**Q1:**

#include <stdio.h>

#include <stdlib.h>

struct node

{

    int data;

    struct node \*next;

    struct node \*prev;

};

struct node \*addatend(struct node \*start, int data)

{

    struct node \*tmp, \*p;

    tmp = (struct node \*)malloc(sizeof(struct node));

    p = start;

    while (p->next != NULL)

    {

        p = p->next;

    }

    tmp->data = data;

    tmp->next = NULL;

    tmp->prev = p;

    p->next = tmp;

    return start;

}

struct node \*addatbeg(struct node \*start, int data)

{

    struct node \*tmp = (struct node \*)malloc(sizeof(struct node));

    start = NULL;

    tmp->data = data;

    tmp->next = NULL;

    tmp->prev = NULL;

    start = tmp;

    return start;

}

struct node \*create(struct node \*start)

{

    int n, i, data;

    printf("\nEnter the number of nodes : ");

    scanf("%d", &n);

    struct node \*tmp, \*p;

    printf("\nEnter the data : ");

    scanf("%d", &data);

    start = addatbeg(start, data);

    p = start;

    for (i = 2; i <= n; i++)

    {

        printf("\nEnter the data : ");

        scanf("%d", &data);

        start = addatend(start, data);

    }

    return start;

}

void Display(struct node \*start)

{

    struct node \*p = start;

    while (p != NULL)

    {

        printf("\n%d ", p->data);

        p = p->next;

    }

}

void count(struct node \*start)

{

    struct node \*p = NULL;

    int cnt = 0;

    p = start;

    while (p != NULL)

    {

        cnt++;

        p = p->next;

    }

    printf("\nThe total number of nodes are : %d", cnt);

}

struct node \*swap(struct node \*start)

{

    struct node \*p1 = NULL, \*p2 = NULL, \*p3 = NULL;

    p1 = start;

    if (p1 == NULL)

    {

        printf("\nList is Empty");

        return start;

    }

    p2 = p1->next;

    start = p2;

    while (p2 != NULL)

    {

        p1->next = p2->next;

        if (p2->next != NULL)

            p2->next->prev = p1;

        p2->next = p1;

        p2->prev = p3;

        if (p3 != NULL)

            p3->next = p2;

        p1->prev = p2;

        p3 = p1;

        p1 = p1->next;

        if (p1 == NULL)

            break;

        p2 = p1->next;

    }

    Display(start);

    return start;

}

int main()

{

    int choice, pos, data;

    struct node \*start = NULL;

    while (1)

    {

        printf("\n1.Create");

        printf("\n2.Count");

        printf("\n3.Display");

        printf("\n4.Swap the adjacent elements");

        printf("\n5.Exit");

        scanf("%d", &choice);

        switch (choice)

        {

        case 1:

            start = create(start);

            break;

        case 3:

            Display(start);

            break;

        case 2:

            count(start);

            break;

        case 4:

            start = swap(start);

            break;

        case 5:

            exit(1);

        }

    }

    return 0;

}

**OUPUT:**

1.Create

2.Count

3.Display

4.Swap the adjacent elements

5.Exit1

Enter the number of nodes : 5

Enter the data : 1

Enter the data : 2

Enter the data : 3

Enter the data : 4

Enter the data : 5

1.Create

2.Count

3.Display

4.Swap the adjacent elements

5.Exit2

The total number of nodes are : 5

1.Create

2.Count

3.Display

4.Swap the adjacent elements

5.Exit3

1

2

3

4

5

1.Create

2.Count

3.Display

4.Swap the adjacent elements

5.Exit4

2

1

4

3

5

1.Create

2.Count

3.Display

4.Swap the adjacent elements

5.Exit5

**Q2:**

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

struct node

{

    int data;

    struct node \*next;

    struct node \*prev;

};

void Display(struct node \*start);

struct node \*addatend(struct node \*start, int data)

{

    struct node \*tmp, \*p;

    tmp = (struct node \*)malloc(sizeof(struct node));

    p = start;

    while (p->next != NULL)

    {

        p = p->next;

    }

    tmp->data = data;

    tmp->next = NULL;

    tmp->prev = p;

    p->next = tmp;

    return start;

}

struct node \*addatbeg(struct node \*start, int data)

{

    struct node \*tmp = (struct node \*)malloc(sizeof(struct node));

    start = NULL;

    tmp->data = data;

    tmp->next = NULL;

    tmp->prev = NULL;

    start = tmp;

    return start;

}

struct node\* create(struct node \*start,int num)

{

    int digit=0;

    while(num!=0){

        digit=num%10;

        num=num/10;

        if(start==NULL){

            start=addatbeg(start,digit);

        }else{

            start=addatend(start,digit);

        }

    }

    Display(start);

    return start;

    }

void Display(struct node \*start)

{

    struct node \*p = start;

    printf("\nStack : ");

    while (p != NULL)

    {

        printf("\n%d ", p->data);

        p = p->next;

    }

    printf("\n");

}

int count(struct node \*start)

{

    struct node \*p = NULL;

    int cnt = 0;

    p = start;

    while (p != NULL)

    {

        cnt++;

        p = p->next;

    }

    return cnt;

}

void add(struct node \*start1,struct node \*start2)

{

    int a=0,b=0,n1,n2;

    struct node \*p1=start1,\*p2=start2;

    n1=count(start1);

    n2=count(start2);

    for(int i=0;i<n1&&p1!=NULL;i++)

    {

        a+=(p1->data)\*(pow(10,i));

        p1=p1->next;

    }

    for(int i=0;i<n2&&p2!=NULL;i++)

    {

        b+=(p2->data)\*(pow(10,i));

        p2=p2->next;

    }

    printf("\nThe Sum Of The Two Numbers is : %d",a+b);

}

int main()

{

    int choice, pos, data, num1, num2;

    struct node \*start1 = NULL,\*start2=NULL,\*start=NULL;

            printf("\nEnter the first number : ");

            scanf("%d", &num1);

            start1=create(start1,num1);

            printf("\nEnter the second number : ");

            scanf("%d", &num2);

            start2=create(start2,num2);

            add(start1,start2);

    return 0;

}

**OUTPUT:**

Enter the first number : 123

Stack :

3

2

1

Enter the second number : 234

Stack :

4

3

2

The Sum Of The Two Numbers is : 357

**Q3:**

#include<stdio.h>

#include<malloc.h>

struct NODE

{

    int info;

    struct NODE \*next;

};

typedef struct NODE node;

node\*altdel(node \*,int);

node \* createlist(node\*,int);

void display(node\*);

node\*createlist(node \*last,int size)

{

    if(size>0)

    {

        int data;

        node \*tmp;

        printf("Enter data for the node :");

        scanf("%d",&data);

        tmp=(node\*)malloc(sizeof(node));

        tmp->info=data;

        tmp->next=tmp;

        last=tmp;

        for(int i=1;i<size;i++)

        {

            printf("Enter data for the node :");

            scanf("%d",&data);

            tmp=(node\*)malloc(sizeof(node));

            tmp->info=data;

            tmp->next=last->next;

            last->next=tmp;

            last=tmp;

        }

        return last;

    }

}

void display(node\*last)

{

    if(last==NULL)

    {

        printf("The list is empty...\n");

        return;

    }

    node \*tmp=last->next;

    do

    {

        printf("%d ",tmp->info);

        tmp=tmp->next;

    }while(tmp!=last->next);

}

node \*altdel(node\*last,int n)

{

    if(last==NULL)

    printf("The list is empty...\n");

    if(last->next==last)

    return last;

    else

    {

        node \*tmp=last->next;

        if(n&1)

        {

        do

        {

            node \*q=tmp->next;

            tmp->next=q->next;

            free(q);

            tmp=tmp->next;

        }while(tmp!=last);

        }

        else

        {

            do

        {

            node \*q=tmp->next;

            tmp->next=q->next;

            free(q);

            tmp=tmp->next;

        }while(tmp->next!=last);

            tmp->next=last->next;

            free(last);

            return tmp;

        }

    }

}

int main()

{

    int n;

    node \*last=NULL;

    printf("Enter the number of nodes : ");

    scanf("%d",&n);

    last=createlist(last,n);

    printf("the original list is : ");

    display(last);

    last=altdel(last,n);

    printf("\nthe list after deletion is : ");

    display(last);

    return 0;

}

**OUTPUT:**

Enter the number of nodes : 3

Enter data for the node :1

Enter data for the node :2

Enter data for the node :3

the original list is : 1 2 3

the list after deletion is : 1 3