Exercises 3

3-1 Describe the similarities and differences among a stack, a queue and a linear list.

3-2 Give the following operation results (here, *S* is a stack, A and B are data): ①pop (push(S, A)); ②push(S, pop(S)); ③push(S, pop(push(S, B))).

3-3 If Pop(S) = A, explain whether there is an equality: pop(push(S, A)) = push(S, pop(S))

3-4 Write an algorithm, inverse a linear linked list by a stack.

3-5 For an appropriate stack, suppose that the entry sequence is A, B, C, D, E, for getting the following processing sequences, what kind of the operation sequences need to be done by PUSH or POP? If the following sequences can not be gotten, try to explain the reasons: ①A, B, C, D, E; ②B, C, D, E, A; ③E, A, B, C, D; ④E, D, C, B, A.

3-6 Suppose there are two stacks, the shared space space[1.. n], as shown in Fig.1, try to write a program or function for any pushing and popping stack operations: push(x, i) and Pop(i), i=1, 2. Here, i=1 represents the left side of the stack, i=2 represents the right of the stack. Requirements: when the entire space[1.. n] are full, the algorithm will overflow.

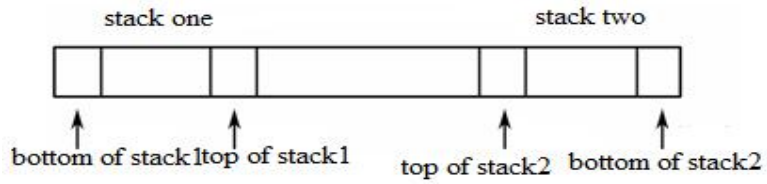

Fig.1 the diagram for problem 3-6

3-7 Using two stacks S1 and S2 to simulate a queue, and write the algorithms of empty queue, enqueue and deqeueue.

3-8 Suppose P is the head pointer of a simple circular list, make the list as a queue, try to write out the enqueue and dequeue algorithms.

3-9 Two-way queue can be inserted and deleted at any one end. Write out the algorithm processes of insertion and deletion at any end of two queue while sequential storage representation.

3-10 Try to use the Boolean functions to represent that the circular queue is empty or full, and the apply the Boolean functions to write the processes of circular queue inserting nodes and deleting nodes.

3-11 The circular queue structure can be achieved with the circular list, in this structure, write the algorithms of enqueue and dequeue.

3-12 Cq[10] is a circular queue, default state is front=rear=0, draw the status changing situations of the queue head and tail indicators after done the following operations. If it can't enqueue, please point out the element, and explain the reasons.

d, e, b, g, y enqueue; b, e a dequeue; i, j, k, l, m enqueue, b a dequeue; n, o, p, q, r enqueue.

3-32 Try to write out the algorithm to calculate the queue length for the circular queue.

3-13 Assuming that the circular linked list with nodes represents the queue, and only set a pointer pointing to the tail element node (note, no head pointer), try to write the corresponding empty queue, enqueue and dequeue algorithm.

3-14 Assuming array cycque[0.. m–1] stores the elements of circular queue, while set the variable rear and quelen, respectively, indicating the location of the tail element and the number of elements in the circular queue. try to give out the full condition of queue, and write out the corresponding enqueue and dequeue algorithms.

3-15 设有一个栈，元素进栈的次序为 a, b, c。问经过栈操作后可以得到哪些输出序列？

3-16 循环队列的优点是什么?如何判断它的空和满?

3-17 设有一个静态顺序队列，向量大小为 MAX，判断队列为空的条件是什么？队列满的条件是什么？

3-18 设有一个静态循环队列，向量大小为 MAX，判断队列为空的条件是什么？队列满的条件是什么？

3-19 利用栈的基本操作，写一个返回栈 S 中结点个数的算法 int StackSize(SeqStack S) ，并说明 S 为何不作为指针参数的算法?

3-20 一个双向栈 S 是在同一向量空间内实现的两个栈，它们的栈底分别设在向量空间的两端。试为此双向栈设计初始化 InitStack(S) ，入栈 Push(S,i,x)，出栈 Pop(S,i,x) 算法，其中 i 为 0 或 1 ，用以表示栈号。

3-21 设 Q[0,6]是一个静态顺序队列，初始状态为 front=rear=0，请画出做完下列操作后队列的头尾指针的状态变化情况，若不能入对，请指出其元素，并说明理由。

a, b, c, d 入队

a, b, c 出队

i , j , k , l , m 入队

d, i 出队

n, o, p, q, r 入队

3-22 假设 Q[0,5]是一个循环队列，初始状态为 front=rear=0，请画出做完下列操作后队列的头尾指针的状态变化情况，若不能入对，请指出其元素，并说明理由。

d, e, b, g, h 入队

d, e 出队

i , j , k , l , m 入队

b 出队

n, o, p, q, r 入队

3-23 If the queue is not set header node, as shown in Fig.2, and make front=rear=null,

representing the queue is empty, try to write algorithms to achieve the five basic operations of queue (setting empty, enqueue, dequeue, judging empty and reading head).

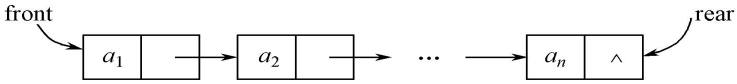

Fig.2 the diagram for problem 3-23

3-24 Fig.3 shows the scheduling of a railway station, it is a stack structure. There are three cars named No.1, No.2, No.3 on the right track. Each car can be brought into the station, and can be towed away at any time. If the cars No.1, No.2, No.3 were brought in turn, how many possible orders of going out from the side of the station orbit? Please write them specifically. What is the situation, if there are 4 four cars named No.1, No.2, No.3, No.4 going into the stack on the track in turn?

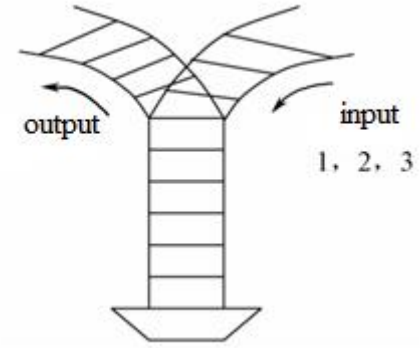

Fig.3 the diagram for problem 3-24

3-25 Ackermann function A (m, n) is defined as follows:

$$A(m,n)=\begin{cases} n+1 & \text{If } m=0 \\ A(m-1) & \text{If } n=0 \\ A(m-1,A(m,n-1)) & \text{others} \end{cases}$$

Write a recursive process of calculating this function, and then write a non-recursive algorithm to calculate the Ackermann function.