof(struct node))) == NULL)
    {
        printf("NO mem\n");
        exit(1);
    }
    tmp->info = n;
    tmp->next = start;
    tmp->prev = NULL;
    start = tmp;
    return start;
}

struct node *addatend(struct node *start, int n)
{
    struct node *ptr = start, *tmp;
    if (start == NULL)
        addatbeg(start, n);
    while (ptr->next != NULL)
        ptr = ptr->next;
    if ((tmp = (struct node *)malloc(sizeof(struct node))) == NULL)
    {
        printf("no mem\n");
        exit(1);
    }
    tmp->info = n;
    tmp->next = ptr->next;
    tmp->prev = ptr;
    ptr->next = tmp;
    return start;
}

struct node *toDigits(int n, struct node *start)
{
    start = NULL;
    start = addatbeg(start, n % 10);
    n /= 10;
    while (n > 0)
    {
        start = addatend(start, n % 10);
        n /= 10;
    }
    return start;
}

void display(struct node *start)
{
    if (start == NULL)
```

```
    {
        printf("The list is empty\n");
        return;
    }
    struct node *ptr = start;
    while (ptr != NULL)
    {
        printf("%3d", ptr->info);
        ptr = ptr->next;
    }
    printf("\n");
}

int numnodes(struct node *start)
{
    int count = 0;
    struct node *ptr = start;
    while (ptr != NULL)
    {
        count++;
        ptr = ptr->next;
    }
    return count;
}

void printrev(struct node *ptr)
{
    if (ptr == NULL)
    {
        printf("The list is empty lol\n");
        return;
    }
    while (ptr->next != NULL)
        ptr = ptr->next;
    while (ptr != NULL)
        printf("%d", ptr->info), ptr = ptr->prev;
    printf("\n");
}

struct node *add(struct node *num1, struct node *num2, struct node *result)
{
    // selecting the smaller list
    struct node *ptr1, *ptr2;
    if (numnodes(num1) > numnodes(num2))
        ptr1 = num1, ptr2 = num2;
    else if (numnodes(num2) > numnodes(num1))
        ptr1 = num2, ptr2 = num1;
    else
        ptr1 = num1, ptr2 = num2;
    int sum = 0;
    // making them the same size by adding zeroes
    while (numnodes(ptr1) != numnodes(ptr2))
        addatend(ptr2, 0);

    // finding sum for the first time
    sum = ptr1->info + ptr2->info;
    if (sum > 9)
    {
        result = addatbeg(result, sum % 10);
```

```
        sum /= 10;
        if (ptr1->next != NULL)
            ptr1->next->info += sum;
        else
        {
            result = addatend(result, sum);
            return;
        }
    }
    else
    {
        result = addatbeg(result, sum);
    }
    ptr1 = ptr1->next;
    ptr2 = ptr2->next;
    while (ptr1 != NULL)
    {
        sum = ptr1->info + ptr2->info;
        if (sum > 9)
        {
            result = addatend(result, sum % 10);
            sum /= 10;
            if (ptr1->next != NULL)
                ptr1->next->info += sum;
            else
            {
                result = addatend(result, sum);
                return;
            }
        }
        else
        {
            result = addatend(result, sum);
        }
        ptr1 = ptr1->next;
        ptr2 = ptr2->next;
    }
    return result;
}

int main(int argc, char const *argv[])
{
    struct node *start = NULL; // start pointer
    printf("Enter a number: ");
    int num;
    scanf("%d", &num);
    start = toDigits(num, start);
    display(start);
    struct node *num1, *num2; // number pointers
    int n1, n2;
    printf("Enter first number: ");
    scanf("%d", &n1);
    printf("Enter second number: ");
    scanf("%d", &n2);
    num1 = toDigits(n1, num1);
    num2 = toDigits(n2, num2);

    struct node *results = NULL;
    results = add(num1, num2, results);
```

```
    printf("The sum of the two numbers is: ");
    printrev(results);
    return 0;
}
```