Open in app Get started

Published in TechToFreedom

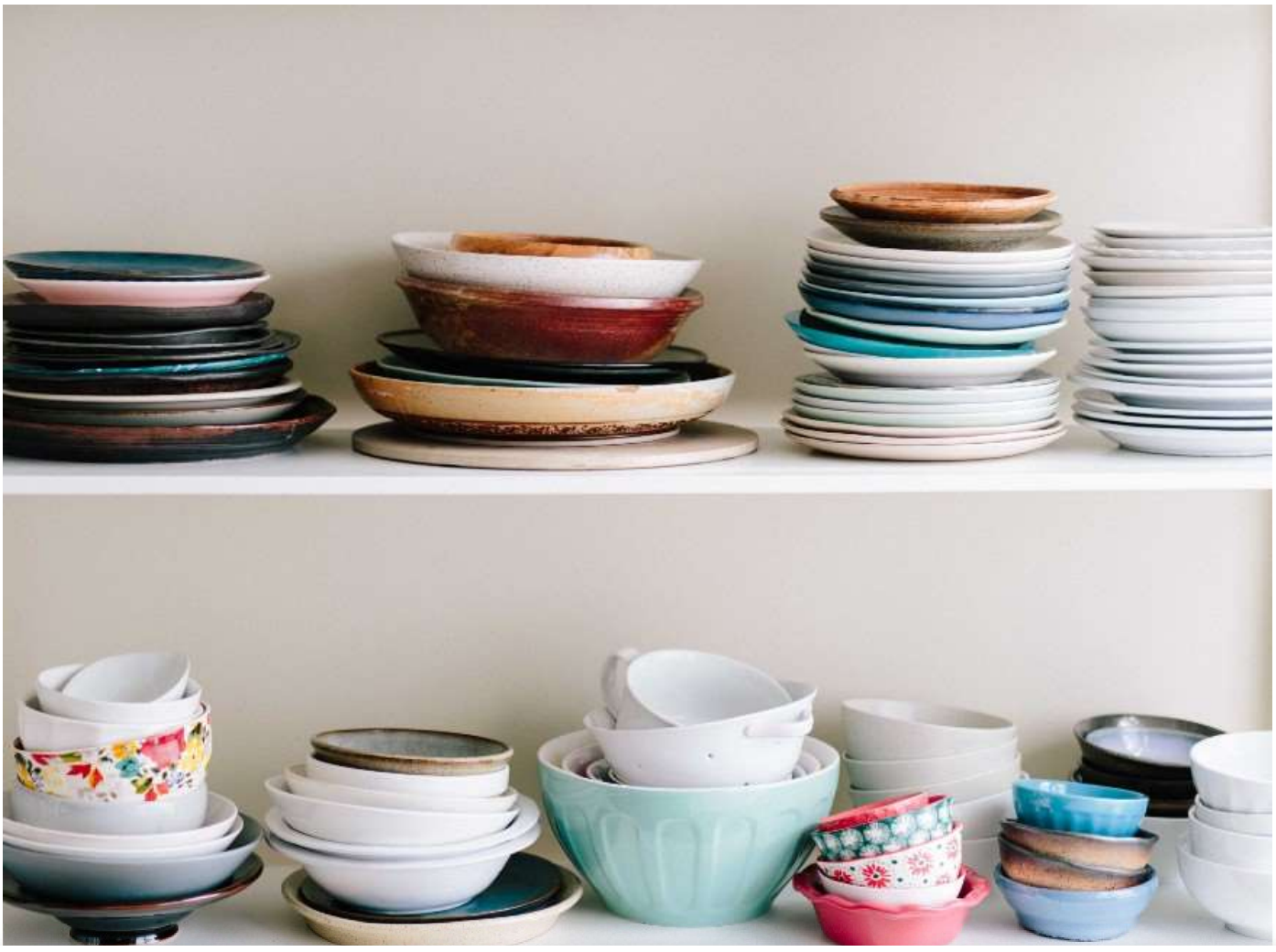
You have **1** free member-only story left this month. Sign up for Medium and get an extra one

Yang Zhou Follow

May 8, 2020 · 4 min read ★ · Listen

Save

Algorithms for Interview 2: Monotonic Stack



Open in app Get started

Monotonic Stack is a special variation of the typical data structure Stack and appeared in many interview questions.

As its name shows, monotonic stack contains all features that a normal stack has and its elements are all monotonic decreasing or increasing.

When to Use Monotonic Stack

Monotonic Stack is the best time complexity solution for many "*range queries in an array*" problems. Because every element in the array could only enter the monotonic stack once, the time complexity is O(N). (N represents the length of the array).

For every "range query" problem, it could be sure to maintain a range using a normal array/list. However, we will do many repetitive operations to get information from every range. The time complexity is very high and the solution is always not good enough to pass an interview.

Using monotonic stack to maintain a range can save lots of time. Because it only updates information within the range based on new adding elements and avoids repetitive operations of existing elements. To be more precisely, the monotonic stack helps us maintain maximum and and minimum elements in the range and keeps the order of elements in the range. Therefore, we don't need to compare elements one by one again to get minima and maxima in the range. At mean while, because it keeps the elements' order, we only need to update the stack based on newest adding element.

The concept of monotonic stack, which is just a stack keeping monotonic, is not difficult to understand. But the hardest part is how to use this data structure to model and solve problems. No one will tell you that a problem should use monotonic stack in a real interview. Sometime it's not obvious which solution is best for a problem if you are not familiar with many techniques and example problems. If a problem is suitable to use monotonic stack, it must has at least three characters:

1. **It is a "range queries in an array" problem.**

Open in app Get started

Let's have a look at some classic problems solved by monotonic stack:

Example Interview Problem 1:

This problem is from LeetCode 84. Largest Rectangle in Histogram.

```python
class Solution:
    def largestRectangleArea(self, heights: List[int]) -> int:
        ret = 0
        mono_stack = []
        heights.append(0)
        for i, v in enumerate(heights):
            while mono_stack and heights[mono_stack[-1]] > v:
                height = heights[mono_stack.pop()]
                if mono_stack:
                    length = i - mono_stack[-1]-1
                else:
                    length = i
                ret = max(ret, height * length)
            mono_stack.append(i)
        return ret
```

example1.py hosted with ❤ by GitHub view raw

Firstly, it looks like a range query problem. We maintain an monotonic increasing stack called `mono_stack` . When we meet a new height:

- If this height is higher than or equal to the last height in mono_stack, we just add it to the mono_stack.
- If this height is lower than the last height in mono_stack, we must pop the last height of mono_stack to keep the monotone.

The hardest operation happens at the second situation. As our rule 3: **When a element is popped from the monotonic stack, it will never be used again.** If we pop the last height, it should be totally thrown away and never added to stack again. If not, we cannot ensure that every item is added to the monotonic stack once and so the time complexity of

Open in app Get started

Therefore, when pop an item, we should calculate the largest possible rectangle it can generate and update the `ret` value. When we generate its largest possible rectangle, we meet rule 2 (**The minima/maxima element or the monotonic order of elements in a range is useful to get answer of every range query.**). Because of the monotone of the stack, we can get the leftmost and rightmost indexes which the popping item can "use" to generate rectangle easily. So we just use `height*length` to get the area of the rectangle.

Finally, we need to deal with the end boundary. We can just add a height 0 to the end of `heights` which can make sure that all previous heights were checked when we end the for loop.

Example Interview Problem 2:

This problem is from LeetCode 42. Trapping Rain Water.

```python
class Solution:
    def trap(self, heights: List[int]) -> int:
        stack = []  # a decreasing stack
        total = 0
        for i, v in enumerate(heights):
            while stack and heights[stack[-1]] < v:
                popped_idx = stack.pop()
                if not stack:
                    break
                height = min(heights[stack[-1]], v) - heights[popped_idx]
                length = i - stack[-1] - 1
                total += height * length
            stack.append(i)
        return total
```

example2.py hosted with ❤ by GitHub view raw

The idea is similar with problem 1. The differences are:

- Use a decreasing stack.

Open in app Get started

The monotonic stack is a very useful data structure and appearing in many interview problems. The most difficult part is to know whether a problem can be solved by this data structure or not.

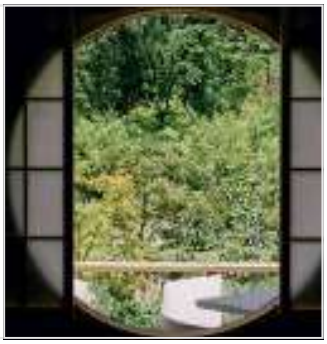
We need to remember the three characters to help us think about that:

1. **It is a "range queries in an array" problem.**
2. **The minima/maxima element or the monotonic order of elements in a range is useful to get answer of every range query.**
3. **When a element is popped from the monotonic stack, it will never be used again.**

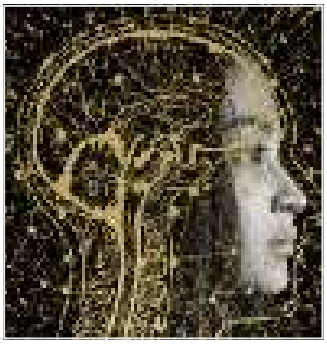
Thanks for reading. If you like it, please follow me and become and Medium member to enjoy more great articles about programming and technologies!

Enjoy more:

The Art of Writing Loops in Python
Simple is better than complex
medium.com



Memory Management in Python: 3 Popular Interview Questions
Dive into the internal mechanisms
medium.com



Open in app Get started

Get an email whenever Yang Zhou publishes.

Your email

Subscribe

By signing up, you will create a Medium account if you don't already have one. Review our Privacy Policy for more information about our privacy practices.

Medium

About Help Terms Privacy

Get the Medium app

Download on the App Store

GET IT ON Google Play