#include <iostream>

#include<stdlib.h>

using namespace std;

typedef int QElemType;

typedef struct QNode

{

QElemType data;

struct QNode \*next;

}QNode;

typedef struct

{

QNode \*front;

QNode \*rear;

}LinkQueue;

// 构造一个空队列q

LinkQueue \*InitQueue(LinkQueue \*q)

{

q->front = q->rear =(QNode\*)malloc(sizeof(QNode));

q->front->next = NULL;

return q;

}

// 元素入队

LinkQueue \*EnQueue(LinkQueue \*q, QElemType e)

{

QNode \*p = (QNode\*)malloc(sizeof(QNode));//为插入节点分配空间

if(!p)

{//分配空间失败

cout<<"插入节点内存分配失败！"<<endl;

}

else

{ //建节点

p->data = e; //为插入节点数据域赋值

p->next = NULL;//为插入节点指针域赋值

//实现插入

q->rear->next = p;//插入到队尾

q->rear = p;//队尾指针重新指向新任队尾

}

return q;

}

//元素出队

LinkQueue \*DeQueue(LinkQueue \*q)

{

QNode \*p;

if(q->front == q->rear)

{

cout<<"链队列已空，不可再执行删除操作！"<<endl;

}

else

{

p = q->front->next;//将欲删除的队头结点暂存给p

QElemType e = p->data;//把队头数据赋给e

cout<<"delete: "<<e<<endl;

q->front->next = p->next;//删除,将原队头结点的后继p->next赋值给头结点后继

if(q->rear == p)

{//若队头就是队尾，则删除后将rear指向头结点

cout<<"链队列数据全部删除完毕！"<<endl;

q->rear = q->front;

}

free(p);

}

return q;

}

//返回队头元素

void GetQHead(LinkQueue \*q)

{

QNode \*p;

if(q->front == q->rear)

{

cout<<"链队列为空，无法返回队头数据"<<endl;

}

else

{

p = q->front->next;//队头

cout<<"队头元素："<<p->data<<endl;

}

}

//求队列长度

void QueueLength(LinkQueue \*q)

{

int length = 0;

QNode \*p;

p = q->front->next;//队头

while(p)

{

length++;

p = p->next;

}

cout<<"队列长度："<<length<<endl;

}

//打印。带头结点，真正存储元素的位置从头结点下一位置（队头）开始！！！

void PrintQueue(LinkQueue \*q)

{

QNode \*p;//队头

p = q->front->next;//头结点的下一节点，即为队头！！！

while(p)

{//从队头开始，依次往后遍历

cout<<p->data<<" ";

p = p->next;

}

cout<<endl;

}

int main()

{

LinkQueue \*q = InitQueue(q);

EnQueue(q, 1);

PrintQueue(q);

EnQueue(q, 2);

PrintQueue(q);

EnQueue(q, 3);

PrintQueue(q);

EnQueue(q, 4);

PrintQueue(q);

GetQHead(q);

QueueLength(q);

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

DeQueue(q);

PrintQueue(q);

GetQHead(q);

DeQueue(q);

PrintQueue(q);

GetQHead(q);

DeQueue(q);

PrintQueue(q);

DeQueue(q);

PrintQueue(q);

QueueLength(q);

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

DeQueue(q);

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

return 0;

}