#include<stdio.h>

#include<stdlib.h>

#define size 5

void insert(int queue[size],int \*front,int \*rear,int item)

{

if((\*front==0 && \*rear==size-1) || (\*front==\*rear+1))

printf("\n\nQueue is full.");

else

{

printf("\n\nEnter ITEM: ");

scanf("%d", &item);

if(\*rear == -1)

{

\*rear = 0;

\*front = 0;

}

else if(\*rear == size-1)

\*rear = 0;

else

(\*rear)++;

queue[\*rear] = item;

printf("\n\nItem inserted: %d\n", item);

}

}

void delet(int queue[size],int \*front,int \*rear)

{

int item;

if(\*front == -1)

printf("\n\nQueue is empty.\n");

else

{

item = queue[\*front];

if(\*front == \*rear)

{

\*front = -1;

\*rear = -1;

}

else if(\*front == size-1)

\*front = 0;

else

(\*front)++;

printf("\n\nITEM deleted: %d", item);

}

}

void display(int queue[size],int \*front,int \*rear)

{

int i;

if((\*front == -1) || (\*front==\*rear+1))

printf("\n\nQueue is empty.\n");

else

{

printf("\n\n");

for(i=\*front; i<=\*rear; i++)

printf("%d ",queue[i]);

}

}

void main()

{

int ch;

int queue[size], rear=-1, front=-1, item;

do

{

system("cls");

printf("1.Insert\n2.Delete\n3.Display\n4.Exit\n");

printf("\nEnter your choice: ");

scanf("%d", &ch);

switch(ch)

{

case 1:

insert(queue,&front,&rear,item);

break;

case 2:

delet(queue,&front,&rear);

fflush(stdin);

getchar();

break;

case 3:

display(queue,&front,&rear);

fflush(stdin);

getchar();

break;

case 4:

exit(0);

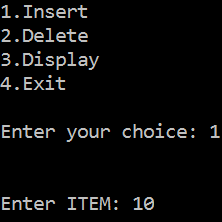
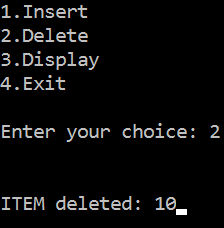
default:

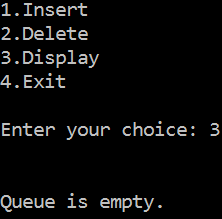
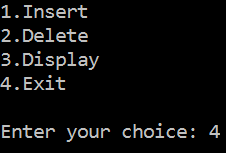
printf("\n\nInvalid choice. Pleasr try again...\n");

}

} while(1);

}

#include<stdio.h>

#include<stdlib.h>

#define pf printf

#define sf scanf

int count=0;

struct queue{

int info;

struct queue \*link;

};

typedef struct queue \* que;

void push(que \*front,que \*rear,int n)

{

que newnode;

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

newnode->info=n;

newnode->link=NULL;

if(count==0)

\*front=newnode;

else

(\*rear)->link=newnode;

\*rear=newnode;

(\*rear)->link=\*front;

count++;

}

int pop(que \*front,que \*rear)

{

int n;

que temp;

if(count==0)

return (-1);

count--;

if(\*front==\*rear)

{

n=(\*front)->info;

free(\*front);

\*front=NULL;

\*rear=NULL;

}else

{

temp= \*front ;

n = temp-> info ;

\*front = (\*front) -> link ;

(\*rear) -> link = \*front ;

free ( temp ) ;

}

return n;

}

void display(que \*front,que \*rear)

{

que temp;

int i;

if(count==0)

pf("Empty");

else

{

temp=\*front;

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

{

pf("%d ",temp->info);

temp=temp->link;

}

}

pf("\n");

}

int size()

{

return count;

}

void main()

{

que front=NULL,rear=NULL;

int n,ch=10;

while(ch!=0)

{

system("cls");

pf("1.Push\n");

pf("2.Pop\n");

pf("3.SizeOfQueue\n");

pf("4.Display\n");

pf("0.EXIT\n");

sf("%d",&ch);

switch(ch)

{

case 1:

{

pf("Enter the Number \n");

sf("%d",&n);

push(&front,&rear,n);

break;

}

case 2:

{

n=pop(&front,&rear);

if(n==-1)

pf("Queue is empty\n");

else

pf("Number poped from queue is %d\n",n);

fflush(stdin);

getchar();

break;

}

case 3:

{

n=size();

pf("Size of queue is %d\n",n);

fflush(stdin);

getchar();

break;

}

case 4:

{

pf("Queue is -->> ");

display(&front,&rear);

fflush(stdin);

getchar();

}

case 0:

break;

default:

pf("Wrong Choice\n");

fflush(stdin);

getchar();

break;

}

}

}