

## Lesson4--栈和队列

### 【本节目标】

- 1.栈
- 2.队列
- 3.栈和队列面试题

### 1.栈

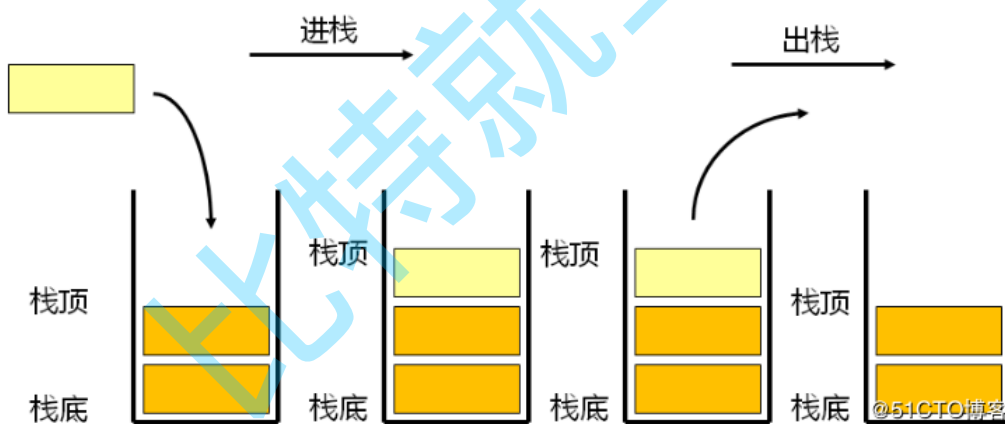
#### 1.1栈的概念及结构

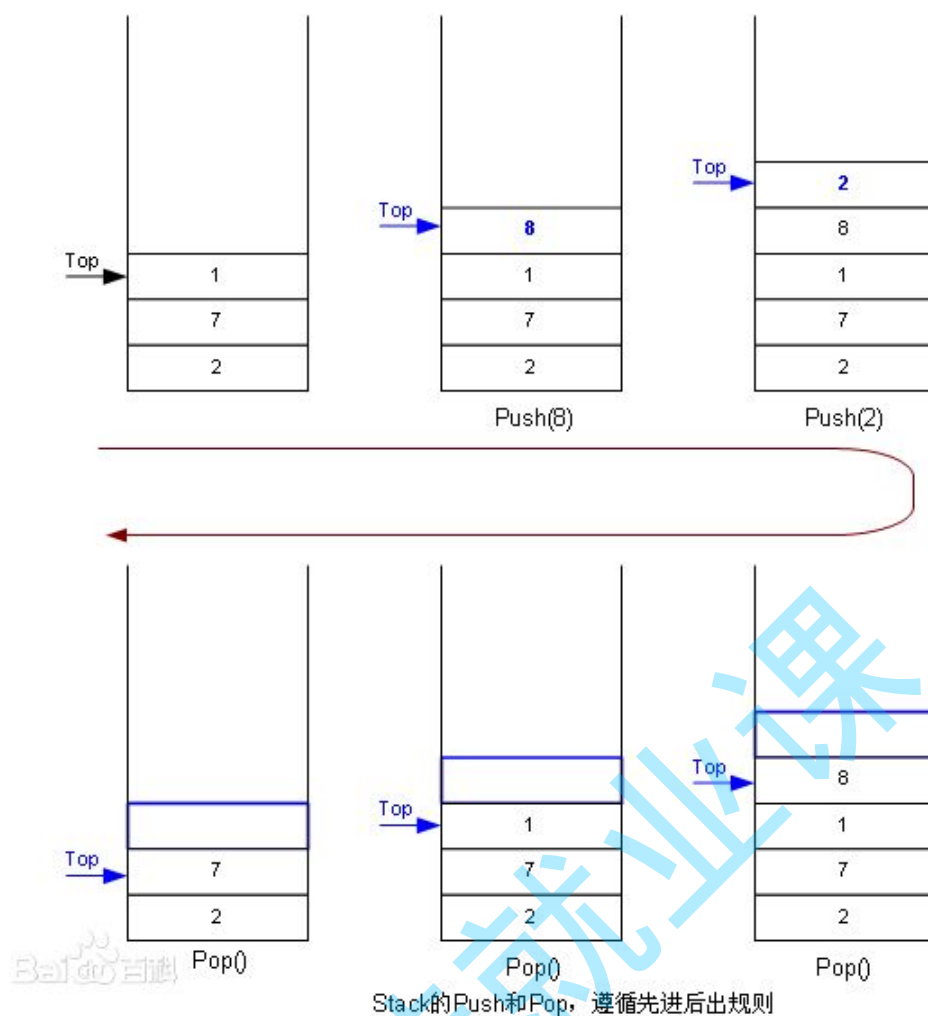
栈：一种特殊的线性表，其只允许在固定的一端进行插入和删除元素操作。进行数据插入和删除操作的一端称为栈顶，另一端称为栈底。栈中的数据元素遵守后进先出LIFO（Last In First Out）的原则。

压栈：栈的插入操作叫做进栈/压栈/入栈，入数据在栈顶。

出栈：栈的删除操作叫做出栈。出数据也在栈顶。

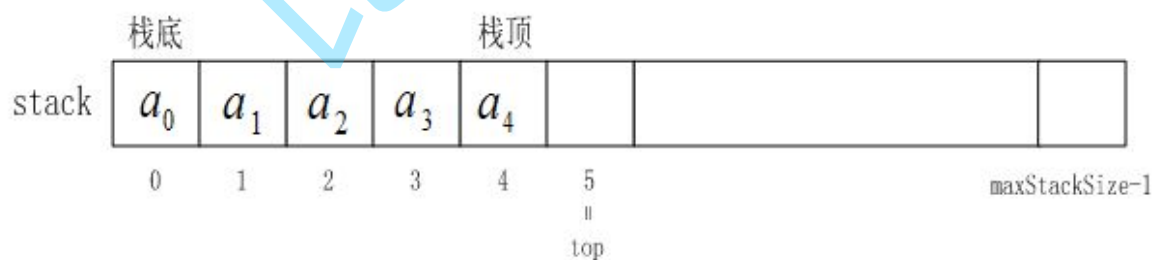
— 后进先出 ( Last In First Out )





## 1.2 栈的实现

栈的实现一般可以使用**数组或者链表实现**，相对而言数组的结构实现更优一些。因为数组在尾上插入数据的代价比较小。



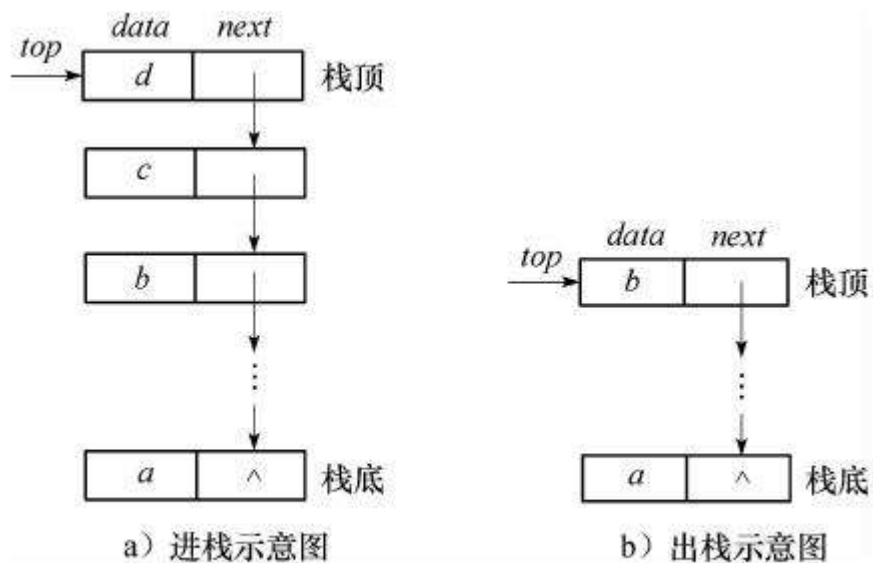


图 2-10 链栈的进栈示意图

```

1  // 下面是定长的静态栈的结构，实际中一般不实用，所以我们主要实现下面的支持动态增长的栈
2  typedef int STDataType;
3  #define N 10
4  typedef struct Stack
5  {
6      STDataType _a[N];
7      int _top; // 栈顶
8  }Stack;
9
10 // 支持动态增长的栈
11 typedef int STDataType;
12 typedef struct Stack
13 {
14     STDataType* _a;
15     int _top; // 栈顶
16     int _capacity; // 容量
17 }Stack;
18 // 初始化栈
19 void StackInit(Stack* ps);
20 // 入栈
21 void StackPush(Stack* ps, STDataType data);
22 // 出栈
23 void StackPop(Stack* ps);
24 // 获取栈顶元素
25 STDataType StackTop(Stack* ps);
26 // 获取栈中有效元素个数
27 int StackSize(Stack* ps);
28 // 检测栈是否为空，如果为空返回非零结果，如果不为空返回0
29 int StackEmpty(Stack* ps);
30 // 销毁栈
31 void StackDestroy(Stack* ps);

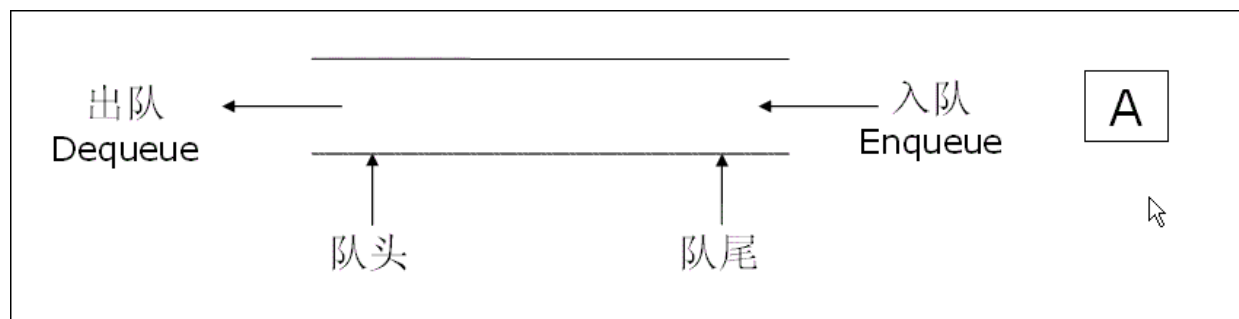
```

## 2.队列

- 1、排队，保持绝对公平性
- 2、广度优先遍历 BFS

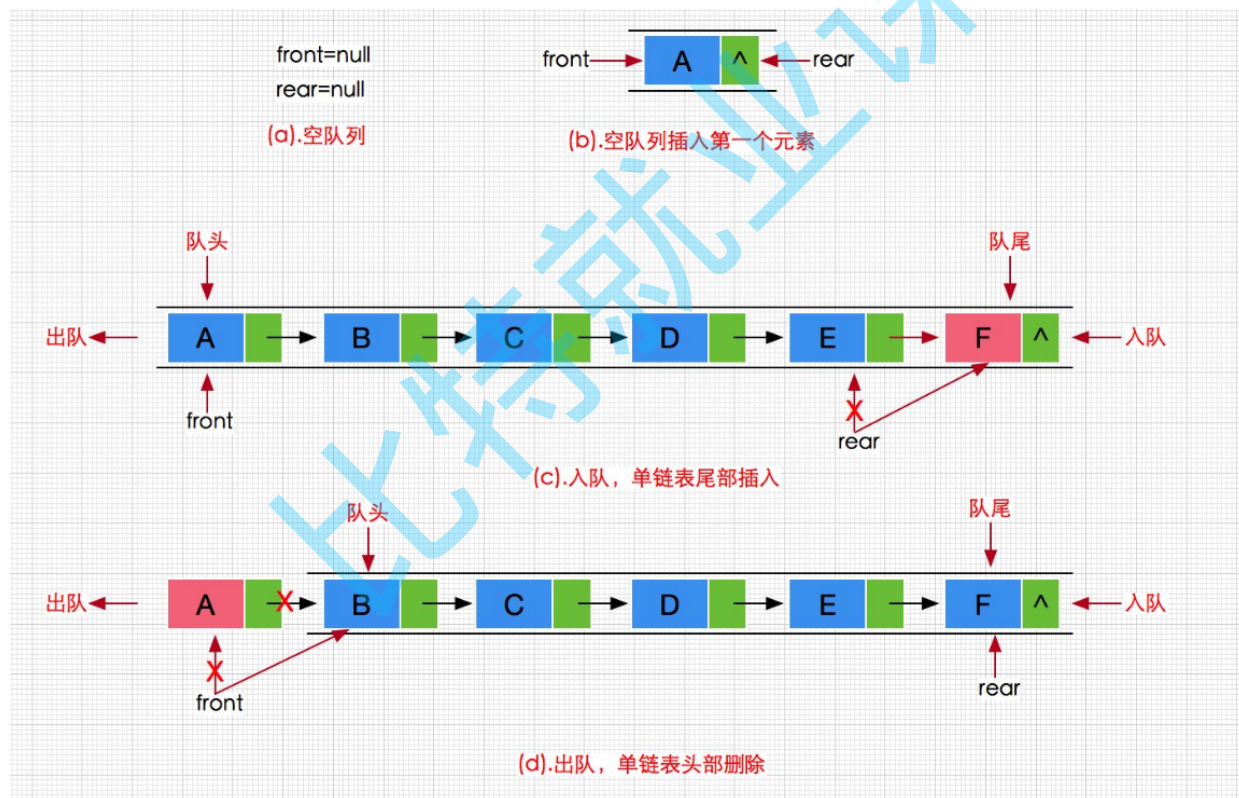
### 2.1队列的概念及结构

队列：只允许在一端进行插入数据操作，在另一端进行删除数据操作的特殊线性表，队列具有先进先出 FIFO(First In First Out) 入队列：进行插入操作的一端称为**队尾** 出队列：进行删除操作的一端称为**队头**



### 2.2队列的实现

队列也可以数组和链表的结构实现，使用链表的结构实现更优一些，因为如果使用数组的结构，出队列在数组头上出数据，效率会比较低。



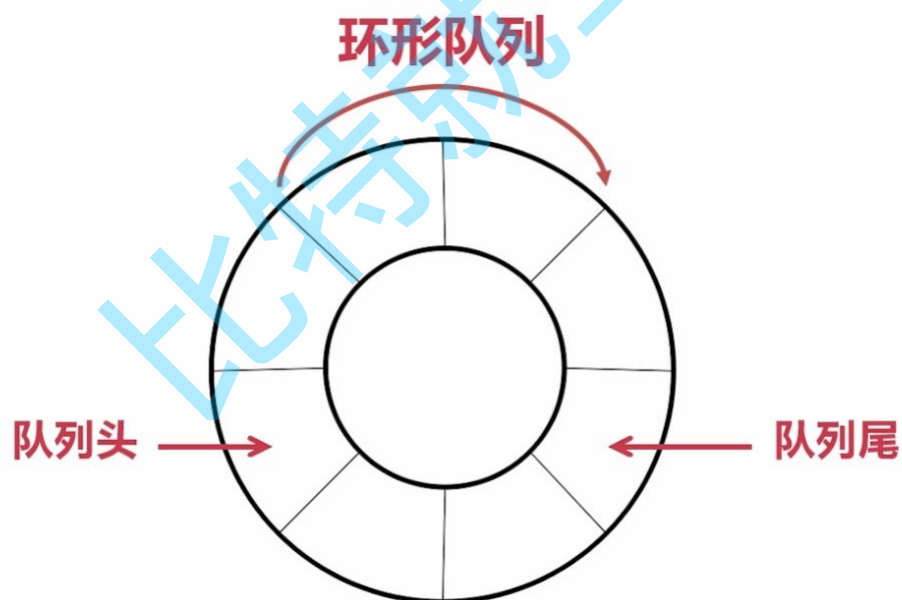
```
1 // 链式结构：表示队列
2 typedef struct QListNode
3 {
4     struct QListNode* _pNext;
5     QDataType _data;
6 }QNode;
7
8 // 队列的结构
9 typedef struct Queue
```

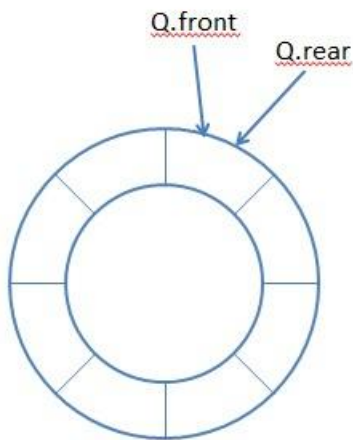
```

10 {
11     QNode* _front;
12     QNode* _rear;
13 }Queue;
14
15 // 初始化队列
16 void QueueInit(Queue* q);
17 // 队尾入队列
18 void QueuePush(Queue* q, QDataType data);
19 // 队头出队列
20 void QueuePop(Queue* q);
21 // 获取队列头部元素
22 QDataType QueueFront(Queue* q);
23 // 获取队列队尾元素
24 QDataType QueueBack(Queue* q);
25 // 获取队列中有效元素个数
26 int QueueSize(Queue* q);
27 // 检测队列是否为空, 如果为空返回非零结果, 如果非空返回0
28 int QueueEmpty(Queue* q);
29 // 销毁队列
30 void QueueDestroy(Queue* q);

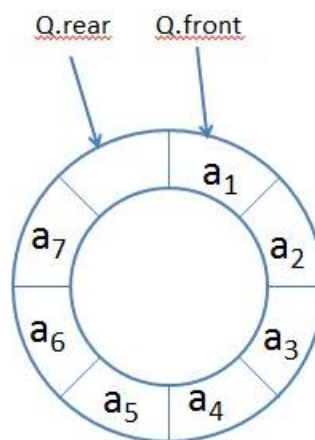
```

另外扩展了解一下, 实际中我们有时还会使用一种队列叫循环队列。如操作系统课程讲解生产者消费者模型时就会使用循环队列。环形队列可以使用数组实现, 也可以使用循环链表实现。





(a) 空的循环队列



(b) 满的循环队列

为了使用  $Q.rear=Q.front$  来区别是队空还是队满，我们常常认为出现左图时的情况即为队满的情况，此时：  
 $rear+1=front$

[http://blog.csdn.net/zhang\\_xinxu](http://blog.csdn.net/zhang_xinxu)

### 3. 栈和队列面试题

1. 括号匹配问题。 [O链接](#)
2. 用队列实现栈。 [O链接](#)
3. 用栈实现队列。 [O链接](#)
4. 设计循环队列。 [O链接](#)

### 4. 概念选择题

#### 选择题

- 1 1. 一个栈的初始状态为空。现将元素1、2、3、4、5、A、B、C、D、E依次入栈，然后再依次出栈，则元素出栈的顺序是（ ）。
- 2 A 12345ABCDE
- 3 B EDCBA54321
- 4 C ABCDE12345
- 5 D 54321EDCBA
- 6
- 7 2. 若进栈序列为 1,2,3,4，进栈过程中可以出栈，则下列不可能的一个出栈序列是（）
- 8 A 1,4,3,2
- 9 B 2,3,4,1
- 10 C 3,1,4,2
- 11 D 3,4,2,1
- 12
- 13 3. 循环队列的存储空间为  $Q(1:100)$ ，初始状态为  $front=rear=100$ 。经过一系列正常的入队与退队操作后， $front=rear=99$ ，则循环队列中的元素个数为（）
- 14 A 1
- 15 B 2
- 16 C 99
- 17 D 0或者100
- 18
- 19 4. 以下（）不是队列的基本运算？
- 20 A 从队尾插入一个新元素
- 21 B 从队列中删除第i个元素
- 22 C 判断一个队列是否为空
- 23 D 读取队头元素的值

- 24
- 25 5.现有一循环队列，其队头指针为front，队尾指针为rear；循环队列长度为N。其队内有效长度为？（假设队头不存放数据）
- 26 A (rear - front + N) % N + 1
- 27 B (rear - front + N) % N
- 28 C ear - front) % (N + 1)
- 29 D (rear - front + N) % (N - 1)

### 答案

- 1 1.B
- 2 2.C
- 3 3.D
- 4 4.B
- 5 5.B

比特就业课