Daily Temperatures

Given a list of daily `temperatures`, produce a list that, for each day in the input, tells you how many days you would have to wait until a warmer temperature. If there is no future day for which this is possible, put `0` instead.

For example, given the list `temperatures = [73, 74, 75, 71, 69, 72, 76, 73]`, your output should be `[1, 1, 4, 2, 1, 1, 0, 0]`.

Note: The length of `temperatures` will be in the range `[1, 30000]`. Each temperature will be an integer in the range `[30, 100]`.

Solution 1

Time: O(N)
Space: O(N)

Method 1. `Stack` (54 ms)

```
public int[] dailyTemperatures(int[] temperatures) {
    Stack<Integer> stack = new Stack<>();
    int[] ret = new int[temperatures.length];
    for(int i = 0; i < temperatures.length; i++) {
        while(!stack.isEmpty() && temperatures[i] > temperatures[stack.peek()]) {
            int idx = stack.pop();
            ret[idx] = i - idx;
        }
        stack.push(i);
    }
    return ret;
}
```

Method 2. `Array` (10 ms)

```
public int[] dailyTemperatures(int[] temperatures) {
    int[] stack = new int[temperatures.length];
    int top = -1;
    int[] ret = new int[temperatures.length];
    for(int i = 0; i < temperatures.length; i++) {
        while(top > -1 && temperatures[i] > temperatures[stack[top]]) {
            int idx = stack[top--];
            ret[idx] = i - idx;
        }
        stack[++top] = i;
    }
    return ret;
}
```

written by luckman original link here

Solution 2

The range of temperature is quite small, so it is possible to have a hash map of temperatures to earliest days when that temperature occurred.

We iterate through the list of temperatures from the back, and for each day, loop through higher temperatures and find the minimum day for existing higher temperatures.

Example, for the input, when we are at 72

```
[73, 74, 75, 71, 69, 72, 76, 73]
                       ^
# We have the following hash map:
{
  73: 7,
  76: 6,
}
```

The minimum day value for temperatures higher than 72 is 6 (at 76 degrees).

- *Yangshun*

```
class Solution(object):
    def dailyTemperatures(self, temperatures):
        """
        :type temperatures: List[int]
        :rtype: List[int]
        """
        # Time: O(n.k), where k is the range of the temperature.
        # Space: O(n + k)
        temps = {}
        results = [0] * len(temperatures)
        for i in range(len(temperatures) - 1, -1, -1):
            temp = temperatures[i]
            day = str('inf')
            for j in range(temp + 1, 101):
                if j in temps:
                    day = min(day, temps[j] - i)
            if day != str('inf'):
                results[i] = day
            temps[temp] = i
        return results
```

written by yangshun original link here

Solution 3

- for each day from `end` to `start`, record the `next` day of temperature `t` for all `t` in `[30, 100]`
- for each day from n-1 to 0; for all temperature higher than temp[i], find the earliest
  Java

```java
class Solution {
    public int[] dailyTemperatures(int[] temps) {
        int n = temps.length;
        int[] waits = new int[n];
        int[] next = new int[101]; // next day with with certain temperature.
        for (int i = n - 1; i >= 0; i--) {
            int earliest = Integer.MAX_VALUE;
            for (int t = temps[i] + 1; t <= 100; t++) {
                if (next[t] != 0) earliest = Math.min(earliest, next[t]);
            }
            if (earliest != Integer.MAX_VALUE) waits[i] = earliest - i;
            next[temps[i]] = i;
        }
        return waits;
    }
}
```

C++

```cpp
class Solution {
public:
    vector<int> dailyTemperatures(vector<int>& temps) {
        int n = temps.size();
        vector<int> waits(n, 0);
        vector<int> next(101, INT_MAX); // next day with with certain temperature.
        for (int i = n - 1; i >= 0; i--) {
            int earliest = INT_MAX;
            for (int t = temps[i] + 1; t <= 100; t++) {
                earliest = min(earliest, next[t]);
            }
            if (earliest != INT_MAX) waits[i] = earliest - i;
            next[temps[i]] = i;
        }
        return waits;
    }
};
```

written by alexander original link here