ALL OPERATIONS IN LINKED LIST

#include<stdio.h>

#include<stdlib.h>

Typedef struct node

{

Int data;

Struct node \*next;

}node;

Node \*Createlist(int);

Void display(node \*start);

Node \*insert\_beg(node \*start);

Node \*insert\_end(node \*start);

Node \*insert\_before(node \*start);

Node \*insert\_after(node \*start);

Node \*delete\_beg(node \*start);

Node \*delete\_end(node \*start);

Node \*delete\_node(node \*start);

Int main()

{

Int n,ch;

Node \*start=NULL;

Do

{

Printf(“\n\n\*\*\*MAIN MENU\*\*\*”);

Printf(“\n 1:Create a list”);

Printf(“\n 2:Display the list”);

Printf(“\n 3:Add a node at the beginning”);

Printf(“\n 4:Add a node at the end”);

Printf(“\n 5:Add a node before a given node”);

Printf(“\n 6:Add a node after a given node”);

Printf(“\n 7:Delete a node from the beginning”);

Printf(“\n 8:Delete a node from the end”);

Printf(“\n 9:Delete a given node”);

Printf(“\n 10: Exit”);

Printf(“\n\n Enter your option:”);

Scanf(“%d”,&ch);

Switch(ch)

{

Case 1:

Printf(“How many nodes:”);

Scanf(“%d”,&n);

Start=Createlist(n);

Break;

Case 2:

Display(start);

Break;

Case 3:

Start=insert\_beg(start);

Break;

Case 4:

Start=insert\_end(start);

Break;

Case 5:

Start=insert\_before(start);

Break;

Case 6:

Start=insert\_after(start);

Break;

Case 7:

Start=delete\_beg(start);

Break;

Case 8:

Start=delete\_end(start);

Break;

Case 9:

Start=delete\_node(start);

Break;

}

}while(ch!=10);

Return 0;

}

Node \*Createlist(int n)

{

Int I;

Node \*start=NULL;

Node \*newnode=NULL;

Node \*ptr=NULL;

For(i=0;i<n;i++)

{

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

Printf(“Enter the data for node %d:”,i+1);

Scanf(“%d”,&newnode->data);

Newnode->next=NULL;

If(start==NULL)

{

Start=newnode;

}

Else

{

Ptr=start;

While(ptr->next!=NULL)

{

Ptr=ptr->next;

}

Ptr->next=newnode;

}

}

Return start;

}

Void display(node \*start)

{

Node \*ptr=start;

While(ptr!=NULL)

{

Printf(“\t %d🡪”,ptr->data);

Ptr=ptr->next;

}

}

Node \*insert\_beg(node \*start)

{

Int item;

Node \*new\_node;

Printf(“Enter the data:”);

Scanf(“%d”,&item);

New\_node=(node \*)malloc(sizeof(node));

New\_node->data=item;

New\_node->next=start;

Start=new\_node;

Return start;

}

Node \*insert\_end(node \*start)

{

Int item;

Node \*new\_node;

Node \*ptr;

Printf(“Enter the data:”);

Scanf(“%d”,&item);

New\_node=(node \*)malloc(sizeof(node));

New\_node->data=item;

Ptr=start;

While(ptr->next!=NULL)

{

Ptr=ptr->next;

}

Ptr->next=new\_node;

New\_node->next=NULL;

Return start;

}

Node \*insert\_before(node \*start)

{

Node \*new\_node;

Node \*ptr;

Node \*preptr;

Int item,val;

Printf(“Enter the data:”);

Scanf(“%d”,&item);

Printf(“\n Enter the value before which the data has to be inserted:”);

Scanf(“%d”,&val);

New\_node=(node \*)malloc(sizeof(node));

New\_node->data=item;

Ptr=start;

Preptr=ptr;

While(ptr->data!=val)

{

Preptr=ptr;

Ptr=ptr->next;

}

Preptr->next=new\_node;

New\_node->next=ptr;

Return start;

}

Node \*insert\_after(node \*start)

{

Node \*new\_node;

Node \*ptr;

Node \*preptr;

Int item,val;

Printf(“Enter the data:”);

Scanf(“%d”,&item);

Printf(“\n Enter the value after which the data has to be inserted:”);

Scanf(“%d”,&val);

New\_node=(node \*)malloc(sizeof(node));

New\_node->data=item;

Ptr=start;

Preptr=ptr;

While(preptr->data!=val)

{

Preptr=ptr;

Ptr=ptr->next;

}

Preptr->next=new\_node;

New\_node->next=ptr;

Return start;

}

Node \*delete\_beg(node \*start)

{

Node \*ptr;

Ptr=start;

Start=start->next;

Free(ptr);

Return start;

}

Node \*delete\_end(node \*start)

{

Node \*ptr,\*preptr;

Ptr=start;

Preptr=ptr;

While(ptr->next!=NULL)

{

Preptr=ptr;

Ptr=ptr->next;

}

Preptr->next=NULL;

Free(ptr);

Return start;

}

Node \*delete\_node(node \*start)

{

Node \*ptr,\*preptr,\*temp;

Int val;

Printf(“\n Enter the value after which the data has to be deleted:”);

Scanf(“%d”,&val);

Ptr=start;

Preptr=ptr;

While(preptr->data!=val)

{

Preptr=ptr;

Ptr=ptr->next;

}

Temp=ptr;

Preptr->next=ptr->next;

Free(temp);

Return start;

}