1)Implementation of stack using linked list.

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head;

int ele;

void push();

void pop();

void display();

int main()

{

int ch;

do

{

printf("\n 1.push");

printf("\n 2.pop");

printf("\n 3.display");

printf("\n 4.exit");

printf("\n enter your choice");

scanf("%d",&ch);

switch(ch)

{

case 1:

push();

break;

case 2:

pop();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

}

}while(ch!=4);

return 0;

}

void push()

{

struct node \*ptr;

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

if(ptr==NULL)

{

printf("\n stack is pverflow");

}

else

{

printf("enter the element");

scanf("%d",&ele);

if(head==NULL)

{

ptr->data=ele;

ptr->next=NULL;

head=ptr;

}

else

{

ptr->data=ele;

ptr->next=head;

head=ptr;

}

printf("\n item is pushed");

}

}

void display()

{

struct node \*ptr;

ptr=head;

if(ptr==NULL)

{

printf("\n stack is empty");

}

else

{

printf("\n stack is \n");

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

}

}

void pop()

{

struct node \*ptr;

if(head==NULL)

{

printf("\n stack is empty");

}

else

{

ptr=head;

printf("%d is poped",ptr->data);

head=head->next;

free(ptr);

}

}

2)Implementation of queue using linked list.

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*front;

struct node \*rear;

int ele;

void insert();

void delete();

void display();

int main()

{

int ch;

do

{

printf("\n 1.insert");

printf("\n 2.delete");

printf("\n 3.display");

printf("\n 4.exit");

printf("\n enter your choice");

scanf("%d",&ch);

switch(ch)

{

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

}

}while(ch!=4);

return 0;

}

void insert()

{

struct node \*ptr;

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

if(ptr==NULL)

{

printf("\noverflow\n");

return;

}

else

{

printf("enter the element");

scanf("%d",&ele);

if(front==NULL)

{

front=ptr;

rear=ptr;

front->next=NULL;

rear->next=NULL;

}

else

{

rear->next=ptr;

rear=ptr;

rear->next=NULL;

}

printf("\n item is inserted");

}

}

void display()

{

struct node \*ptr;

ptr=front;

if(front==NULL)

{

printf("\nempty queue\n");

}

else

{

printf("\n queue is \n");

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

}

}

void delete()

{

struct node \*ptr;

if(front==NULL)

{

printf("\nunderflow\n");

return;

}

else

{

ptr=front;

front=front->next;

free(ptr);

printf("\n item is deleted");

}

}

3)Decimal to hexadecimal matrix program.

#include<stdio.h>

int main()

{

int n,a[10][10],i,j;

scanf("%d",&n);

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

{

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

{

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

}

}

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

{

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

{

if(i!=0 && j!=0 && i!=n-1 && j!=n-1)

{

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

}

}

printf("\n");

}

return 0;

}

4)Find out given 2 input strings are anagrams or not.

#include<stdio.h>

int main()

{

char str1[100],str2[100];

int a[26]={0};

int b[26]={0};

int i,flag;

scanf("%s%s",str1,str2);

for(i=0;str1[i]!='\0';i++)

{

a[str1[i]-'a']++;

}

for(i=0;str2[i]!='\0';i++)

{

b[str2[i]-'a']++;

}

for(i=0;i<26;i++)

{

if(a[i]!=b[i])

{

flag=1;

break;

}

}

if(flag==0)

printf("\nstrings are anagrams/yes");

else

printf("\nstrings are not anagrams/no");

return 0;

}