算法整理

1. 单调栈

单调栈是不断往栈内放东西，根据从大到小或者从小到大的原则来规划栈。当栈内的元素不符合单调关系时候，对栈进行操作。常见的操作是往栈内放index，比较完后，进行一系列操作。如果符合栈内的单调关系，直接入栈。

例题：1475，739

1. 优先队列

优先队列容器与队列一样，只能从队尾插入元素，从队首删除元素。但是它有一个特性，就是队列中最大的元素总是位于队首，所以出队时，并非按照先进先出的原则进行，而是将当前队列中最大的元素出队。这点类似于给队列里的元素进行了由大到小的顺序排序。元素的比较规则默认按元素值由大到小排序，可以重载“<”操作符来重新定义比较规则。

声明方式：

1、普通方法：

priority\_queue<int>q;   　　　　　　　　　　　　  //通过操作，按照元素从大到小的顺序出队

priority\_queue<int,vector<int>,greater<int>>q;  　　//通过操作，按照元素从小到大的顺序出队

2、自定义优先级：

struct cmp {

　　operator bool ()(int x, int y)

　　{

　　　　 return　x > y;　　 // x小的优先级高       //也可以写成其他方式，如： return p[x] > p[y];表示p[i]小的优先级高

　　}

};

priority\_queue<int, vector<int>, cmp> q;    //定义方法

优先队列也可以用队列或者双向队列deque进行维护，选择的时候根据实际需求进行选择。

例题：1696

1. 栈实现队列

栈实现队列，需要两个栈，一个栈用来作为输入栈，一个栈作为输出栈。输入栈用于接收输入，输出栈用于输出。

核心思路就是当输入栈要将元素倾倒至输出栈中，并且一定要注意，此时输出栈必须非空

1. 队列实现栈

使用两个队列，进行交换，两个队列只有一个队列有元素，有元素的队列便是输出队列。核心思想就是不断交换两个的队列的元素。一个空的队列接收数据，然后将另一个非空的队列中的元素倒到接收数据的队列，不断倾倒即可。

1. 队列的最大值

使用deque和queue进行操作，queue负责接收数据，deque保存最大值。Deque的操作如下，当要push的value大于deque.back()时候，循环pop()，直到遇到比value大的数，或者deque为空的时候，然后放入该元素。如果比deque.back()小，直接放入。也就是始终维护deque从大到小。所以队首就是最大值。