|  |  |  |  |
| --- | --- | --- | --- |
| **Serial No.** | **Aim of the Program** | **Date of Expt.** | **Teacher’s Sign.** |
| **1.** | TO PERFORM LINEAR AND BINARY SEARCH ON AN ARRAY OF NUMBERS. |  |  |
| **2.** | TO REPRESENT A MATRIX IN SPARSE FORM. |  |  |
| **3.** | TO CREATE A LINKED LIST HAVING INFORMATION ABOUT A STUDENT AND  PERFORM INSERTION, DELETION AND REVERSAL. |  |  |
| **4.** | TO CREATE A DOUBLY LINKED LIST WITH EMPLOYEE INFORMATION AND  PERFORM INSERTION AT FRONT, AND DELETION AT BACK. |  |  |
| **5.** | TO CREATE A CIRCULAR LINKED LIST WITH INFO ABOUT ACLASS AND  PERFORM INSERTION AND DELETION AT THE ENDS. |  |  |
| **6.** | TO CREATE A STACK AND PERFORM PUSH, POP AND TRAVERSE OPERATIONS  USING A LINEAR LINKED LIST. |  |  |
| **7.** | TO CREATE A LINEAR QUEUE USING A LINEAR LINKED LIST AND PERFORM  OPERATIONS LIKE INSERTION, DELETION and TRAVERSAL. |  |  |

**INDEX**

**AIM – TO PERFORM LINEAR AND BINARY SEARCH ON AN ARRAY OF NUMBERS**

<Source Listing for **SEARCH.H**>

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

int linear\_search(int \*array, int length, int find)

{

int i = 0;

while( i < length)

{

if(array[i] == find)

return i;

i++;

}

return -1;

}

int binary\_search(int \*array, int min, int max, int find)

{

if(min > max)

{

return -1;

}

int mid = (min + max) / 2;

if(array[mid] == find)

{

return mid;

}

else if( array[mid] < find)

{

return binary\_search(array, mid + 1, max, find);

}

else if(array[mid] > find)

{

return binary\_search(array, min, mid - 1, find);

}

return -1;

}

void DipslayArr(int \*array,int pos)

{

int i=0,a,j=0,k=0;

char buff[20];

while(i<10)

{

if(j<400)

{

a=array[i];

//DATA

sprintf(buff,"%d", a);

outtextxy(j+45,40+k,buff);

sprintf(buff,"%d", i);

outtextxy(j+95,60+k,buff);

//DATA HEADING

outtextxy(j+35,k+20, "VALUE");

outtextxy(j+35,k+60, "INDEX");

//boxes

if(i==pos)

{

setcolor(9);

delay(600);

}

rectangle(j+30,k+10,j+120,k+50);

rectangle(j+30,k+50,j+120,k+70);

setcolor(15);

delay(500);

i++;

j+=100;

}

else

{

k+=100;

j=0;

}

}

getch();

closegraph();

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include<graphics.h>

#include <conio.h>

#include "**SEARCH.h**"

int main()

{

int i=0,again = 1, num, search, pos, a[10];

clrscr();

printf("\nEnter an array of 10 integers:\n");

while( i < 10 )

{

scanf("%d", &a[i]);

i++;

}

while(again)

{

printf("\nEnter a number to search for: ");

scanf("%d", &num);

printf("\nLinear(1) or Binary(2) search? : ");

scanf("%d", &search);

switch(search)

{

case 1:

clrscr();

pos = linear\_search(a, 10, num);

if(pos == -1)

printf("\nElement not present in array!!!");

else

{initialize();

DipslayArr(a,pos);}

break;

case 2:

clrscr();

pos = binary\_search(a, 0, 10, num);

if(pos == -1)

printf("\nElement not present in array!!!");

else

{initialize();

DipslayArr(a,pos);}

break;

default:

printf("\nPlease enter enter a valid option!!! 1 or 2!!!");

}

printf("\nSearch again? Y(1), N(0) : ");

scanf("%d", &again);

}

return 0;

}

**AIM – TO REPRESENT A MATRIX IN SPARSE FORM**

<Source Listing for **SPARSE.H**>

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

setcolor(15);

outtextxy(200,10,"SPARSE MATRIX is a Kind of Matrix whose");

outtextxy(200,30,"more than 50% elements are 0");

outtextxy(200,50,"In this program you can ");

outtextxy(200,70," Enter Maximum of 14 Non-Zero elements ");

setcolor(4);

outtextxy(200,100,"First row :-");

outtextxy(200,120,"First Column :No. of rows");

outtextxy(200,140,"Second Column :No. of Columns");

outtextxy(200,160,"Third Column :No. of non-Zero Elements");

setcolor(2);

outtextxy(200,280,"Second row Onwards :-");

outtextxy(200,300,"First Column :rows Index of Non-Zero Element");

outtextxy(200,320,"Second Column :Columns Index of Non-Zero Element");

outtextxy(200,340,"Third Column : The Non-Zero Element ");

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include <conio.h>

#include <graphics.h>

#include "**SPARSE.h**"

int main()

{

int i,j,mat[5][5],sparse[3][15],count = 0, k =1,element;

clrscr();

printf(YELLOW "\nEnter a 5 x 5 matrix rowwise:\n");

for(i = 0; i < 5; i++)

{

for(j = 0; j < 5; j++)

{

element = mat[i][j];

scanf("%d", &element);

if(element != 0)

{

sparse[k][0]=i;

sparse[k][1]=j;

sparse[k][2]=element;

k++;

count++;

}

}

}

sparse[0][0]=5;

sparse[0][1]=5;

sparse[0][2]=count;

initialize();

printf(YELLOW "\nSparse Matrix form:\n\n");

for(i = 0; i < count+1; i++)

{

for(j = 0; j < 3; j++)

{

if(i==0)

{

printf(RED "%d \t",sparse[i][j]);

}

else

{

printf(GREEN "%d \t",sparse[i][j]);

}

}

printf("\n\n");

}

getch();

closegraph();

return 0;

}

**AIM — TO CREATE A LINKED LIST HAVING INFORMATION ABOUT A STUDENT AND**

**PERFORM INSERTION, DELETION AND REVERSAL**

<Source Listing for **SINGLY\_LL.H**>

struct node

{

int enrollNo;

char name[20];

struct node \*next;

};

typedef struct node node;

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

node \*createNode()

{

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

printf("\n\nEnter Enrollment No.: ");

scanf("%d", &(temp -> enrollNo));

printf("\nEnter Name of Student: ");

scanf("\n%s", temp -> name);

temp -> next = NULL;

return temp;

}

void reverse()

{

node \*p1, \*p2,\*p3;

p3=start;

p1 = p2 = NULL;

while(p3!=NULL)

{

p1=p2;

p2=p3;

p3=p3->next;

p2->next=p1;

}

start=p2;

}

void deletenode(int roll)

{

i = start;

while(i -> next != NULL)

{

if((i->next -> enrollNo) == roll)

break;

i = i -> next;

}

temp=i->next;

i= i -> next-> next;

free(temp);

}

void insertnodeafter(int roll)

{

if(start == NULL)

{

start=createNode();

}

else

{

i = start;

while(i -> next != NULL)

{

if((i -> enrollNo) == roll)

break;

i = i -> next;

}

temp=createNode();

temp -> next = i -> next;

i -> next = temp;

}

}

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

void displaylist()

{

i=start;

printf("\n\n The Current List of student with EnRoll status is :\n");

while(i!=NULL)

{

printf("%d -> ",i->enrollNo);

i=i->next;

}

}

void displayme()

{

int j=0,k=0,color=2;

char buf[50];

initialize();

i = start;

while(i != NULL)

{

if(j<=450)

{

//DATA

sprintf(buf,"%d", i->enrollNo);

outtextxy(j+75,15+k,buf);

outtextxy(j+35,40+k,i->name);

sprintf(buf,"%d",i->next);

outtextxy(j+115,15+k,buf);

sprintf(buf,"%d",i);

outtextxy(j+115,60+k,buf);

//DATA HEADING

setcolor(10);

outtextxy(j+35,k+15, "RNo. ");

outtextxy(j+35,k+30, "Name");

outtextxy(j+115,k+40, "NEXT");

outtextxy(j+35,k+60, "CURRENT");

//boxes

setcolor(color);

rectangle(j+30,k+10,j+110,k+50);

rectangle(j+110,k+10,j+150,k+50);

rectangle(j+30,k+50,j+150,k+70);

color++;

delay(200);

//Arrow

setcolor(15);

line(j+130,k+29,j+200,k+29);

if(i->next!=NULL)

{

setcolor(15);

outtextxy(j+190,k+23, " \\ ");

outtextxy(j+190,k+30, " / ");

}

else

{

setcolor(15);

line(j+200,k+29,j+200,k+49);

outtextxy(j+190,k+55, "NULL");

}

setcolor(15);

j+=180;

delay(1000);

i=i->next;

}

else

{

j=0; k+=150;

}

}

getch();

closegraph();

}

void menu()

{

int choice,roll;

printf("\n\nWhat do you want to do?

\n1. DISPLAY the Linked List

\n2. INSERT a new node

\n3. DELETE a node

\n4. REVERSE the Linked List

\nEnter [1 - 4]:");

scanf("%d", &choice);

switch(choice)

{

case 1:

displayme();

break;

case 2:

displaylist();

printf("\n Insert Record After Enroll Number: \n");

scanf("%d", &roll);

insertnodeafter(roll);

break;

case 3:

displaylist();

printf("\n Delete Record With Enroll Number: \n");

scanf("%d", &roll);

deletenode(roll);

break;

case 4:

reverse();

break;

default:

printf("\nPlease enter a valid option!!!\n");

}

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <graphics.h>

#include "**SINGLY\_LL.H**"

int main()

{

int again = 1;

clrscr();

printf("\nEnter data for first node:\n");

start = createNode();

while(again)

{

menu();

printf("\n Wanna do more operations?\nYes(1), No(0): ");

scanf("%d", &again);

}

return 0;

}

**AIM — TO CREATE A LINEAR QUEUE USING A LINEAR LINKED LIST AND PERFORM**

**OPERATIONS LIKE INSERTION, DELETION and TRAVERSAL**

<Source Listing for **QUEUE.H**>

struct node

{

int enrollNo;

char name[20];

struct node \*next;

};

typedef struct node node;

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

node \* createNode()

{

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

printf("\n\nEnter Enrollment No.: ");

scanf("%d", &(temp -> enrollNo));

printf("\nEnter Name of Student: ");

scanf("\n%s", temp -> name);

temp -> next = NULL;

return temp;

}

void push()

{

if(start==NULL)

{start=createNode();}

else

{

i=start;

while(i->next!=NULL)

{i=i->next;}

temp=createNode();

i->next=temp;

}

}

void del()

{

i=start;

start=start->next;

free(i);

}

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

void displaylist()

{

i=start;

printf("\n\n The Current List of student with EnRoll status is :\n");

while(i!=NULL)

{

printf("%d -> ",i->enrollNo);

i=i->next;

}

}

void displayme()

{

int j=0,k=0,color=2;

char buf[50];

initialize();

i = start;

while(i != NULL)

{

if(j<=450)

{

//DATA

sprintf(buf,"%d", i->enrollNo);

outtextxy(j+75,15+k,buf);

outtextxy(j+35,40+k,i->name);

sprintf(buf,"%d",i->next);

outtextxy(j+115,15+k,buf);

sprintf(buf,"%d",i);

outtextxy(j+115,60+k,buf);

//DATA HEADING

setcolor(10);

outtextxy(j+35,k+15, "RNo. ");

outtextxy(j+35,k+30, "Name");

outtextxy(j+115,k+40, "NEXT");

outtextxy(j+35,k+60, "CURRENT");

//boxes

setcolor(color);

rectangle(j+30,k+10,j+110,k+50);

rectangle(j+110,k+10,j+150,k+50);

rectangle(j+30,k+50,j+150,k+70);

color++;

delay(200);

//Arrow

setcolor(15);

line(j+130,k+29,j+200,k+29);

if(i->next!=NULL)

{

setcolor(15);

outtextxy(j+190,k+23, " \\ ");

outtextxy(j+190,k+30, " / ");

}

else

{

setcolor(15);

line(j+200,k+29,j+200,k+49);

outtextxy(j+190,k+55, "NULL");

}

setcolor(15);

j+=180;

delay(1000);

i=i->next;

}

else

{

j=0; k+=150;

}

}

getch();

closegraph();

}

void menu()

{

int choice,roll;

printf("\n\nWhat do you want to do?

\n1. DISPLAY the Queue

\n2. INSERT a new node(PUSH)

\n3. DELETE a node [1 - 3]: n");

scanf("%d", &choice);

switch(choice)

{

case 1:

displayme();

break;

case 2:

displaylist();

push();

displaylist();

break;

case 3:

del();

displaylist();

break;

default:

printf("\nPlease enter a valid option!!!\n");

}

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <graphics.h>

#include "**QUEUE.H**"

int main()

{

int again = 1;

clrscr();

printf("\nEnter data for first node:\n");

start = createNode();

while(again)

{

menu();

printf("\n Wanna do more operations?\nYes(1), No(0): ");

scanf("%d", &again);

}

return 0;

}

**AIM — TO CREATE A STACK AND PERFORM PUSH, POP AND TRAVERSE OPERATIONS**

**USING A LINEAR LINKED LIST**

<Source Listing for **STACK.H**>

struct node

{

int enrollNo;

char name[20];

struct node \*next;

};

typedef struct node node;

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

node \* createNode()

{

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

printf("\n\nEnter Enrollment No.: ");

scanf("%d", &(temp -> enrollNo));

printf("\nEnter Name of Student: ");

scanf("\n%s", temp -> name);

temp -> next = NULL;

return temp;

}

void push()

{

if(start==NULL)

{start=createNode();}

else

{

i=start;

while(i->next!=NULL)

{i=i->next;}

temp=createNode();

i->next=temp;

}

}

void pop()

{

i=start;

while(i->next->next!=NULL)

{

i=i->next;

}

free(i->next);

i->next=NULL;

}

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

void displaylist()

{

i=start;

printf("\n\n The Current List of student with EnRoll status is :\n");

while(i!=NULL)

{

printf("%d -> ",i->enrollNo);

i=i->next;

}

}

void displayme()

{

int j=0,k=0,color=2;

char buf[50];

initialize();

i = start;

while(i != NULL)

{

if(j<=450)

{

//DATA

sprintf(buf,"%d", i->enrollNo);

outtextxy(j+75,15+k,buf);

outtextxy(j+35,40+k,i->name);

sprintf(buf,"%d",i->next);

outtextxy(j+115,15+k,buf);

sprintf(buf,"%d",i);

outtextxy(j+115,60+k,buf);

//DATA HEADING

setcolor(10);

outtextxy(j+35,k+15, "RNo. ");

outtextxy(j+35,k+30, "Name");

outtextxy(j+115,k+40, "NEXT");

outtextxy(j+35,k+60, "CURRENT");

//boxes

setcolor(color);

rectangle(j+30,k+10,j+110,k+50);

rectangle(j+110,k+10,j+150,k+50);

rectangle(j+30,k+50,j+150,k+70);

color++;

delay(200);

//Arrow

setcolor(15);

line(j+130,k+29,j+200,k+29);

if(i->next!=NULL)

{

setcolor(15);

outtextxy(j+190,k+23, " \\ ");

outtextxy(j+190,k+30, " / ");

}

else

{

setcolor(15);

line(j+200,k+29,j+200,k+49);

outtextxy(j+190,k+55, "NULL");

}

setcolor(15);

j+=180;

delay(1000);

i=i->next;

}

else

{

j=0; k+=150;

}

}

getch();

closegraph();

}

void menu()

{

int choice,roll;

printf("\n\nWhat do you want to do?

\n1. DISPLAY the STACK

\n2. INSERT a new node(PUSH)

\n3. DELETE a node(POP) [1 - 3]: \n");

scanf("%d", &choice);

switch(choice)

{

case 1:

displayme();

break;

case 2:

displaylist();

push();

displaylist();

break;

case 3:

pop();

displaylist();

break;

default:

printf("\nPlease enter a valid option!!!\n");

}

}

<SOURCE LISTING FOR **MAIN.C**>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <graphics.h>

#include "**STACK.H**"

int main()

{

int again = 1;

clrscr();

printf("\nEnter data for first node:\n");

start = createNode();

while(again)

{

menu();

printf("\n Wanna do more operations?\nYes(1), No(0): ");

scanf("%d", &again);

}

return 0;

}

**AIM — TO CREATE A DOUBLY LINKED LIST WITH EMPLOYEE INFORMATION AND**

**PERFORM INSERTION AT FRONT, AND DELETION AT BACK**

<Source Listing for DLL.H>

struct node

{

int empNo;

char name[20];

struct node \*next;

struct node \*prev;

};

typedef struct node node;

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

node \* createNode()

{

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

printf("\nEnter Employee No.: ");

scanf("%d", &(ptr -> empNo));

printf("\nEnter Name of Employee: ");

scanf("\n%s", ptr -> name);

ptr -> next = NULL;

ptr -> prev = NULL;

return ptr;

}

void insertAtbeg()

{

if(start==NULL)

{start=createNode();}

else

{

temp=createNode();

temp->next=start;

start->prev=temp;

start=temp;

}

}

void deleteatend()

{

i=start;

while(i->next->next!=NULL)

{i=i->next;}

free(i->next);

i->next=NULL;

}

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

void displayme()

{

int j=0,k=0,color=2;

char buf[50];

initialize();

i = start;

while(i != NULL)

{

if(j<=450)

{

//DATA

sprintf(buf,"%d",i->empNo);

outtextxy(j+115,15+k,buf);

outtextxy(j+75,40+k,i->name);

sprintf(buf,"%d",i->next);

outtextxy(j+155,15+k,buf);

sprintf(buf,"%d",i);

outtextxy(j+155,60+k,buf);

sprintf(buf,"%d",i->prev);

outtextxy(j+35,15+k,buf);

//DATA HEADING

setcolor(10);

outtextxy(j+75,k+15, "ENo. ");

outtextxy(j+75,k+30, "Name");

outtextxy(j+155,k+40, "NEXT");

outtextxy(j+35,k+40, "PREV");

outtextxy(j+35,k+60, "CURRENT");

//boxes

setcolor(color);

rectangle(j+30,k+10,j+70,k+50);

rectangle(j+70,k+10,j+150,k+50);

rectangle(j+150,k+10,j+190,k+50);

rectangle(j+30,k+50,j+190,k+70);

color++;

delay(200);

//Arrow

setcolor(15);

line(j+170,k+28,j+250,k+28);

if(i->prev!=NULL)

{

line(j-40,k+38,j+55,k+38);

outtextxy(j-50,k+33, " / ");

outtextxy(j-50,k+40, " \\ ");

}

else

{

line(10,28,50,28);

line(10,28,10,70);

outtextxy(5,75, "NULL");

}

if(i->next!=NULL)

{setcolor(15);

outtextxy(j+240,k+23, " \\ ");

outtextxy(j+240,k+30, " / ");

}

else

{setcolor(15);

line(j+250,k+29,j+250,k+49);

outtextxy(j+240,k+55, "NULL");

}

setcolor(15);

j+=240;

delay(1000);

i=i->next;

}

else

{

j=0;k+=150;

}

}

getch();

closegraph();

}

void displaylist()

{

i=start;

printf("\n The Current List of student with EnRoll status is :\n");

while(i!=NULL)

{

printf("%d ->",i->empNo);

i=i->next;

}

}

void menu()

{

int choice;

printf("\nWhat do you want to do?

\n1. DISPLAY the Linked List

\n2. INSERT a new node

\n3. DELETE a node\nEnter [1 - 3]: ");

scanf("%d", &choice);

switch(choice)

{

case 1:

displayme();

break;

case 2:

displaylist();

insertAtbeg();

break;

case 3:

displaylist();

deleteatend();

break;

default:

printf("\nPlease enter a valid option!!!\n");

}

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <graphics.h>

#include "**DLL.H**"

int main()

{ int again = 1;

clrscr();

printf("\nEnter data for first node:\n");

start = createNode();

while(again)

{

menu();

printf("\nPerform more operations?\nYes(1), No(0): ");

scanf("%d", &again);

}

return 0;

}

**AIM — TO CREATE A CIRCULAR LINKED LIST WITH INFO ABOUT ACLASS AND**

**PERFORM INSERTION AND DELETION AT THE ENDS**

<Source Listing for **CIRCLL.H**>

struct node

{

int enrollNo;

char name[20];

struct node \*next;

};

typedef struct node node;

node \*start=NULL,\*i=NULL,\*end=NULL,\*temp=NULL;

node \* createNode()

{

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

printf("\n\nEnter Enrollment No.: ");

scanf("%d", &(temp -> enrollNo));

printf("\nEnter Name of Student: ");

scanf("\n%s", temp -> name);

temp -> next = NULL;

return temp;

}

void insertAtbeg()

{

if(start==NULL)

{start=createNode();}

else

{

temp=createNode();

temp->next=start;

start=temp;

}

}

void deleteatend()

{

i=start;

while(i->next->next!=end)

{i=i->next;}

free(i->next);

i->next=start;

}

void initialize()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

cleardevice();

}

void displaylist()

{

i=start;

printf("\n\n The Current List of student with EnRoll status is :\n");

while(i!=end)

{

printf("%d -> ",i->enrollNo);

i=i->next;

}

}

void displayme()

{

int j=0,k=0,color=2;

char buf[50];

initialize();

i = start;

while(i != end)

{

if(j<=450)

{

//DATA

sprintf(buf,"%d", i->enrollNo);

outtextxy(j+75,15+k,buf);

outtextxy(j+35,40+k,i->name);

sprintf(buf,"%d",i->next);

outtextxy(j+115,15+k,buf);

sprintf(buf,"%d",i);

outtextxy(j+115,60+k,buf);

//DATA HEADING

setcolor(10);

outtextxy(j+35,k+15, "RNo. ");

outtextxy(j+35,k+30, "Name");

outtextxy(j+115,k+40, "NEXT");

outtextxy(j+35,k+60, "CURRENT");

//boxes

setcolor(color);

rectangle(j+30,k+10,j+110,k+50);

rectangle(j+110,k+10,j+150,k+50);

rectangle(j+30,k+50,j+150,k+70);

color++;

delay(200);

//Arrow

setcolor(15);

line(j+130,k+29,j+200,k+29);

if(i->next!=end)

{setcolor(15);

outtextxy(j+190,k+23, " \\ ");

outtextxy(j+190,k+30, " / ");

}

else

{setcolor(15);

line(j+200,k+29,j+200,k+49);

outtextxy(j+190,k+55, "NULL");

}

setcolor(15);

j+=180;

delay(1000);

i=i->next;

}

else

{

j=0; k+=150;

}

}

getch();

closegraph();

}

void menu()

{

int choice,roll;

printf("\n\nWhat do you want to do?

\n1. DISPLAY the Linked List

\n2. INSERT a new node

\n3. DELETE a node\n [1 - 3]: ");

scanf("%d", &choice);

switch(choice)

{

case 1: displayme();

break;

case 2:

displaylist();

insertAtbeg();

break;

case 3:

displaylist();

deleteatend();

break;

default:

printf("\nPlease enter a valid option!!!\n");

}

}

<Source Listing for **MAIN.C**>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <graphics.h>

#include "**CIRCLL.h**"

int main()

{

int again = 1;

clrscr();

printf("\nEnter data for first node:\n");

start = createNode();

end=start;

end->next=start;

while(again)

{

menu();

printf("\n Wanna do more operations?\nYes(1), No(0): ");

scanf("%d", &again);

}

return 0;

}