实验二

栈与队列的应用

**一、问题分析**

（1）通过数组来实现栈的基本操作：Push（将元素压入栈顶）、Pop（获取并移除栈顶元素）、GetTOp（获取栈顶元素）、StackEmpty（判断栈顶是否为空）

（2）通过（1）中实现的栈来实现队列的基本操作：EnQueue（将元素插入队列尾）、DeQueue（获取并且移除队列头的元素）、GetHead（获取开头的元素）、QueueEmpty（判断队列是否为空）

**二、详细设计**

**2.1设计思想**

（1）设置一个记号top记录栈顶位置，top原始值设定为-1以便与数组对齐，一个数组seq[]。seq[top]为栈顶元素。当元素压入栈顶时，top+1，seq[top] 等于该元素。当执行Pop时，用e返回seq[top]，将seq[top]置为零，top-1。GetTop用e返回seq[top]实现。执行StackEmpty时，若top=-1，则栈为空，否则栈不为空。

（2）建立两个栈stack1、stack2，当有元素进队列时，直接将其压入栈stack1。当有元素要出队列时，将stack1中除了栈底元素之外的元素依次弹出，压入stack2，再将栈底元素弹出，用e返回值。最后将stack2中的元素依次弹出，压入到stack1。获取队列头元素时，将stack1中除了栈底元素之外的元素依次弹出，压入stack2，再用e返回栈底元素的值。最后将stack2中的元素依次弹出，压入到stack1。获取队列的一个拷贝时，先将stack1中的元素依次弹出，压入stack2。这时将stack2的元素依次弹出即可。

**2.2存储结构及操作**

存储结构：

(1)栈的节点结构体

包含：栈顶指针（int）、储存数据的数组（Datatype）

(2)队列的节点结构体

包含：两个栈节点stack1,stack2

操作：

(1)栈：

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | 参数 | 功能 | 返回值 |
| Push | 栈节点指针，操作数 | 向栈插入一个元素 | 布尔值（是否操作成功） |
| Pop | 栈节点指针，接受弹出栈值的变量指针 | 从栈中弹出一个元素 | 布尔值（是否操作成功） |
| GetTop | 栈节点，接受栈顶元素值的变量指针 | 获取栈顶元素，不删除该元素 | 布尔值（是否操作成功） |
| StackEmpty | 栈节点 | 判断栈是否空 | 布尔值（栈是否为空） |

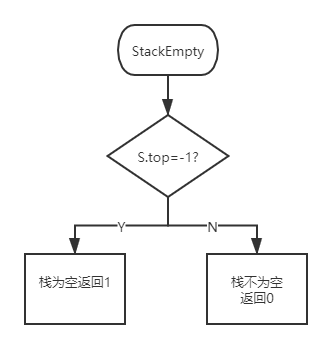
(2)队列：

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | 参数 | 功能 | 返回值 |
| QueueEmpty | 队列节点 | 判断队列是否为空 | 布尔值（队列是否为空） |
| EnQueue | 队列节点指针，插入元素 | 将元素插入队列 | 布尔值（操作是否成功） |
| DeQueue | 队列节点指针，接受出队元素的变量指针 | 从队列中取出一个元素 | 布尔值（操作是否成功） |
| GetHead | 队列节点，接受队列头元素的变量指针 | 获取队列头 | 布尔值（操作是否成功） |
| QueueToArray | 队列节点，数组地址 | 获取队列的一个数组拷贝，顺序为从队列头到队列尾 | 无 |

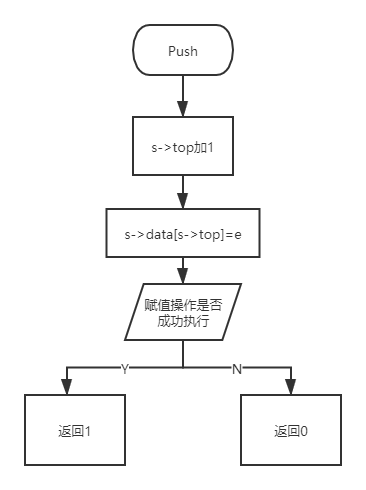
**2.3程序整体流程**

1、栈

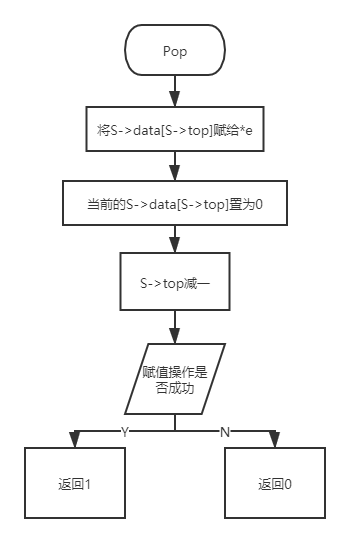
(1)StackEmpty:



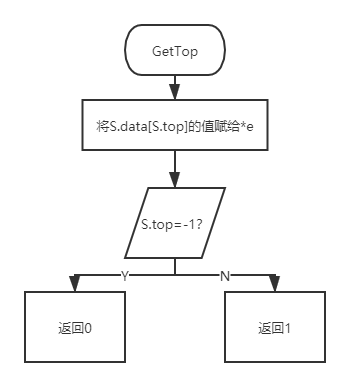
(2)Push



(3)Pop

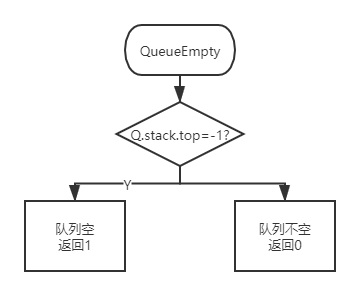


(4)GetTop

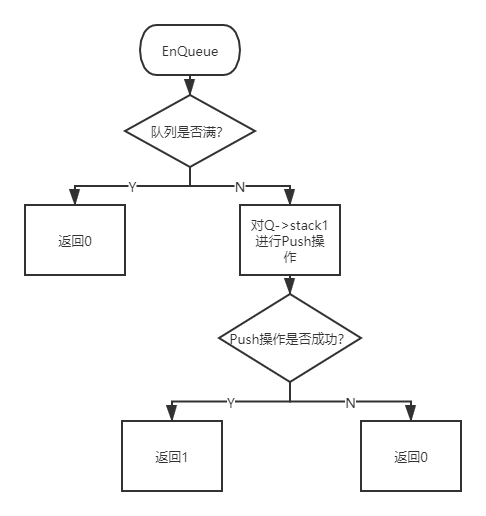


2、队列

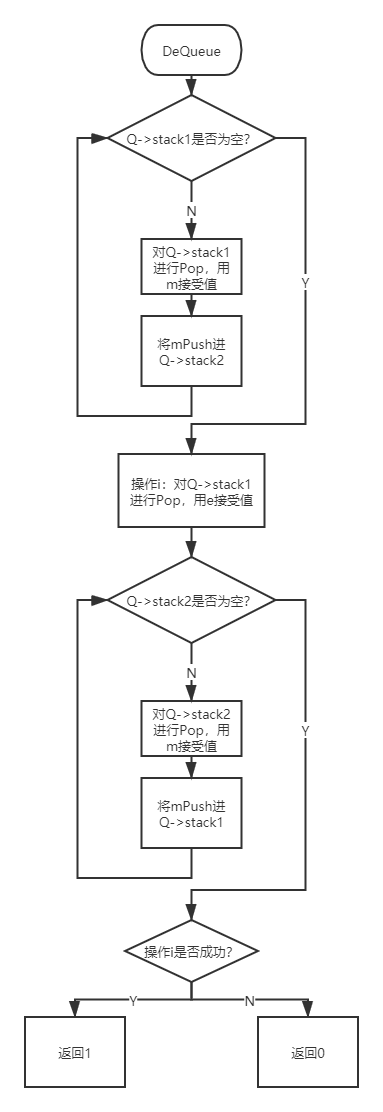
(1)QueueEmpty



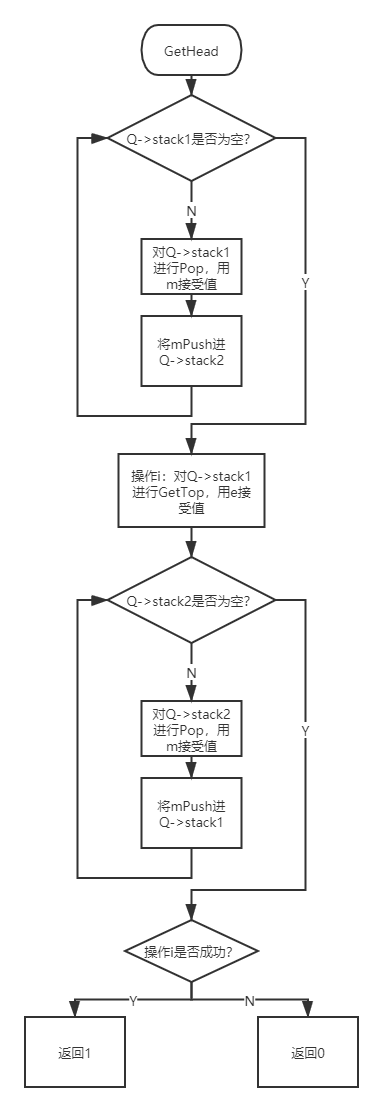
(2)EnQueue



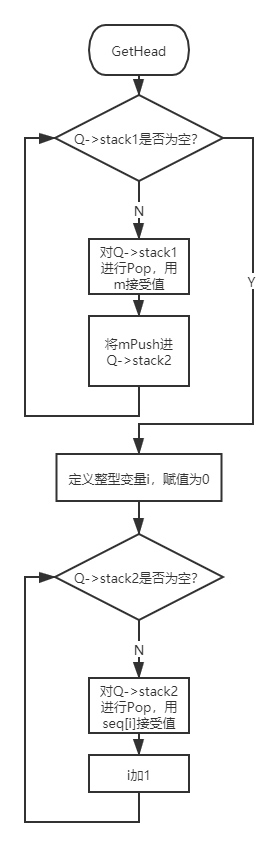
(3)DeQueue



(4)GetHead



(5)QueueToArrey



**三、用户手册**

程序接受数据的方式时从一个名为“test\_stu.in”中的程序读取

输入格式如下：

(1)输入0，进行Push操作

此时需输入多个正整数作为输入栈的数据，输入小于等于0的整数后结束输入。此后会打印出压入栈中的数据。

(2)输入1，进行Pop操作

此时无需输入数据，程序将栈顶元素成功弹出后，将其打印。若Pop操作失败，打印失败提示。

(3)输入2，进行GetTop操作

此时无需输入数据，程序获取栈顶元素后将其打印。若GetTop操作失败，打印失败提示。

(4)输入3，进行StackEmpty操作

此时需输入数据，若栈为空，打印栈为空提示；若栈不为空，打印栈不为空提示。

(5)输入4，进行EnQueue操作

此时需输入多个正整数作为输入队列的数据，输入小于等于0的整数后结束输入。此后会打印出输入队列中的数据。

(6)输入5，进行DeQueue操作

此时无需输入数据，程序将队列头元素成功取出后，将其打印。若DeQueue操作失败，打印失败提示。

(7)输入6，进行GetHead操作

此时无需输入数据，程序获取队列头元素后将其打印。若GetHead操作失败，打印失败提示。

(8)输入7，进行QueueEmpty操作

此时需输入数据，若队列为空，打印队列为空提示；若队列不为空，打印队列不为空提示。

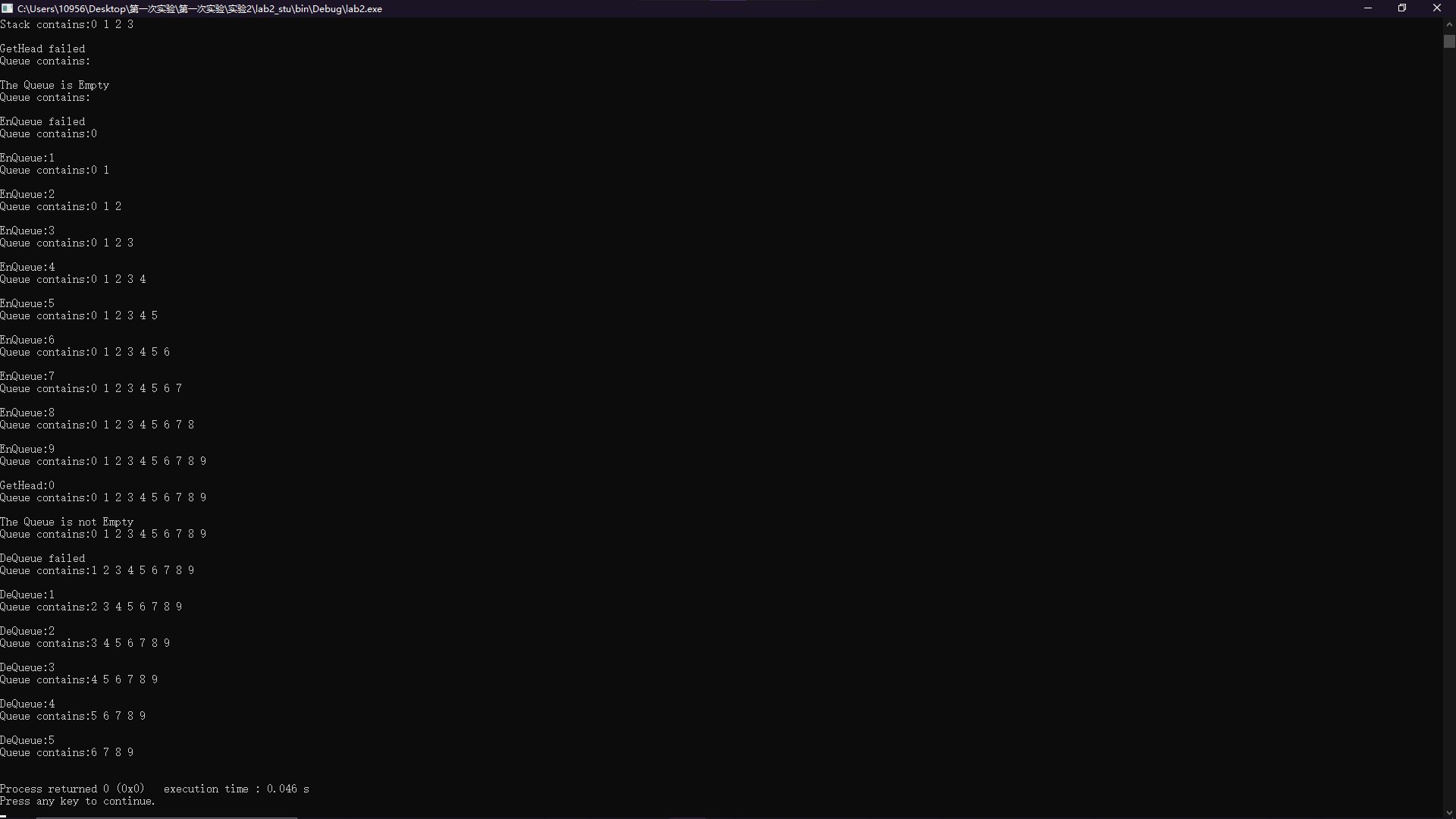
(9)输入-1，结束程序

注：若输入其他值，无效，需重新输入。

程序从头开始读取，运行程序。直到读到文件末尾或者在除了进行Push和EnQueue时读到-1时，终止程序

**四、结果**

****

****

**五、总结**

数据结构：队列，数组，栈

算法：

1. 设置头标志top，用seq[top]代表栈顶，从而实现了用数组实现栈。
2. 设置两个栈stack1，stack2，stack1作为储存栈，stack2作为辅助栈。stack1的栈底作为队列头，栈顶作为队列尾。当对队列头进行操作时，先将stack1中除了栈底元素外其他元素依次弹出，压入stack2中，再对栈底元素进行Pop或者GetTop。从而用栈的LIFO实现了队列的FIFO。

遇到的问题：

进行队列程序的编写时，没有注意哪些变量是指针变量，导致编写错误

收获：

1. 学会了用数组实现栈，用栈实现队列
2. 学会了用ProcessOn绘制流程图