实验二 队列

班级**: 计算机17-1班** 姓名**: 冯子旋** 学号：**07**

# 一、实验目的

1．掌握循环队列的存储方式及基本操作的实现；

2．掌握利用队列解决问题的方法；

# 二、实验题目与要求

题目：循环队列的实现要求：

（1）编程实现循环队列的类型定义及其初始化、入队、出队、取队头、遍历等操作；

（2）编写程序，对所实现的循环队列进行验证。

# 三、实验内容

1．循环队列的实现

完成循环队列的类型定义及其基本操作函数代码，并保存为 “CirQueue.h”。设计完成空缺代码并上机调试。

typedef struct{

QElemType \*base; int front, rear;

int queueSize;

}CirQueue;

Status InitCirQueue(CirQueue &Q,int QSize)

{

//构造一个空间大小为 QSize 的初始空队列 Q

Q.base=(QElemType\*) malloc(QSize \*sizeof(QElemType));

if(!Q.base) return OVERFLOW; //分配存储空间失败

Q.queueSize=QSize;

Q.front=Q.rear=0; //置为空队

return OK;

}

Status queueIsEmpty(CirQueue Q)

{

//判断循环队列 Q 是否为空

if(Q.front==Q.rear) return TRUE; //队列为空

else return FALSE; //队列非空

}

void clearQueue(CirQueue &Q)

{ //将循环队列 Q 清空

Q.front=Q.rear=0;

}

int queueLength(CirQueue Q)

{ //求循环队列 Q 的长度

return (Q.rear-Q.front+Q.queueSize)%Q.queueSize;

}

Status getFront(CirQueue Q,QElemType &e)

{ //取队列 Q 的队头元素值

if(Q.front==Q.rear) return ERROR;//队空，操作出错

e=Q.base[Q.front];

return OK;

}

Status insertQueue(CirQueue &Q,QElemType e)

{ //在循环队列 Q 的队尾端插入元素 e

if((Q.rear+1)%Q.queueSize==Q.front) return OVERFLOW;//队满，空间溢出，入队失败

Q.base[Q.rear]=e;

Q.rear=(Q.rear+1)%Q.queueSize; //队尾指针后移一个位置

return OK;

}

Status deleteQueue(CirQueue &Q,QElemType &e)

{ //将队列 Q 的队头元素出队并由 e 返回其值

if(Q.front==Q.rear) return ERROR; //队空，操作失败

e=Q.base[Q.front]; //由参数e返回出队元素的值

Q.front=(Q.front+1)%Q.queueSize; //队头指针后移一个位置

return OK;

}

void queueTraverse(CirQueue Q,void(\*visit)(QElemType e))

{ //遍历操作

int i=Q.front;

while(i!=Q.rear)

{ visit(Q.base[i]);

i=(i+1)%Q.queueSize; 、

}

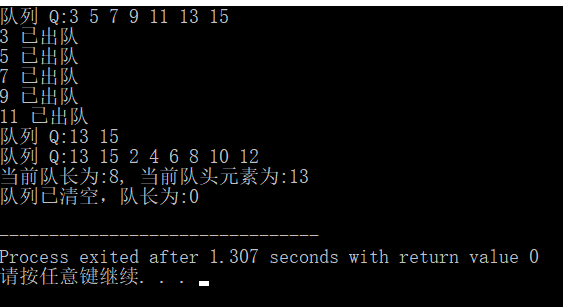
}

2．循环队列测试程序设计

#include <stdio.h> #include <stdlib.h> #include "Status.h" typedef int QElemType; #include "CirQueue.h"

void visitElem(QElemType e){ printf("%d ", e);

3．程序运行结果



}

int main() {

//用于队列测试的主函数

CirQueue Q; QElemType e; int i;

InitCirQueue(Q,11); for(i=1;i<=7; i++)

insertQueue(Q, 2\*i+1);

printf(" 队 列 Q:"); queueTraverse(Q, visitElem);

printf("\n"); for(i=1;i<=5;i++){

deleteQueue(Q,e) ; //出队, 用e 保存出队元素

printf("%d 已出队\n", e);

}

printf("队列 Q:");

queueTraverse(Q, visitElem); printf("\n");

for(1=1;i<=6; i++)

insertQueue(Q,2\*i); //将 2\*i 入队

printf("队列 Q:");

queueTraverse(Q, visitElem) ; //利用遍历输出队列

printf("\n");

getFront(Q,e) ; //取队头元素给 e

printf("当前队长为:%d, 当前队头元素为:%d\n",

queueLength(Q), e );

clearQueue(Q) ; //队列清空

printf("队列已清空，队长为:%d\n",

queueLength(Q) ); system("pause");

return 0;

}