**Collage : Vishwakarma Institute of Technology**

**Course Name : Data Structure in C**

**Name : Vedika Vikas Sontakke**

**Roll no : 37**

**PRN NO 12220206**

Assignment No 2 : Implement following Assignments Based on Linked list..

1. Create Single Linked list and implement insert , delete , display operation.

Program :

#include <stdio.h>

#include<stdlib.h>

struct node

{

    int data;

    struct node \*next;

};

struct node \*head , \*temp = NULL;

void display();

void delete\_specific\_node()

{

    struct node\* prev = head;

    struct node\* current = head;

    int y;

    printf("enter the element which you want to delete :\n");

    scanf("%d" , &y);

    while(current->data != y)

    {

       prev = current;

       current = current->next;

    }

    prev->next = current->next;

    current->next = NULL;

    free(current);

    printf("%d element deleted successfully \n" , y);

    display();

}

void delete\_last\_node()

{

    struct node\* prev = head;

    struct node\* current = head;

    while(current->next != NULL){

       prev = current;

       current = current->next;

    }

    prev->next = NULL;

    free(current);

    printf("last element deleted successfully \n");

    display();

    delete\_specific\_node();

}

void delete\_first\_node()

{

   struct node\* first = head;

   head = head->next;

   free(first);

   printf("first element deleted successfully \n");

   display();

   delete\_last\_node();

}

void insert\_before\_specific\_node()

{

    struct node\* current = head;

    struct node\* prev = head;

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

   int y;

   printf("enter element before that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the ellement which you want to insert :\n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   while(current->data != y){

       prev = current;

      current = current->next;

   }

   newnode->next = current;

   prev->next = newnode;

   display();

   delete\_first\_node();

}

void insert\_after\_specific\_node()

{

   struct node\* current = head;

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

   int y;

   printf("enter element after that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the ellement which you want to insert :\n");

      scanf("%d" , &newnode->data);

   newnode->next = NULL;

   while(current->data != y)

      current = current->next;

   newnode->next = current->next;

   current->next = newnode;

   display();

   insert\_before\_specific\_node();

}

void insert\_end()

{

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

   printf("enter element which you want to insert at end : \n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   struct node \*current\_ = head;

   while( current\_->next != NULL )

        current\_ = current\_->next;

   current\_->next = newnode;

   display();

   insert\_after\_specific\_node();

}

void addnode(int x)

{

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

    newnode->data = x;

    newnode->next = NULL;

    if(head == NULL)

    {

        head = newnode;

        temp = newnode;

    }else

    {

        temp->next = newnode;

        temp = newnode;

    }

}

void insert\_begin()

{

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

   printf("enter element which you want to insert at begin : \n");

   scanf("%d" , &newnode->data);

    newnode->next = head;

    head = newnode;

   display();

   insert\_end();

}

void display()

{

    struct node \*current = head;

    printf("elements in the linked list are :");

     while(current != NULL)

     {

        printf("%d ", current->data);

        current = current->next;

     }

     printf("\n");

}

void creation()

{

    int size;

    printf("enter the size of linked list :");

    scanf("%d",&size);

    printf("enter elements of linked list :\n");

    for(int i=0 ; i<size ; i++)

    {

       int n;

       scanf("%d",&n);

       addnode(n);

    }

    display();

    insert\_begin();

}

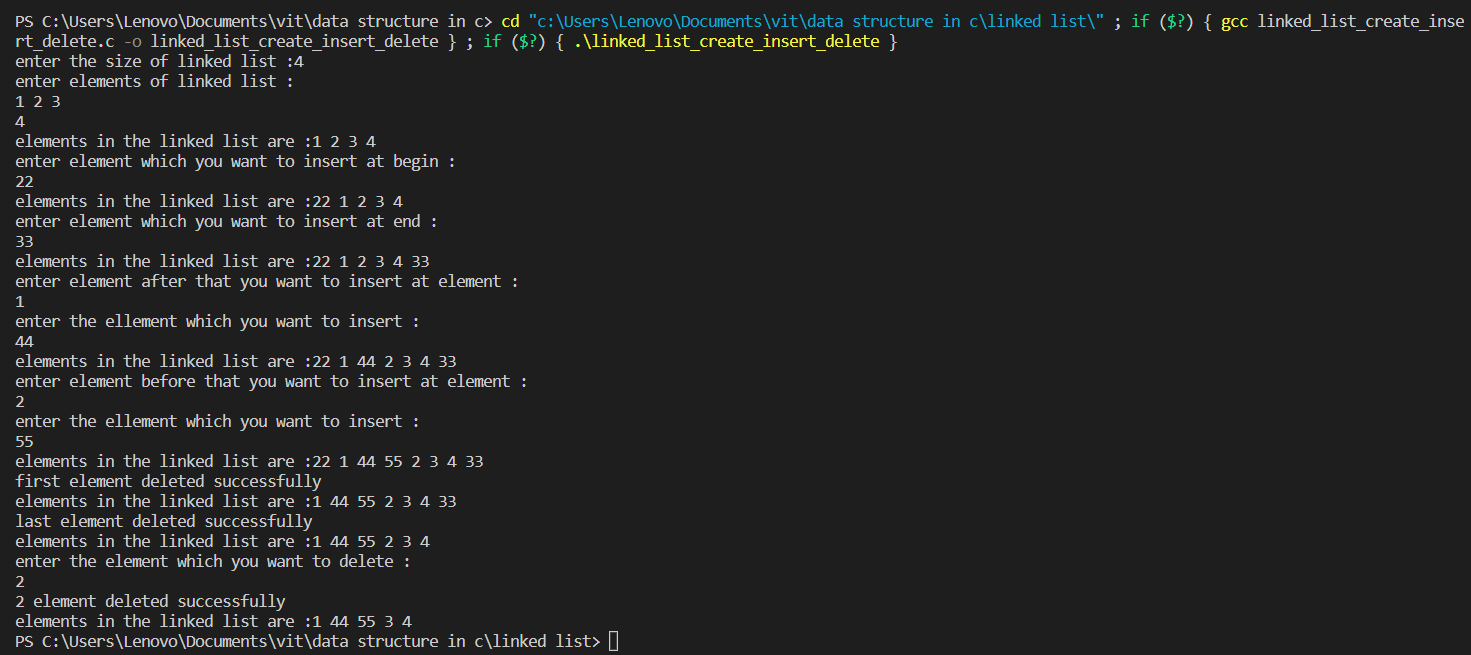
int main()

{

    creation();

}

Output :



1. Create Doubley linked list and implement insert , delete , display operation.

Program :

#include<stdio.h>

#include<stdlib.h>

struct node

{

    struct node \*prev;

    int data;

    struct node \*next;

};

struct node \*head , \*temp = NULL;

void display()

{

   struct node \*temp = head;

    printf("elements in the linked list are :");

    while(temp != NULL){

        printf("%d " ,temp->data);

        temp = temp->next;

    }

    printf("\n");

}

void delete\_speciefic\_node(){

     int y ;

     struct node \*temp = head;

     printf("enter element which you want delete \n");

     scanf("%d",&y);

    while(temp->data != y)

       temp = temp->next;

    temp->prev->next = temp->next;

    temp->next->prev = temp->prev;

    display();

}

void delete\_last\_node()

{

    struct node \*prev = head;

    struct node \*first = head;

    while(first->next != NULL){

        prev = first;

        first = first->next;

    }

    prev->next = NULL;

    free(first);

    printf("last element deleted successfully \n");

    display();

}

void delete\_first\_node()

{

    struct node \*first = head;

    head = head->next;

    free(first);

    printf("first element deleted successfully \n");

    display();

}

void insert\_element\_before\_specefic\_node()

{

   struct node \*temp = head;

   struct node \*newnode\_ = (struct node \*)malloc(sizeof(struct node \*));

   int y;

   printf("enter element before that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the element which you want to insert :\n");

   scanf("%d" , &newnode\_->data);

   while(temp->data != y)

        temp = temp->next;

    newnode\_->prev = temp->prev;

    temp->prev->next = newnode\_;

    temp->prev = newnode\_;

    newnode\_->next = temp;

   display();

}

void insert\_after\_specific\_node()

{

    struct node \*temp = head;

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

   int y;

   printf("enter element after that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the ellement which you want to insert :\n");

   scanf("%d" , &newnode->data);

    while(temp->data != y)

      temp = temp->next;

     temp->next->prev = newnode;

     newnode->next = temp->next;

     temp->next = newnode;

     newnode->prev = temp;

   display();

}

void insert\_end()

{

    struct node \*temp = head;

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

   printf("enter element which you want to insert at end : \n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   while(temp->next != NULL)

     temp = temp->next;

   temp->next = newnode;

   newnode->prev = temp;

   display();

}

void insert\_begin()

{

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

    printf("enter element which you want to insert at begin : \n");

    scanf("%d" , &newnode->data);

    newnode->next = head;

    head = newnode;

    display();

}

void addnode(int x)

{

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

    newnode->data = x;

    newnode->prev = NULL;

    newnode->next = NULL;

   if(head == NULL)

   {

     head = newnode;

     temp = newnode;

   }else

   {

     temp->next = newnode;

     newnode->prev = temp;

     temp = newnode;

   }

}

void creation()

{

    int size;

    printf("enter the size of linked list :");

    scanf("%d",&size);

    printf("enter elements of linked list :\n");

    for(int i=0 ; i<size ; i++)

    {

       int n;

       scanf("%d",&n);

       addnode(n);

    }

    display();

}

int main()

{

    creation();

    insert\_begin();

    insert\_end();

  //insert\_after\_specific\_node();

  //insert\_element\_before\_specefic\_node();

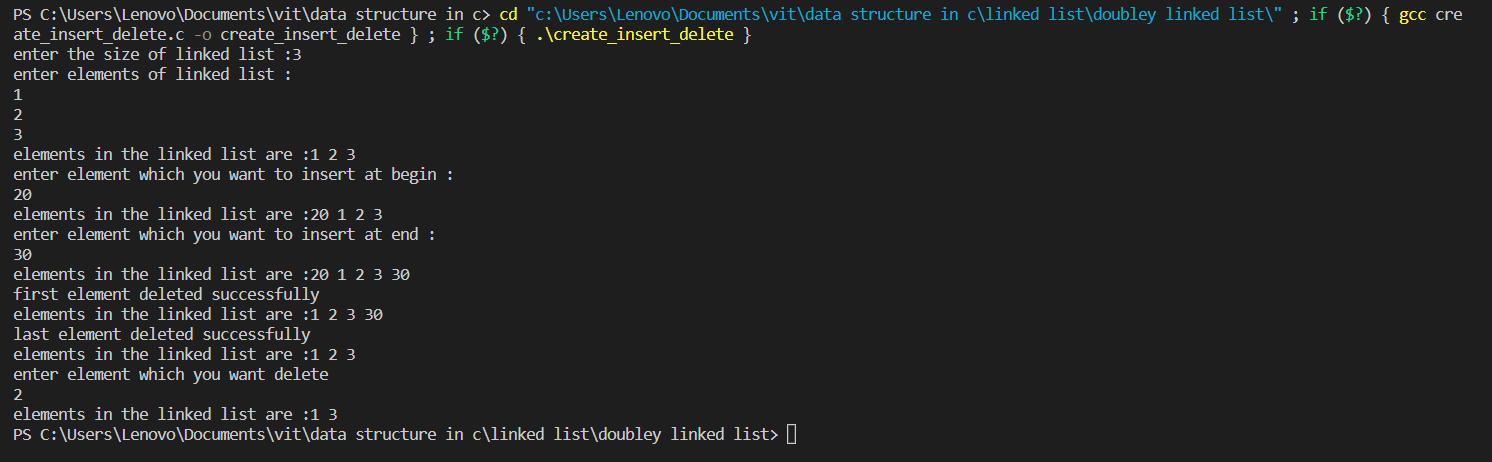
    delete\_first\_node();

    delete\_last\_node();

    delete\_speciefic\_node();

}

Output :



1. Create Circular linked list and implement insert , delete , display operation.

Program :

#include<stdio.h>

#include<stdlib.h>

struct node

{

    int data;

    struct node \*next;

};

struct node \*head , \*temp = NULL;

void display()

{

    struct node \*temp = head;

    printf("elements in the linked list are :");

    while(temp->next != head)

    {

        printf("%d ",temp->data);

        temp = temp->next;

    }

    printf("%d ",temp->data);

    printf("\n");

}

void delete\_first\_node()

{

    struct node \*temp = head;

    while(temp->next != head)

      temp = temp->next;

    head = head->next;

    temp->next = head;

    printf("first element deleted successfully \n");

    display();

}

void delete\_last\_node()

{

    struct node \*temp = head;

    struct node \*prev = head;

    while(temp->next != head){

        prev = temp;

        temp = temp->next;

    }

   prev->next = head;

   printf("last node deleted successfully \n");

   display();

}

void delete\_specefic\_node()

{

   struct node\* prev = head;

    struct node\* current = head;

    int y;

    printf("enter the element which you want to delete :\n");

    scanf("%d" , &y);

    while(current->data != y)

    {

       prev = current;

       current = current->next;

    }

    prev->next = current->next;

    current->next = NULL;

    free(current);

    printf("%d element deleted successfully \n" , y);

    display();

}

void insert\_begin()

{

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

   printf("enter element which you want to insert at begin : \n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   struct node \*temp = head;

   newnode->next = head;

   while(temp->next != head)

     temp = temp->next;

   temp->next = newnode;

   head = newnode;

   display();

}

void insert\_end()

{

   struct node \*temp = head;

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

   printf("enter element which you want to insert at end : \n");

   scanf("%d" , &newnode->data);

   while(temp->next != head)

      temp = temp->next;

    temp->next = newnode;

    newnode->next = head;

    display();

}

void insert\_before\_specific\_node()

{

    struct node\* temp = head;

    struct node\* prev = head;

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

   int y;

   printf("enter element before that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the ellement which you want to insert :\n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   while(temp->data != y){

       prev = temp;

      temp = temp->next;

   }

   newnode->next = temp;

   prev->next = newnode;

   display();

}

void insert\_after\_specific\_node()

{

   struct node\* temp = head;

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

   int y;

   printf("enter element after that you want to insert at element : \n");

   scanf("%d" , &y);

   printf("enter the ellement which you want to insert :\n");

   scanf("%d" , &newnode->data);

   newnode->next = NULL;

   while(temp->data != y)

      temp = temp->next;

   newnode->next = temp->next;

   temp->next = newnode;

   display();

}

void addnode(int n)

{

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

   newnode->data = n;

   newnode->next = NULL;

   if(head == NULL)

   {

      head = newnode;

      temp = newnode;

      newnode->next = head;

   }else

   {

     newnode->next = head;

     temp->next = newnode;

     temp = newnode;

   }

}

int main()

{

    int size;

   printf("enter the size of list :");

   scanf("%d",&size);

   printf("enter the elements in the list \n");

   for(int i=0 ; i<size ; i++)

   {

      int n;

      scanf("%d",&n);

      addnode(n);

   }

   display();

   insert\_begin();

   insert\_end();

   insert\_after\_specific\_node();

   insert\_before\_specific\_node();

   delete\_first\_node();

   delete\_last\_node();

  delete\_specefic\_node();

}

Output :

