数据结构(下) - 堆与栈难题精讲

主讲 侯卫东

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: ninechapter
微博: http://www.weibo.com/ninechapter
知乎: http://zhuanlan.zhihu.com/jiuzhang
官网: http://www.jiuzhang.com

今日大纲

九章算法

堆 Heap

- Data Stream Median
- Sliding Window Median
- Trapping Rain Water II

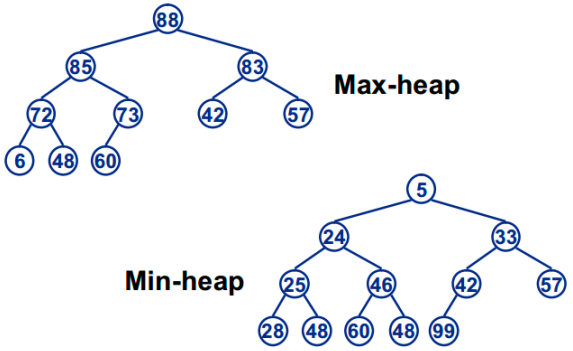
栈 Stack

- Min Stack
- Decode Strings
- Largest Rectangle in Histogram
- Maximum Rectangle
- Max Tree

堆 Heap

支持操作 O(1) Min/Max / log(N) Push / log(N) Pop

python: heapq
Java: PriorityQueue
C++: priority_queue



Find Median from Data Stream

https://www.lintcode.com/problem/data-stream-median/

https://www.jiuzhang.com/solutions/data-stream-median/

LintCode 81

九章算法

- 给定N个数，求前1个数、前2个数…、前N个数的中位数

- 例子：
- 输入：[4, 5, 1, 3, 2, 6, 0]
- 输出：[4, 4, 4, 3, 3, 3, 3]

分析：

- 每次排序，寻找中位数：$O(n^2 \log n)$
- 类似插入排序，寻找中位数：$O(n^2)$
- K个数的中位数，需要知道第K/2小和第K/2大
  - 最大堆+最小堆
- 动态维护中位数一般都是用双堆解决
  - 同理：动态维护第K大数

编程

Sliding Window Median

https://www.lintcode.com/problem/sliding-window-median/

https://www.jiuzhang.com/solutions/sliding-window-median/

LintCode 360

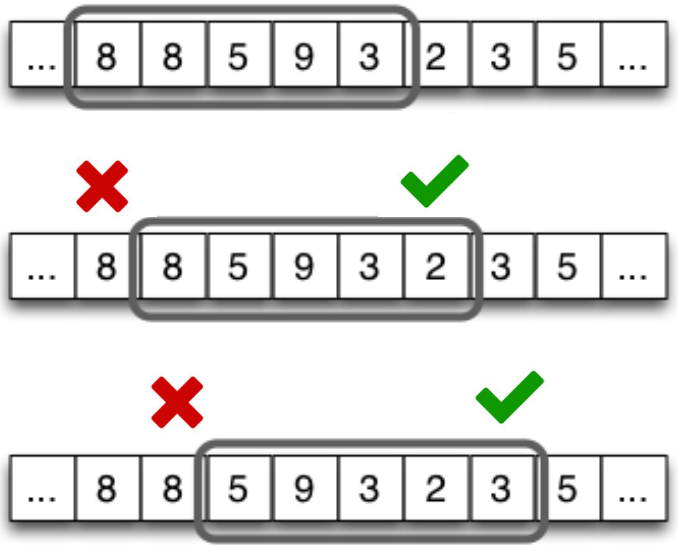
- 给定N个数，求每连续K个数的中位数

- 例子：
- 输入：[1,2,7,8,5], K = 3
- 输出：[2, 7, 7]

分析：

九章算法

- 和Data Stream Median类似，只是因为窗口移动需要删除已经不在窗口的元素
- 维护第K/2小和第K/2大
  - 最大堆+最小堆



Trapping Rain Water 2

http://www.lintcode.com/problem/trapping-rain-water-ii/

http://www.jiuzhang.com/solutions/trapping-rain-water-ii/

LintCode 364

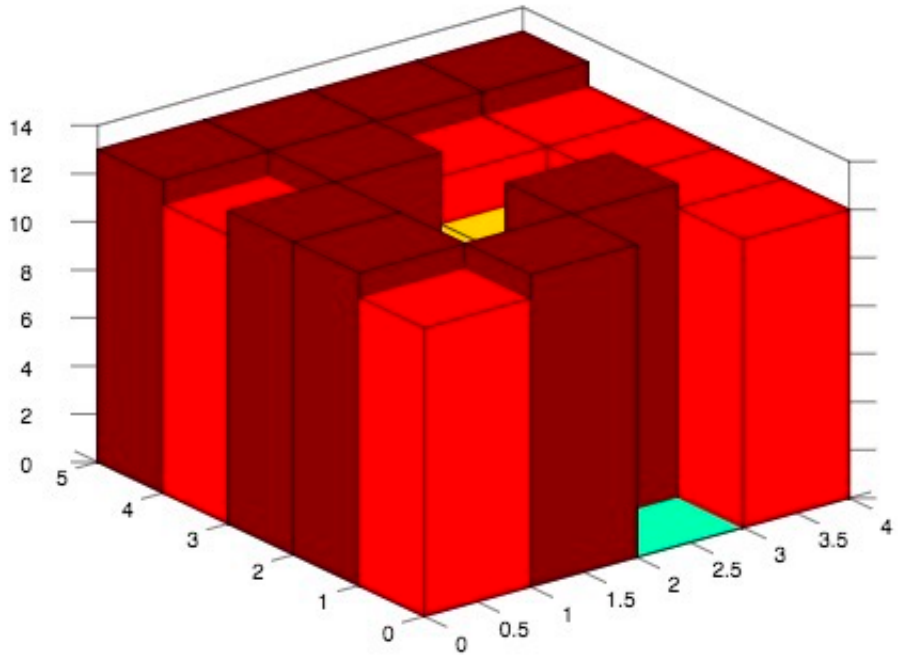
九章算法

- 给定一个nxm的2D地图每个格子的高度，求其中可以装多少单位的水

- 例子：
- 输入：

```
[12,13,0,12]
[13,4,13,12]
[13,8,10,12]
[12,13,12,12]
[13,13,13,13]
```

- 输出：14



分析：

- 一个格子高度h，它上面能盛多少水
  - 检查所有从这个格子到边界的路径，每条路径l都有最大高度值M_l
  - 所有路径最小的M_l值就是这个格子的**吃水线**，吃水线-h=盛水量
- 用最小堆维护访问的点，先访问边界一圈的点
- 每次从最小堆顶拿出点P，向其周围4个方向上看未曾访问过的点，如Q
  - 如果Q的高度<=P的吃水线，说明Q的吃水线就是P的吃水线！因为Q是第一次被访问到
  - 如果Q的高度>P的吃水线，则说明它不能装水，Q的吃水线就是自身的高度
  - 将(Q, Q的吃水线)加入Heap中

编程

栈 Stack

支持操作：O(1) Push / O(1) Pop / O(1) Top

Min Stack

https://www.lintcode.com/problem/min-stack/

https://www.jiuzhang.com/solutions/min-stack/

LintCode 12

九章算法

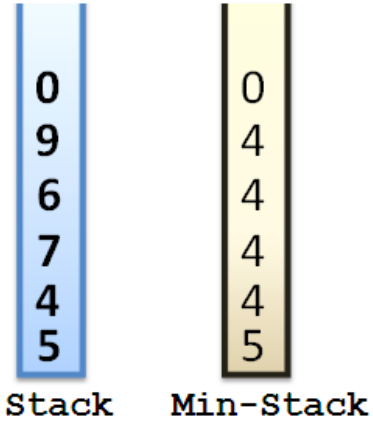
- 支持一个栈的push，pop和min操作，时间复杂度都要求是O(1)

- 例子：

push(1)
pop() // return 1
push(2)
push(3)
min() // return 2
push(1)
min() // return 1

分析：

- 栈的push和pop都是O(1)
- 求min可以用一个辅助栈minStack，和stack一样大
- minStack里每个元素表示stack里对应位置元素到栈底的最小值



休息五分钟

Decode String

https://www.lintcode.com/problem/expression-expand/
https://www.jiuzhang.com/solutions/expression-expand/

LintCode 575

九章算法

- 给定一个表达式，其中“数字[表达式]”表示方括号里的表达式重复数字次。输出展开的表达式

- 例子：

s = abc3[a] 输出： abcaaa
s = 3[abc] 输出： abcabcabc
s = 4[ac]dy, 输出： acacacacdy
s = 3[2[ad]3[pf]]xyz, 输出： adadpfpfpfadadpfpfpfadadpfpfpfxyz

分析：

- 可以DFS
- 改成非递归需要栈
- 数字和字符都push
  - 见到“[”push当前数字入栈
  - 字符直接压栈
- 见到“]”就pop字符直到碰到数字A
  - 这些字符组成的字符串重复A次

编程

单调栈
Monotonous stack

栈中只保存升序序列

How? 新元素插入前 pop 掉所有比它大的

stack([1,2,8,10]).push(5) => stack([1,2,5])

Largest Rectangle in Histogram

http://www.lintcode.com/problem/largest-rectangle-in-histogram/

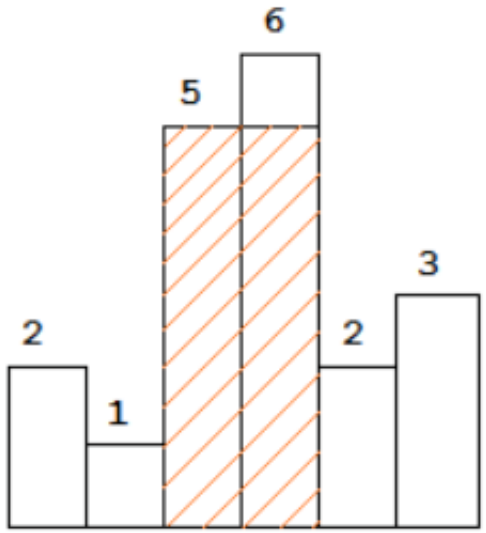
http://www.jiuzhang.com/solutions/largest-rectangle-in-histogram

LintCode 122

九章算法

- 给定直方图中n个柱形的高度，输出其中最大的矩形

- 例子：
- 输入：[2, 1, 5, 6, 2, 3]
- 输出：10



分析：

- 最大矩形一定是某一个柱形往左往右直到不能前进，形成的矩形
- 需要知道一个数字往左和往右第一个小于这个数字的位置
- 单调递增栈
  - 压栈时弹出大于等于自己的值
  - 最后停下来时碰到的栈顶就是左边第一个比自己小的值
  - 一个数X被新来的值R弹出栈顶，那么R就是X右边第一个小于等于X的值
    - 如果有相同的数，那么最靠右的bar会求得最大面积
    - 最后插入-1
- 时间复杂度O(N)

编程

Maximal Rectangle

http://www.lintcode.com/problem/maximal-rectangle

http://www.jiuzhang.com/solutions/maximal-rectangle/

LintCode 510

九章算法

- 给定一个01矩阵，求其中最大的全1矩形的面积

- 例子：
- 输入：

```
[
  [1, 1, 0, 0, 1],
  [0, 1, 0, 0, 1],
  [0, 0, 1, 1, 1],
  [0, 0, 1, 1, 1],
  [0, 0, 0, 0, 1]
]
```

- 输出：6

分析：

- 可以利用直方图的算法
- 以矩阵每一行为直方图的底部，通过1确定每个柱形的高度
- 求出最大全1子矩阵
- 时间复杂度O(N^2)

```
[
  [1, 1, 0, 0, 1],
  [0, 1, 0, 0, 1],
  [0, 0, 1, 1, 1],
  [0, 0, 1, 1, 1],
  [0, 0, 0, 0, 1]
]
```

Max Tree

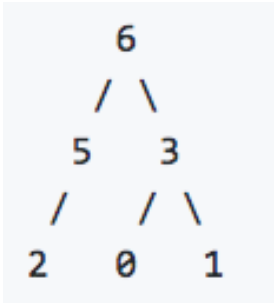
http://www.lintcode.com/problem/max-tree/

http://www.jiuzhang.com/solutions/max-tree/

LintCode 126

九章算法

- 给定一个无重复的整数数组。要求建立一棵二叉树，规则如下：
  - 最大值作为二叉树的根
  - 最大值分出的左右两段数组分别继续建树，作为根节点的左右子树
- 例子：
- 输入：[2, 5, 6, 0, 3, 1]
- 输出：



分析：

- 我们发现，每个值X的父亲一定是min{左边第一个比它大的值L，右边第一个比它大的值R}
  - ….., L, <X, …,<X, X, <X, …, <X, R,…
  - 如果L<R，[L, R]里一定R先做根。然后[L, R)里L先做根，然后就是X
  - 如果L>R， [L, R]里一定L先做根。然后(L, R]里R先做根，然后就是X
- 如何找到每个值左右第一个比它大的值？
  - 单调递减栈

小结

- 单调栈专门解决找一个值左/右第一个比它大/小的值
- 线性时间复杂度

总结

- 堆
  - 解决动态求最大/小值
  - 可以解决动态第K大/小问题
  - 双堆可以解决动态中位数
- 栈
  - 实现非递归
  - 单调栈解决找一个值左/右第一个比它大/小的值