|  |
| --- |
| 哈尔滨工业大学(深圳) |
| **《数据结构》实验报告** |
|  |
| 实验二  栈与队列的应用  学 院: 计算机科学与技术学院   |  |  | | --- | --- | | 姓 名: | 苏亦凡 | | 学 号: | 200111229 | | 专 业: | 计算机科学与技术 | | 日 期: | 2021-03-30 | |

# 一、问题分析

（1）用数组实现栈的基本操作，包括：

将元素压入栈顶

获取并移除栈顶元素

获取栈顶元素

判断栈是否为空

（2） 用栈实现先进先出队列的基本操作，包括：

将元素插入到队列的尾

获取并移除队列头元素

获取队列开头的元素

判断队列是否为空

# 二、详细设计

## 2.1 设计思想

（1）用数组实现栈的基本操作，包括：

将元素压入栈顶：若栈不满，栈顶指针上移，指针所指位置赋值为元素值，返回1；若栈满，返回0.

获取并移除栈顶元素：若栈不为空，获取栈顶指针所指节点的数值域，栈顶指针下移，返回1；若栈为空，返回0.

获取栈顶元素：若栈不为空，获取栈顶指针所指节点的数值域，返回1；若栈为空，返回0.

判断栈是否为空：若栈顶指针值为-1，返回1；否则返回0.

（2） 用栈实现先进先出队列的基本操作，包括：

用栈模拟队列：一个队列用两个栈模拟，分别为stack1与stack2，元素入队即压入stack1，出队则从stack2弹出元素。

将元素插入到队列的尾：若队满，则插入失败，返回0；若队未满，stack1元素<=队列元素，则stack1未满，将元素压入stack1，返回1.

获取并移除队列头元素：若队为空，则获取失败，返回0；若队不为空，则执行一下操作：若stack2不为空，则从stack2弹出元素，返回1；若stack2为空，由队非空可知stack1非空，获取stack1的数组拷贝后逆序依次压入stack2，最后从stack2弹出元素，返回1.

获取队列开头的元素：若队为空，则获取失败，返回0；若队不为空，则执行一下操作：若stack2不为空，则从stack2获取栈顶元素，返回1；若stack2为空，由队非空可知stack1非空，获取stack1的数组拷贝，获取数组最后一元素，返回1.

判断队列是否为空：若stack1与stack2的长度和为0，返回1；否则返回0.

## 2.2 存储结构及操作

(1) 存储结构：栈、队列

#define MaxSize 40 // 数组容量

typedef int DataType;

typedef struct

{

DataType data[MaxSize];

int top; // 栈顶指针

} Stack; // 结构体类型名

typedef struct

{

Stack stack1;

Stack stack2;

} Queue;

(2) 涉及的操作：栈、队列

其中队列的操作全部通过栈的操作完成。

void InitStack(Stack \*S);

int StackEmpty(Stack S);

int GetTop(Stack S, DataType \*e);

int Push(Stack \*S, DataType e);

int Pop(Stack \*S, DataType \*e);

void StackToArray(Stack S, DataType \*seq);

int StackLength(Stack S);

void InitQueue(Queue \*Q);

int GetHead(Queue Q, DataType \*e);

int QueueEmpty(Queue Q);

int EnQueue(Queue \*Q, DataType e);

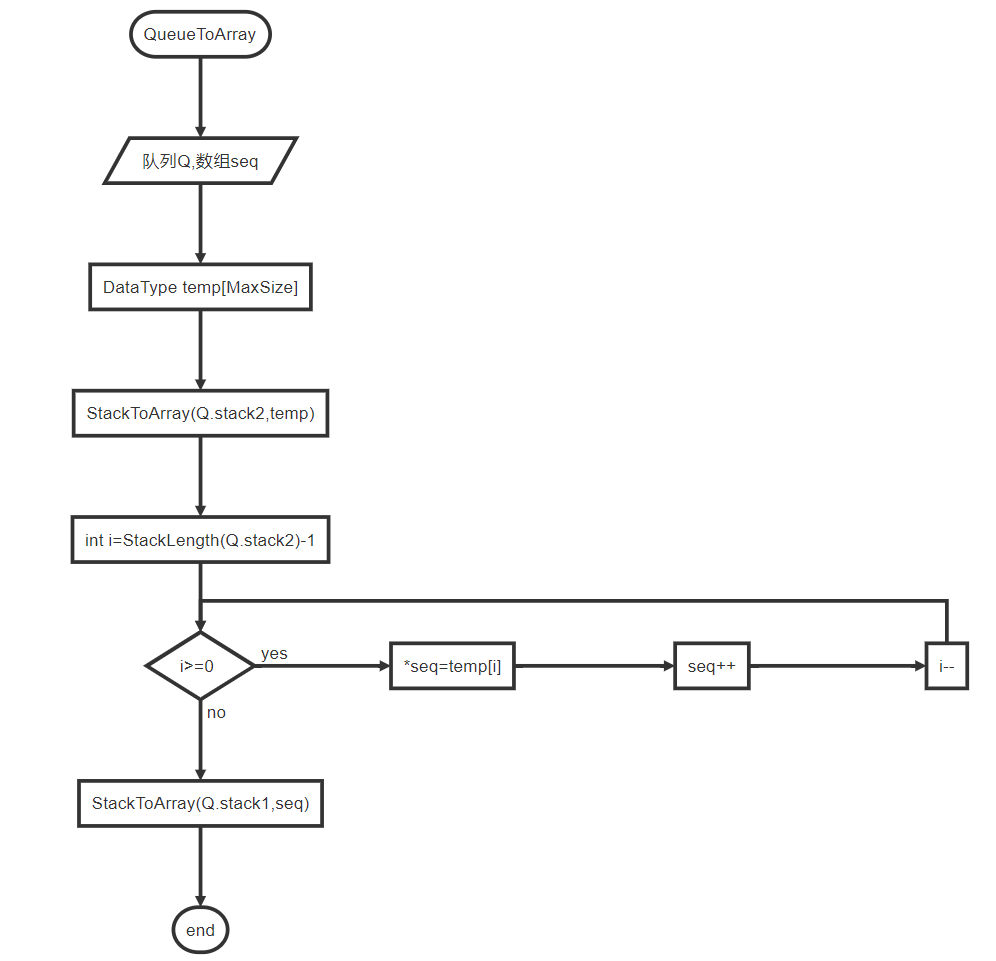
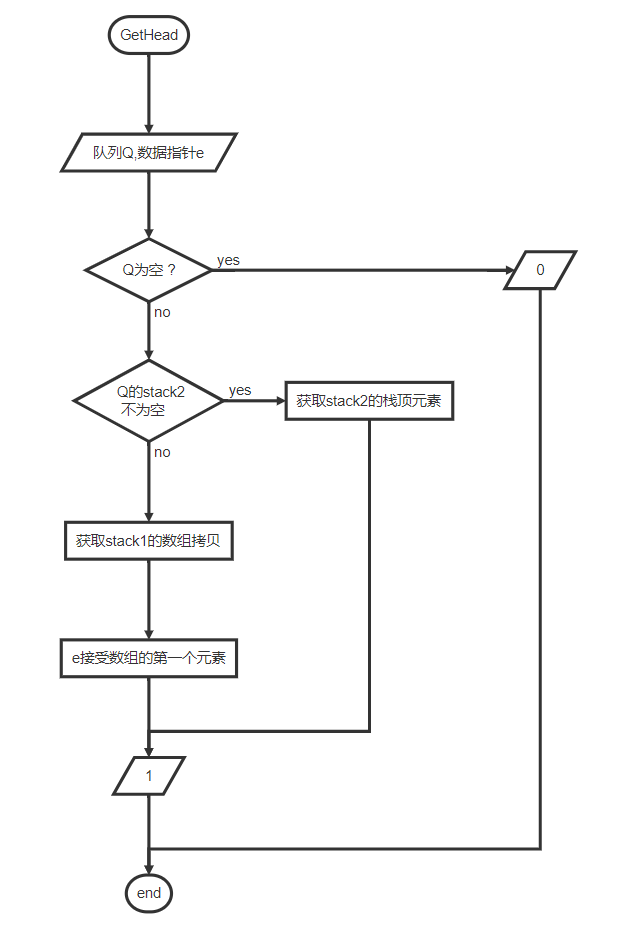
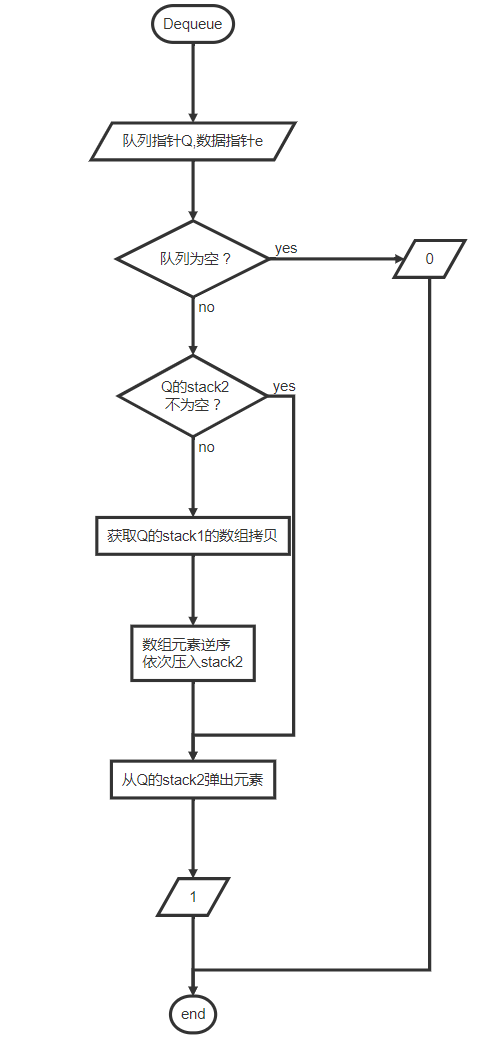
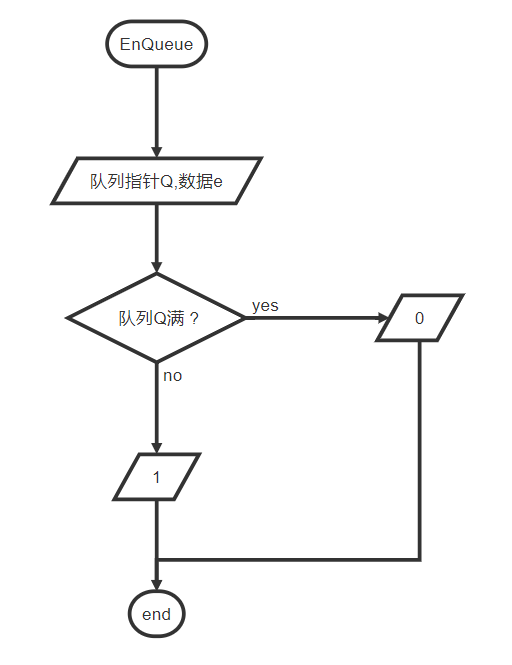
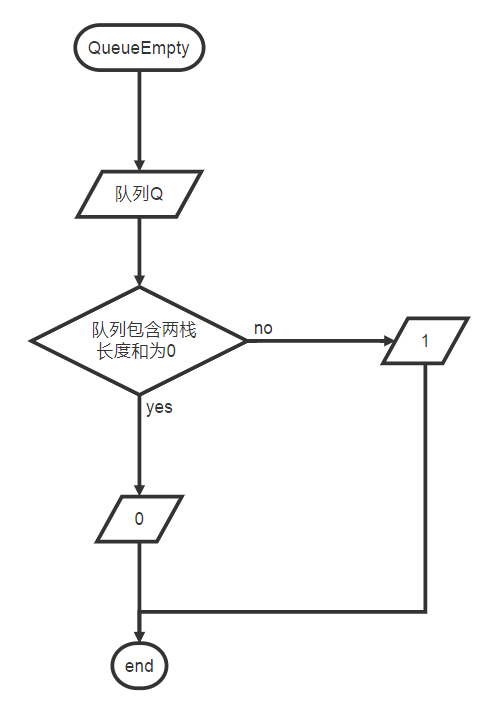
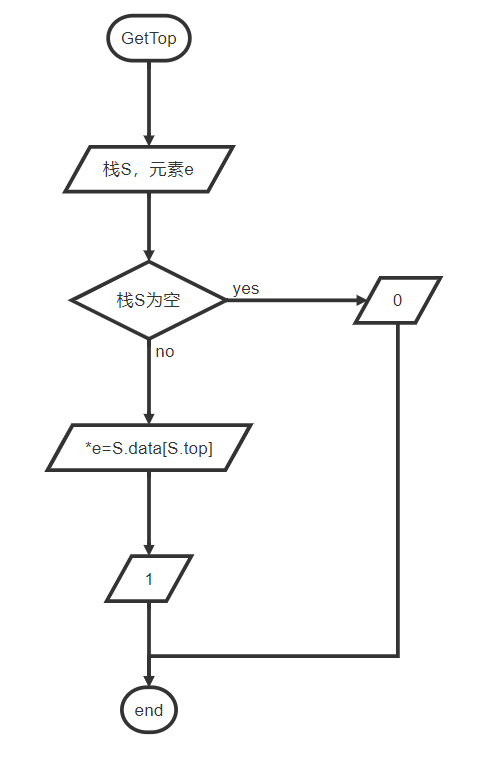
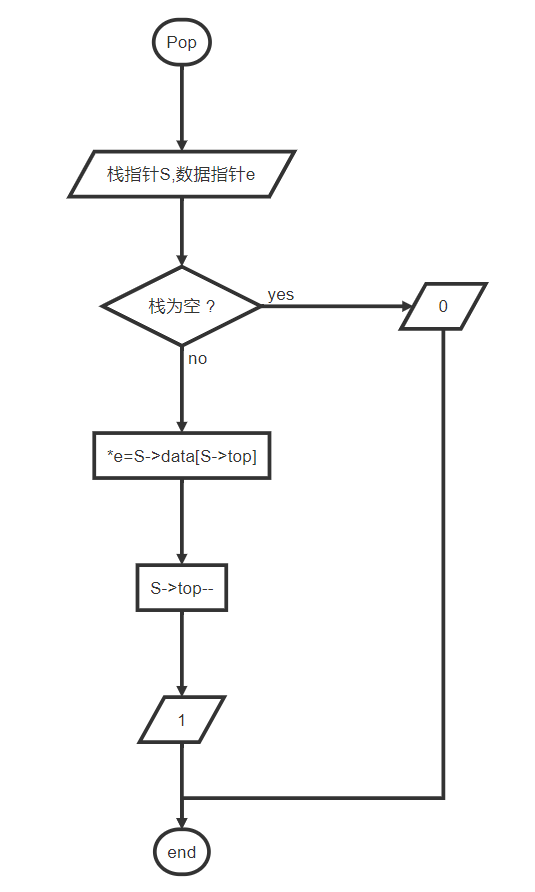
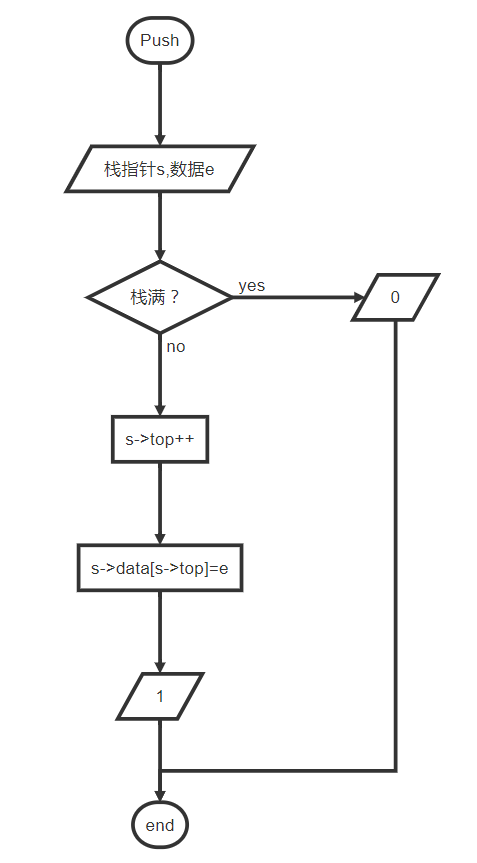
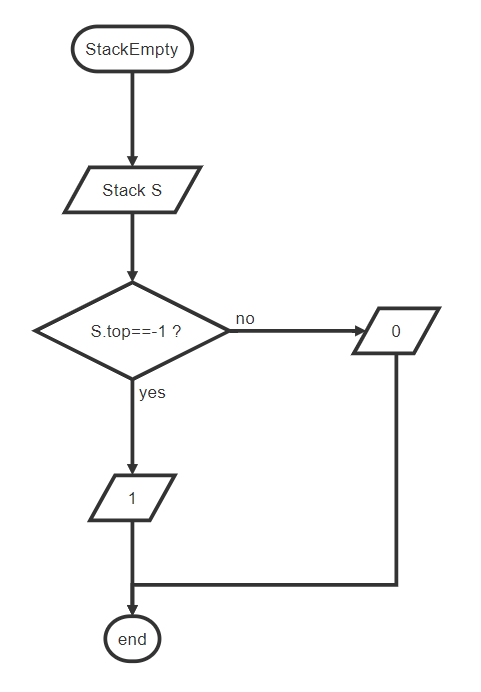
int DeQueue(Queue \*Q, DataType \*e);

void ClearQueue(Queue \*Q);

int QueueLength(Queue Q);

void QueueToArray(Queue Q, DataType \*seq);

## 2.3 程序整体流程



# 三、用户手册

通过文件输入数据。

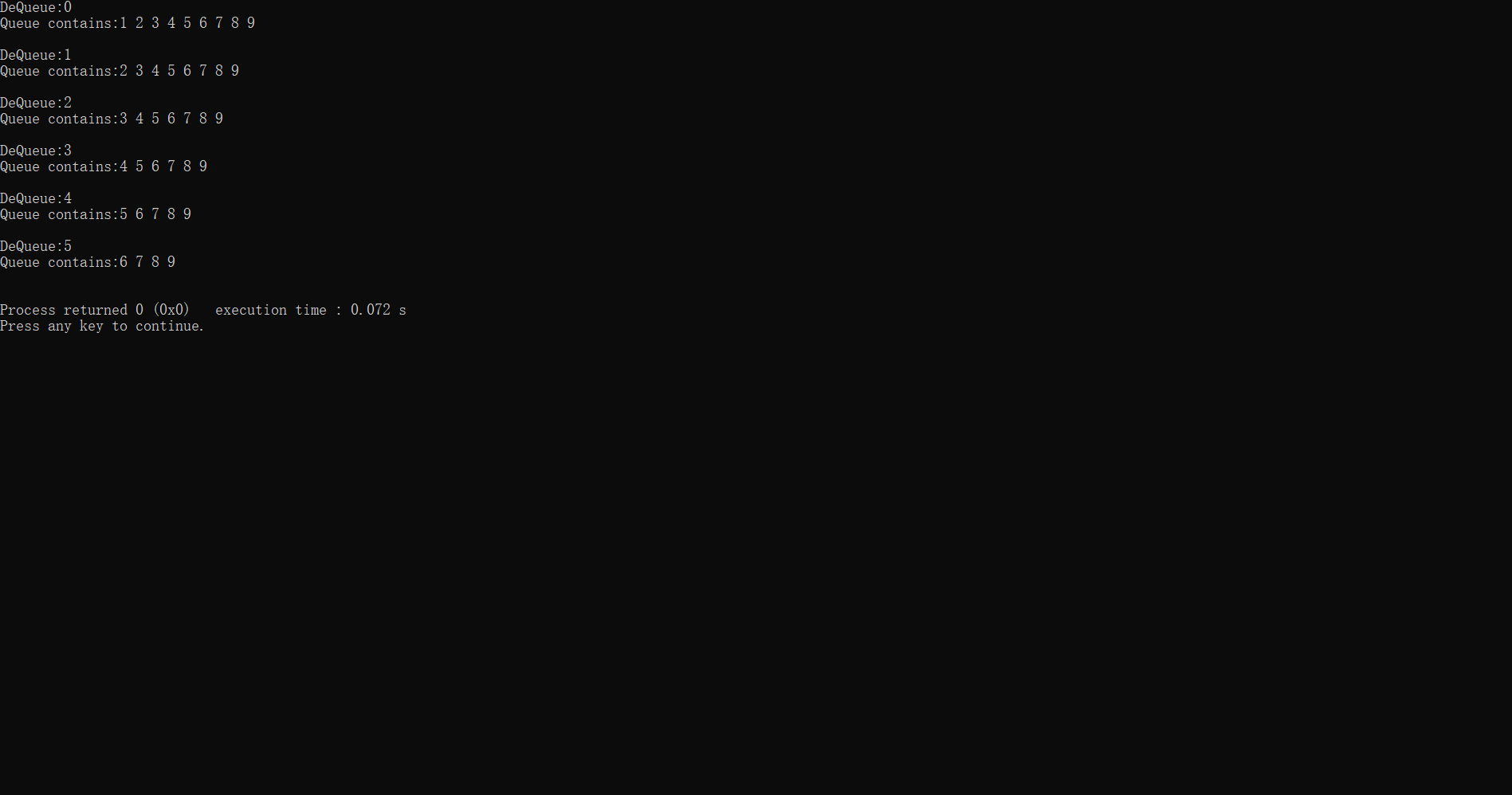
建立输入文件，相对路径为./gradeImport.in

文件中，每行代表一个操作，所有数据都为整数，且使用一个空格隔开。其中，每行第一个参数为 0-7 的整数（0~7的对应关系为：0-Push, 1-Pop, 2-GetTop, 3-StackEmpty, 4-EnQueue, 5-Dequeue, 6-GetHead, 7-QueueEmpty），每行第二个参数为操作的次数（判断栈/队列空 、GetTop 等操作没有这个参数），每行后续所有数据表示入栈/入队列的具体数据，输入的数据数量与第二个参数的次数相同。

# 四、结果







# 五、总结

实验涉及的数据结构为栈与队列，都是特殊的线性结构。在本实验中，通过数组实现栈的基本操作，并使用栈来模拟队列，且只用到栈的基本操作。

本实验最大的难点就是如何设计队列，让两个栈模拟的队列在使用队列的基本操作时效率最高。我在一开始的时候也走了弯路，使得EnQueue与DeQueue的时间复杂度都为O(n)。再仔细思考后，我成功将EnQueue的时间复杂度降为O(1)，DeQueue也有所优化。

本次实验加深了我对栈与队列及其基本操作的理解，重新熟悉了C语言多文件的组织结构，锻炼了我独立思考并自己设计储存结构的能力。