Linear queue

#include<stdio.h>

#define MAX 5

struct queue

{

int a[MAX];

int front,rear;

}q;

void init()

{

q.front=-1;

q.rear=-1;

}

int isempty()

{

if(q.front==-1 || q.front > q.rear)

return 1;

else

return 0;

}

int isfull()

{

if(q.rear==MAX-1)

return 1;

else

return 0;

}

void insert(int num)

{

if(q.front==-1)

q.front=0;

if(isfull())

{

printf("queue is full");

}

else

{

q.rear++;

q.a[q.rear]=num;

}

}

void delete()

{

int val;

if(isempty())

{

printf("queue is empty");

}

else

{

val=q.a[q.front];

q.front++;

}

}

void reverse()

{

int i;

for(i=q.rear;i>=0;i--)

{

printf("%d\t",q.a[i]);

}

}

int peek()

{

return (q.a[q.front]);

}

void display()

{

int i;

for(i=q.front;i<=q.rear;i++)

{

printf("%d\t",q.a[i]);

}

}

int main()

{

init();

int ch,num;

do

{

printf("\n 1:insert \n 2:display \n 3:delete \n 4:reverse \n 5:peek");

printf("\n enter choice=");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter num=");

scanf("%d",&num);

insert(num);

break;

case 2:display(num);

break;

case 3:delete();

break;

case 4:reverse();

break;

case 5:printf("top most element=%d",peek());

break;

}

}while(ch<6);

}

Circular queue

#include<stdio.h>

#define MAX 5

struct queue

{

int a[MAX];

int front,rear;

}q;

void init()

{

q.front=-1;

q.rear=-1;

}

int isempty()

{

if(q.front==-1)

return 1;

else

return 0;

}

int isfull()

{

if((q.front==0 && q.rear==MAX-1) || (q.front==q.rear+1))

return 1;

else

return 0;

}

void insert(int num)

{

if(isfull())

{

printf("queue is full");

}

else

{

if(q.front==-1)

{

q.front=0;

}

q.rear=(q.rear+1)%MAX;

q.a[q.rear]=num;

}

}

void delete()

{

int val;

if(isempty())

{

printf("queue is empty");

}

else

{

val=q.a[q.front];

if(q.front==q.rear)

{

printf("queue is empty");

q.front=q.rear=-1;

}

q.front=(q.front+1)%MAX;

}

}

void display()

{

int i;

if(q.front==-1)

{

printf("queue is empty");

}

i=q.front-1;

do

{

i=(i+1)%MAX;

printf("%d\t",q.a[i]);

}while(i!=q.rear);

}

int main()

{

init();

int num,ch;

do

{

printf("\n 1:insert \n 2:delete \n 3:display");

printf("\n enter choice=");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter num=");

scanf("%d",&num);

insert(num);

break;

case 2:delete();

break;

case 3:display();

break;

}

}while(ch<4);

}

Dynamic linear queue

#include<stdio.h>

#include<stdlib.h>

typedef struct queue

{

int data;

struct node \*next;

}NODE;

NODE \*front=NULL;

NODE \*rear=NULL;

int isempty()

{

if(front==NULL)

return 1;

else

return 0;

}

void insert(int num)

{

NODE \*newnode;

newnode=(NODE\*)malloc(sizeof(NODE));

newnode->data=num;

newnode->next=NULL;

if(front==NULL)

{

front=rear=newnode;

}

else

{

rear->next=newnode;

rear=newnode;

}

}

void delete()

{

int val;

if(isempty())

{

printf("queue is empty");

}

else

{

val=front->data;

front=front->next;

}

}

int peek()

{

return (front->data);

}

void display()

{

NODE \*temp;

for(temp=front;temp!=NULL;temp=temp->next)

{

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

}

}

int main()

{

int ch,num;

do

{

printf("\n 1:insert \n 2:delete \n 3:display \n 4:peek");

printf("\n enter choice=");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter num=");

scanf("%d",&num);

insert(num);

break;

case 2:delete();

break;

case 3:display();

break;

case 4:printf("top most element=%d",peek());

break;

}

}while(ch<5);

}

Dynamic circular queue

#include<stdio.h>

#include<stdlib.h>

typedef struct queue

{

int data;

struct queue \*next;

}NODE;

NODE \*front=NULL;

NODE \*rear=NULL;

int isempty()

{

if(front==NULL || front >= rear)

return 1;

else

return 0;

}

void insert(int num)

{

NODE \*newnode;

newnode=(NODE\*)malloc(sizeof(NODE));

newnode->data=num;

if(front==NULL)

{

front=rear=newnode;

rear->next=front;

}

else

{

rear->next=newnode;

rear=newnode;

rear->next=front;

}

}

void delete()

{

if(isempty())

{

printf("queue is empty");

}

else

{

NODE \*temp;

temp=front;

front=front->next;

rear->next=front;

free(temp);

}

}

int peek()

{

return (front->data);

}

void display()

{

NODE \*temp=front;

do

{

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

temp=temp->next;

}while(temp!=rear->next);

}

int main()

{

int num,ch;

do

{

printf("\n 1:insert \n 2:delete \n 3:display \n 4:peek");

printf("\n enter choice=");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter num=");

scanf("%d",&num);

insert(num);

break;

case 2:delete();

break;

case 3:display();

break;

case 4:printf("top most element=%d",peek());

break;

}

}while(ch<5);

}

Priority queue

#include<stdio.h>

#define MAX 5

struct queue

{

int a[MAX];

int front,rear;

}q;

void init()

{

q.front=-1;

q.rear=-1;

}

int isempty()

{

if(q.front==-1 || q.front > q.rear)

return 1;

else

return 0;

}

int isfull()

{

if(q.rear==MAX-1)

return 1;

else

return 0;

}

void insert(int num)

{

int i;

if(isfull())

{

printf("queue is full");

}

else

{

if(q.front==-1)

{

q.front=0;

}

for(i=q.rear;i>=q.front;i--)

{

if(num > q.a[i])

q.a[i+1]=q.a[i];

else

break;

}

q.a[i+1]=num;

q.rear++;

}

}

void delete()

{

int val;

if(isempty())

{

printf("queue is empty");

}

else

{

val=q.a[q.front];

q.front++;

}

}

int peek()

{

return(q.a[q.front]);

}

void display()

{

int i;

for(i=q.front;i<=q.rear;i++)

{

printf("%d\t",q.a[i]);

}

}

int main()

{

int ch,num;

init();

do

{

printf("\n 1:insert \n 2:delete \n 3:display \n 4:peek");

printf("\n enter choice=");

scanf("%d",&ch);

switch(ch)

{

case 1:printf("enter num=");

scanf("%d",&num);

insert(num);

break;

case 2:delete();

break;

case 3:display();

break;

case 4:printf("top most element=%d",peek());

break;

}

}while(ch<5);

}