7. Implementation of singly linked list

**PROGRAM CODE**

#include<stdio.h>

#include<stdlib.h>

struct str

{

int data;

struct str\* next;

};

void main()

{

typedef struct str node;

int ch,pos,item;

node \*temp,\*start=NULL,\*p;

while(1)

{

printf("\nMENU\n1.Insert at beginning\n2.Insert at end\n3.Insert at any position");

printf("\n4.Delete from beginning\n5.Delete from a particular position\n6.Delete from the end");

printf("\n7.Display elements\n8.Exit");

printf("\nEnter your choice: ");

scanf("%d",&ch);

switch(ch)

{

case 1: temp=(node \*)malloc(sizeof(node));

printf("Enter the value: ");

scanf("%d",&item);

temp->data=item;

if(start==NULL)

{

temp->next=NULL;

start=temp;

}

else

{

temp->next=start;

start=temp;

}

break;

case 2: temp = (node \*)malloc(sizeof(node));

printf("Enter the value: ");

scanf("%d",&item);

if(start==NULL)

{

temp->next = NULL;

start = temp;

}

else

{

temp->data=item;

p=start;

while(p->next!=NULL)

p=p->next;

temp->next=NULL;

p->next=temp;

}

break;

case 3: temp = (node \*)malloc(sizeof(node));

printf("Enter position you want to be inserted at: ");

scanf("%d",&pos);

p = start;

for (int i = 1; i < pos-1; i++)

p = p->next;

printf("Enter the value: ");

scanf("%d",&item);

temp->data=item;

temp->next=p->next;

p->next=temp;

break;

case 4: if (start == NULL)

printf("\nDeletion is not possible");

else if (start->next == NULL)

{

temp = start;

start = NULL;

printf("\nDeleted item is %d", temp->data);

free(temp);

}

else

{

temp = start;

start = start->next;

printf("\nDeleted item is %d", temp->data);

free(temp);

}

break;

case 5: printf("\nEnter the position: ");

scanf("%d", &pos);

temp = start;

for (int i = 1; i < pos - 1; i++)

temp = temp->next;

p = temp->next;

temp->next = p->next;

printf("\nDeleted item is %d", p->data);

free(p);

break;

case 6: if (start == NULL)

printf("\nDeletion is not possible");

else if (start->next == NULL)

{

temp = start;

start = NULL;

printf("\nDeleted item is %d", temp->data);

free(temp);

}

else

{

temp = start;

p = start->next;

while (p->next != NULL)

{

p = p->next;

temp = temp->next;

}

temp->next = NULL;

printf("\nDeleted item is %d", p->data);

free(p);

}

break;

case 7: p=start;

printf("\nDisplaying\n");

while (p != NULL)

{

printf("%d ", p->data);

p = p->next;

}

break;

case 8:exit(0);

break;

default:printf("\nInvalid Position");

}

}

}

**OUTPUT**







