**Circular queue**

#include<stdio.h>

#include<stdlib.h>

#define MAX 5

int x,front=-1,rear=-1;

int data[MAX];

void enqueue();

int dequeue();

void display();

int main()

{

int c;

while(1)

{

printf("\n1.Insert");

printf("\n2.Delete");

printf("\n3.Display");

printf("\n4.Exit\n");

printf("Enter a choice:");

scanf("%d",&c);

switch(c)

{

case 1:enqueue();

break;

case 2:dequeue();

break;

case 3:display();

break;

case 4:exit(1);

default:printf("Invalid choice");

break;

}

}

}

void enqueue()

{

if(((rear==MAX-1)&&(front=0))||(front>0&&(rear==front-1)))

{

printf("\nQueue full\n");

}

else

{

printf("Enter element to insert:");

scanf("%d",&x);

if((front>0)&&(rear==MAX-1))

{

rear=0;

data[rear]=x;

}

else

{

if((front==0&&rear==-1)||(rear!=front-1))

{

data[++rear]=x;

}

}

}

}

int dequeue()

{

if(front==-1)

{

printf("\nQueue empty\n");

}

else if(front==rear)

{

x=data[front];

rear=-1;

front=0;

}

else if(front==MAX-1)

{

x=data[front];

front=0;

}

else

{

x=data[front++];

}

printf("\nDeleted item is %d\n",x);

}

void display()

{

int i,j;

if(front==0&&rear==-1)

{

printf("\nQueue empty\n");

}

else if(front>rear)

{

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

{

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

}

for(j=front;j<=MAX-1;j++)

{

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

}

printf("\nRear is at %d",data[j]);

printf("\nFront is at %d",data[j]);

}

else

{

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

{

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

}

printf("\nRear is at %d",data[i]);

printf("\nFront is at %d",data[i]);

}

printf("\n");

}

////////////////////////////////////////////////////////////////

**Double ended queue**

#include<stdio.h>

#define MAX 100

int data[MAX];

int rear=0,front=0;

int dequeueFront();

int dequeueRear();

void enqueueRear(int);

void enqueueFront(int);

void display();

int main()

{

int a,b,val;

while(1)

{

printf("\n1.Input-restricted dequeue");

printf("\n2.Output-restricted dequeue\n");

printf("Enter a choice:");

scanf("%d",&a);

switch(a)

{

case 1:

while(1)

{

printf("\n1.Insert");

printf("\n2.Delete from Rear");

printf("\n3.Delete from Front");

printf("\n4.Display\n");

printf("Enter a choice:");

scanf("%d",&b);

switch(b)

{

case 1:enqueueRear(val);

display();

break;

case 2:val=dequeueRear();

display();

break;

case 3:val=dequeueFront();

display();

break;

case 4:display();

break;

default:printf("Invalid choice");

break;

}

}

case 2:

while(1)

{

printf("\n1.Insert");

printf("\n2.Delete from Rear");

printf("\n3.Delete from Front");

printf("\n4.Display\n");

printf("Enter a choice:");

scanf("%d",&b);

switch(b)

{

case 1:enqueueRear(val);

display();

break;

case 2:enqueueFront(val);

display();

break;

case 3:val=dequeueFront();

display();

break;

case 4:display();

break;

default:printf("Invalid choice");

break;

}

}

}

}

}

void enqueueRear(int val)

{

if(front==MAX/2)

{

printf("\nQueue Full\n");

}

else

{

printf("Enter number to insert:");

scanf("%d",&val);

data[front]=val;

front++;

}

}

void enqueueFront(int val)

{

if(front==MAX/2)

{

printf("\nQueue Full\n");

}

else

{

printf("Enter number to insert:");

scanf("%d",&val);

rear--;

data[rear]=val;

}

}

int dequeueRear()

{

int x;

if(front==rear)

{

printf("\nQueue Empty\n");

}

front--;

x=data[front+1];

return x;

}

int dequeueFront()

{

int x;

if(front==rear)

{

printf("\nQueue Empty\n");

}

rear++;

x-data[rear-1];

return x;

}

void display()

{

int i;

if(front==rear)

{

printf("\nQueue Empty\n");

}

else

{

printf("Elements in double ended queue are\n");

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

{

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

}

}

}

////////////////////////////////////////////////////////////////