1)Program for Array Insertion , Deletion and Traversal in Array

# include <stdio.h>

int main() {

    int arr[50],n=5,i;

    printf("Enter five elements : ");

    for(i=0;i<n;i++){

        scanf("%d",&arr[i]);

    }

    printf("Array before insertion \n");

    for(i=0;i<n;i++){

        printf("%d ",arr[i]);

    }

    printf("\n");

    int data,pos;

    printf("Enter data you want to insert in array : ");

    scanf("%d",&data);

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

    scanf("%d",&pos);

    for(i=n-1;i>=pos-1;i--){

        arr[i+1]=arr[i];

    }

    arr[pos-1]=data;

    n++;

    printf("Array after insertion \n");

    for(i=0;i<n;i++){

        printf("%d ",arr[i]);

    }

    printf("\n");

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

    scanf("%d",&pos);

    for(i=pos-1;i<n;i++){

        arr[i]=arr[i+1];

    }

    n--;

    printf("Array after deletion \n");

    for(i=0;i<n;i++){

        printf("%d ",arr[i]);

    }

    printf("\n");

return 0;

}

Output :

Enter five elements : 1 2 3 4 5

Array before insertion

Enter data you want to insert in array : 23

Enter position you want to insert at : 4

Array after insertion

1 2 3 23 4 5

Enter position you want to delete at : 4

Array after deletion

1 2 3 4 5

2)Program for Matrix Multiplication

#include<stdio.h>

#include<stdlib.h>

int main(){

int a[10][10],b[10][10],mul[10][10],r,c,i,j,k;

printf("Enter the number of row = ");

scanf("%d",&r);

printf("Enter the number of column = ");

scanf("%d",&c);

printf("Enter the first matrix element = \n");

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

{

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

{

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

}

}

printf("Enter the second matrix element = \n");

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

{

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

{

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

}

}

printf("Multiply of the matrix = \n");

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

{

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

{

mul[i][j]=0;

for(k=0;k<c;k++)

{

mul[i][j]+=a[i][k]\*b[k][j];

}

}

}

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

{

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

{

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

}

printf("\n");

}

return 0;

}

Output :

Enter the number of row = 3

Enter the number of column = 3

Enter the first matrix element =

1 2 1

3 4 7

8 1 4

Enter the second matrix element =

2 4 1

7 5 9

2 3 1

Multiply of the matrix =

18 17 20

48 53 46

31 49 21

3)Program for finding length

# include <stdio.h>

# include <string.h>

int main() {

    char str[50];

    printf("Enter the String : ");

    gets(str);

    int i=0,count=0;

    while(str[i]!='\0'){

        count++;

        i++;

    }

    printf("%d\n",count);

return 0;

}

Output :

Enter the String : Hello

5

Singly linked list

4) Program for reversing the string

# include <stdio.h>

# include <string.h>

int main() {

    char str[50],temp;

    printf("Enter the String : ");

    gets(str);

    int i=0,count=0,j;

    while(str[i]!='\0'){

        count++;

        i++;

    }

    j=count-1;

    i=0;

    while(i!=count/2 && j!=count/2){

        temp=str[i];

        str[i]=str[j];

        str[j]=temp;

        i++;

        j--;

    }

    i=0;

    while(str[i]!='\0'){

        printf("%c",str[i]);

        i++;

    }

return 0;

}

Output :

Enter the String : Hello World

dlroW olleH

5) Program to write and call sum function

# include <stdio.h>

int sum(int a,int b){

    return a+b;

}

int main() {

    printf("Function is being called : \n");

    printf("%d",sum(2,4));

return 0;

}

Output :

Function is being called :

6

7) Singly linked list insertion , deletion.

#include <stdio.h>

#include<stdlib.h>

struct Node{

int data;

struct Node \*next;

};

void insertion(struct Node \*\*head){

int data;

scanf("%d",&data);

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

newnode->data=data;

newnode->next=NULL;

struct Node \*temp=\*head;

if((\*head)==NULL){

\*head=newnode;

return;

}

while(temp->next!=NULL){

temp=temp->next;

}

temp->next=newnode;

return;

}

void print(struct Node \*head){

while(head!=NULL){

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

head=head->next;

}

printf("\n");

}

void delete(struct Node \*\*head,int key){

struct Node \*temp=\*head;

struct Node \*prev;

if(temp!=NULL&&temp->data==key){

\*head=temp->next;

free(temp);

return;

}

while(temp!=NULL&&temp->data!=key){

prev=temp;

temp=temp->next;

}

if(temp==NULL){return;}

prev->next=temp->next;

free(temp);

}

int main()

{

struct Node \*head=NULL;

insertion(&head);

insertion(&head);

insertion(&head);

insertion(&head);

print(head);

int num;

scanf("%d",&num);

delete(&head,num);

print(head);

return 0;

}

8) Doubly linked list insertion , deletion , display

#include<stdio.h>

#include<stdlib.h>

struct Node{

struct Node \*prev;

int data;

struct Node \*next;

};

struct Node \*createNode(){

struct Node \*newNode;

newNode=(struct Node \*)malloc(sizeof(struct Node));

newNode->prev=NULL;

newNode->next=NULL;

return newNode;

}

void insertion(struct Node \*\*head){

int data;

scanf("%d",&data);

struct Node \*newNode=createNode();

newNode->data=data;

struct Node \*temp=\*head;

if((\*head)==NULL){

\*head=newNode;

}

else{

while(temp->next!=NULL){

temp=temp->next;

}

temp->next=newNode;

newNode->prev=temp;

}

}

void del\_beg(struct Node \*\*head){

struct Node \*temp=\*head;

(\*head)=(\*head)->next;

(\*head)->prev=NULL;

free(temp);

}

void del\_end(struct Node \*\*head){

struct Node \*temp=\*head;

struct Node \*prev=NULL;

while(temp->next!=NULL){

prev=temp;

temp=temp->next;

}

if(prev==NULL){

\*head=NULL;

}

else{

prev->next=NULL;

free(temp);

}

}

void del\_pos(struct Node \*\*head,int pos){

struct Node \*temp=\*head;

struct Node \*prv;

while(pos--){

prv=temp;

temp=temp->next;

}

if(temp==NULL){

\*head=NULL;

free(prv);

}

else if(temp->next==NULL){

\*head=temp;

free(prv);

}

else{

free(prv);

\*head=temp;

}

}

void print(struct Node \*head){

while(head!=NULL){

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

head=head->next;

}

printf("\n");

}

int main()

{

struct Node \*head=NULL;

insertion(&head);

insertion(&head);

insertion(&head);

insertion(&head);

print(head);

del\_beg(&head);

print(head);

del\_end(&head);

print(head);

printf("Enter position: ");

int pos;

scanf("%d",&pos);

del\_pos(&head,pos);

print(head);

return 0;

}

9) Circular link list insertion , deletion

#include <stdio.h>

#include <stdlib.h>

struct node{

int data;

struct node \*next;

};

struct node \* createnode(){

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

newnode->next=NULL;

return newnode;

}

void insertion(struct node \*\*head){

int data;

scanf("%d",&data);

struct node \*newnode=createnode();

newnode->data=data;

struct node \*temp=\*head;

if((\*head)==NULL){

newnode->next=newnode;

\*head=newnode;

return;

}

while(temp->next!=(\*head)){

temp=temp->next;

}

temp->next=newnode;

newnode->next=\*head;

}

void delete\_beg(struct node \*\*head){

if((\*head)==NULL){

printf("list is empty");

return;

}

if((\*head)->next==(\*head)){

\*head=NULL;

return;

}

struct node \*temp=\*head;

struct node \*temp2=\*head;

while(temp->next!=(\*head)){

temp=temp->next;

}

temp->next=(\*head)->next;

\*head=(\*head)->next;

free(temp2);

}

void delete\_end(struct node \*\*head){

if((\*head)==NULL){

printf("LIST IS EMPTY");

return;

}

struct node \*temp=\*head;

if(temp->next==(\*head)){

\*head=NULL;

return;

}

struct node \*prev=NULL;

while(temp->next!=(\*head)){

prev=temp;

temp=temp->next;

}

prev->next=\*head;

free(temp);

}

void delete\_pos(struct node \*\*head){

int pos;

scanf("%d",&pos);

struct node \*temp=\*head;

struct node \*prev=NULL;

while(pos--){

prev=temp;

temp=temp->next;

}

prev->next=temp->next;

free(temp);

}

void print(struct node \*head){

struct node \*temp=head;

while(temp->next!=head){

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

temp=temp->next;

}

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

printf("\n");

}

int main()

{

struct node \*head=NULL;

insertion(&head);

insertion(&head);

insertion(&head);

insertion(&head);

insertion(&head);

print(head);

delete\_beg(&head);

print(head);

delete\_end(&head);

print(head);

delete\_pos(&head);

print(head);

return 0;

}