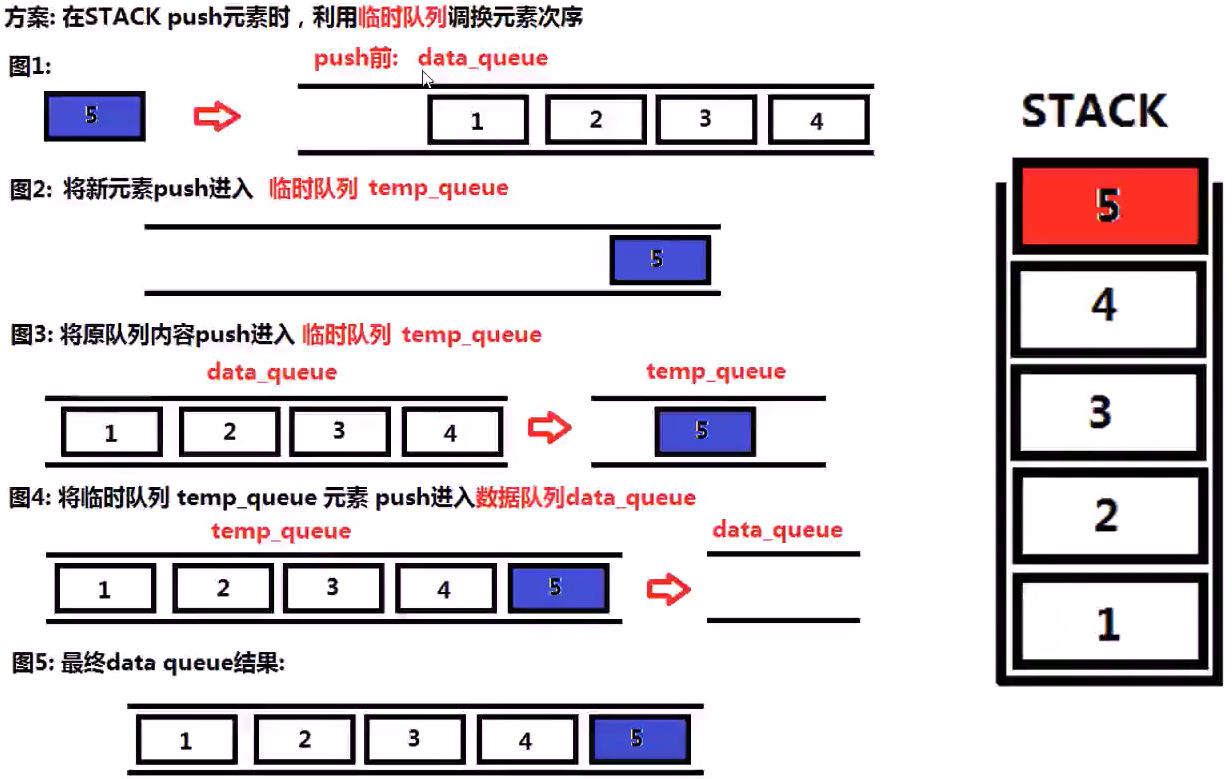
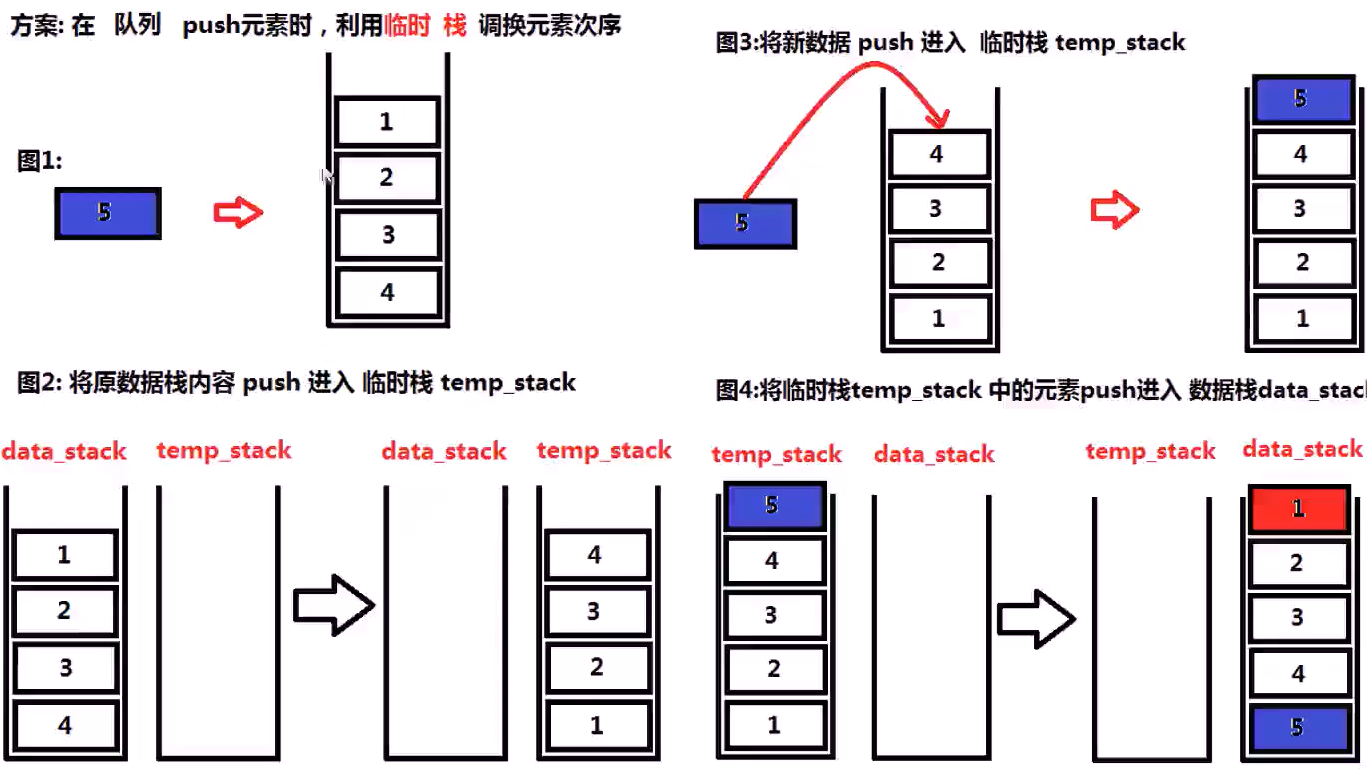
* 栈、队列、堆
  + 用队列实现栈

思路：实现push(x)：用一个临时队列，将x放入临时队列后，再将存储队列中的数据放入临时队列中，最后把临时队列的数据放回存储队列中



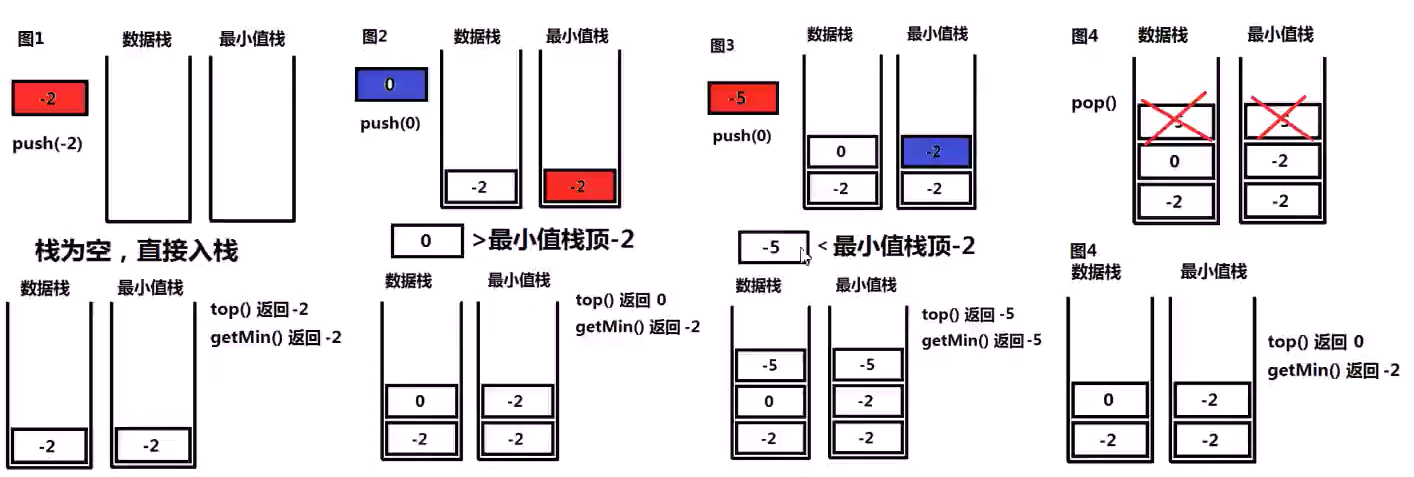
* + 用栈实现队列

思路：实现push(x)：用一个临时栈，将存储栈中的数据放入临时栈中，再将x放入临时栈中，最后把临时栈的数据放回存储栈中



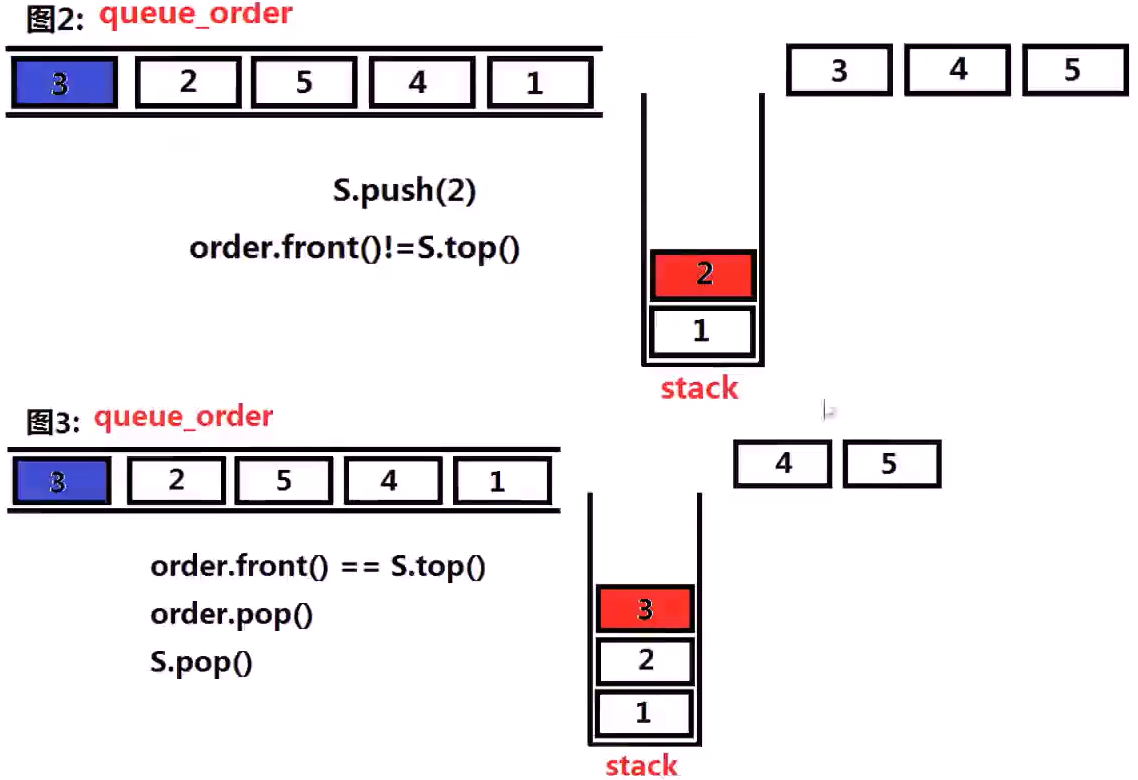
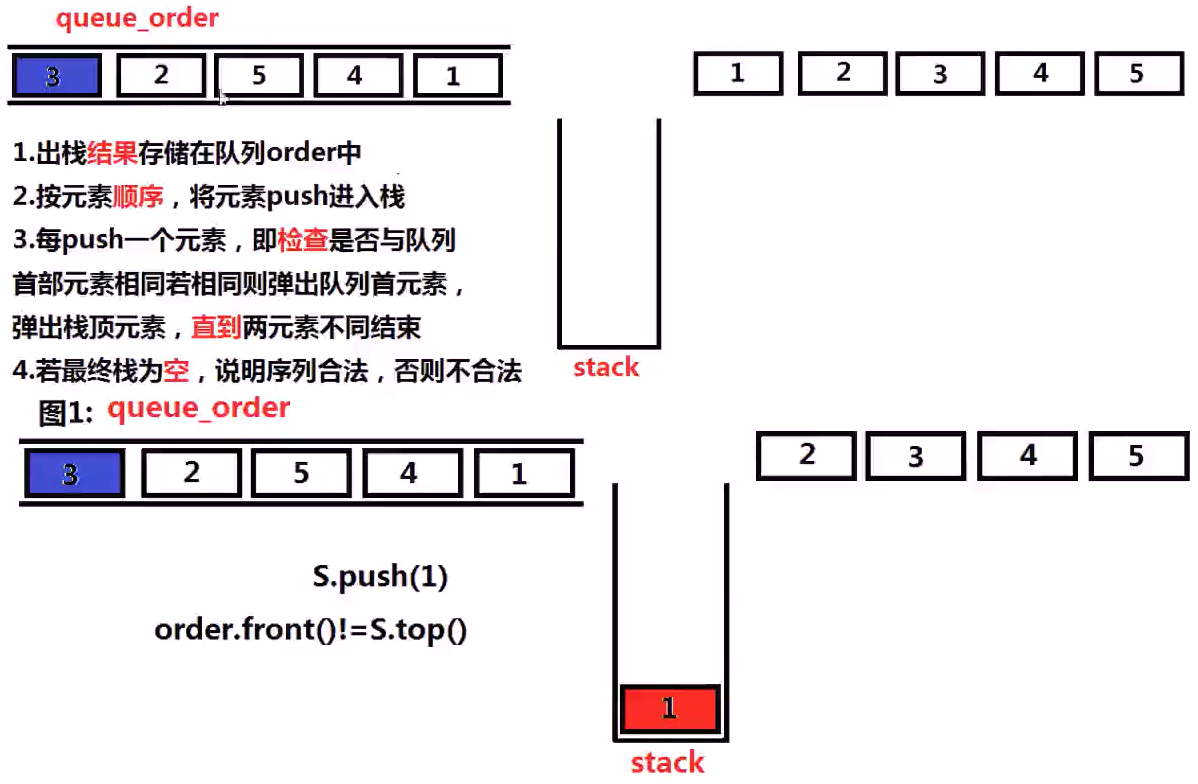
* + 包含min函数的栈，获取栈中最小元素的复杂度为O(1)

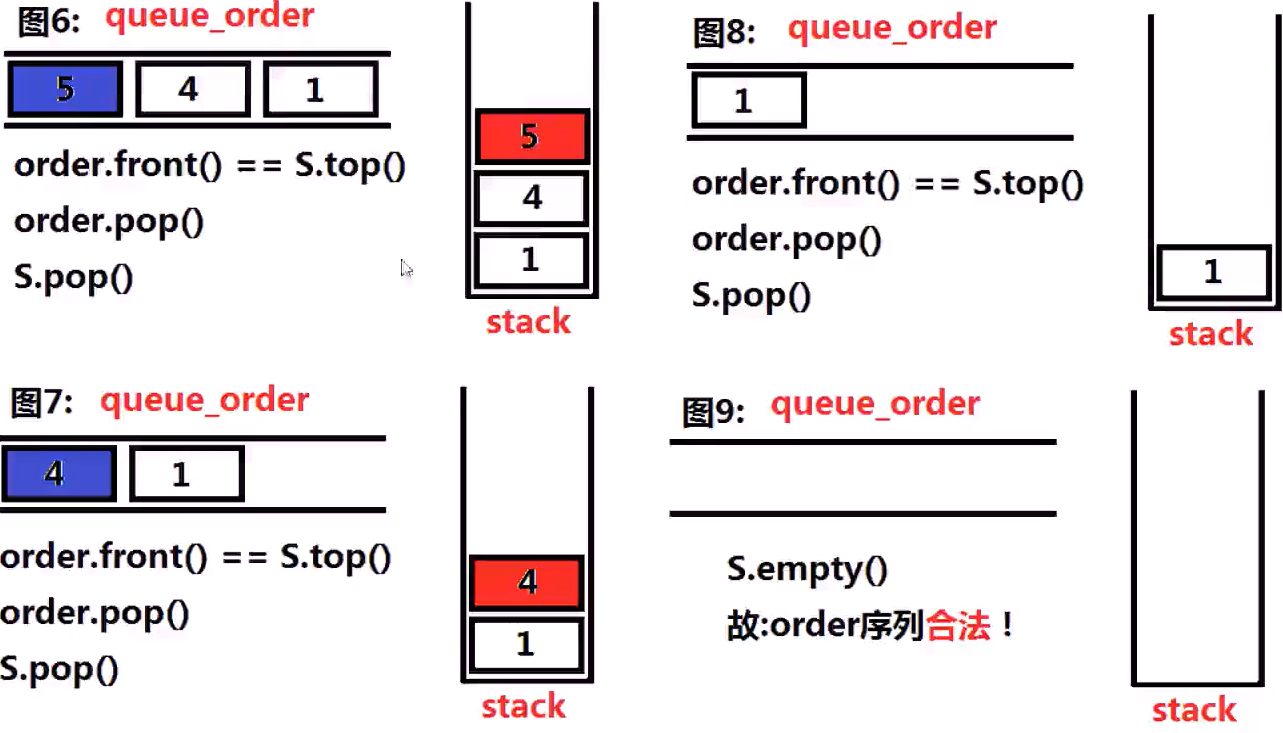
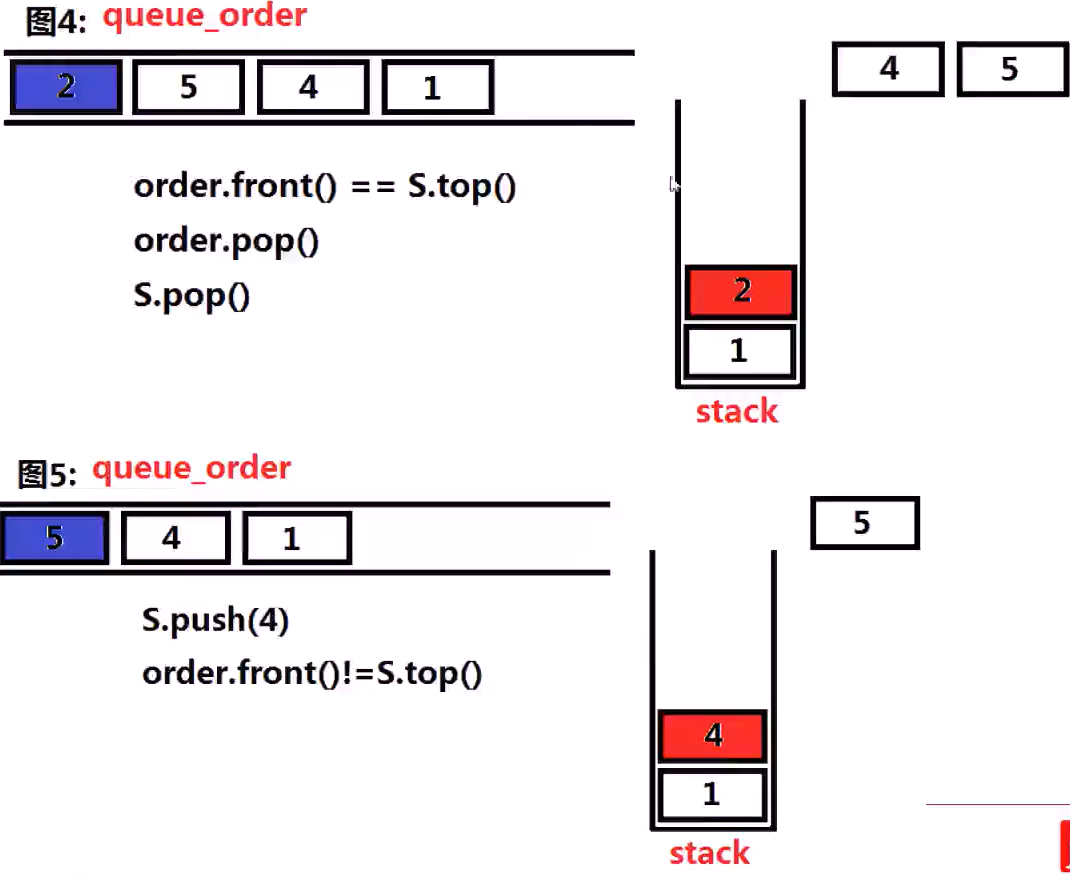
思路：用一个 存储各个状态时最小值的栈，获取最小值即获取最小值栈的栈顶——改写push(x)方法，当x<cur\_min时，将x放入最小值栈，cur\_min更新为x，否则将cur\_min放入最小值栈；改写pop()方法，存储栈每次pop()，最小值栈也随之pop()。这样两个栈中的数据量保持一致，且最小值栈的栈顶为存储栈当前状态对应的最小值！



* + 判断1~n的递增序列对应的一个序列是否是合法的出栈序列

思路：将递增序列的首元素放入一个栈中，用一个指针指向待判断的序列的首元素，当栈顶元素！=指针指向元素时继续压栈，否则弹栈并将指针后移一位；当指针遍历完待测序列的同时栈为空则合法，否则不合法——**拓展：n个数对应的合法出栈序列个数f(n)=C(2n,n)-C(2n,n+1)，其中f(0)=0,f(1)=1（卡塔兰数）**





* + 简单计算器：设计一个计算器，输入一个字符串存储的数学表达式，可以计算包括(,),+,-四种符号的数学表达式，输入的数学表达式字符串保证是合法的。输入的数学表达式中存在空格

思路：用两个栈：**运算数栈**，**运算符栈**；<https://leetcode-cn.com/problems/basic-calculator/solution/ji-ben-ji-suan-qi-by-leetcode/>

* + 数组中第K大的数

思路：维护一个K大小的最小堆，堆中元素个数<K时，新元素直接进入堆；否则当堆顶<新元素时，弹出堆顶并将新元素加入堆中。最后得到的堆顶元素即为结果

* + 寻找中位数：设计一个数据结构，可以动态维护一组数据，且支持如下操作：添加元素；返回中位数

思路：动态维护一个最大堆和一个最小堆，最大堆堆顶<最小堆堆顶，两者各存放一半的数据

* + - 添加元素add(x)：
      * 最大堆大小=最小堆大小：x<最大堆堆顶时，添加到最大堆中；x>最小堆堆顶时，添加到最小堆中
      * 最大堆大小=最小堆大小 + 1：x<最大堆堆顶时，将最大堆堆顶弹出并放入最小堆，并把x放入最大堆；x>最大堆堆顶时，将x放入最小堆中
      * 最大堆大小+1=最小堆大小：x<最小堆堆顶时，添加到最大堆中；x>最小堆堆顶时，将最小堆堆顶弹出并放入最大堆，并把x放入最小堆
    - 获取中位数：若两个堆的大小一样，中位数为两堆顶中间值；否则中位数为较大的堆的堆顶元素