< Back | Using heapq.merge

StefanPochmann ★ 46942 June 17, 2016 8:53 PM 665 VIEWS

2

Handle inputs with positive `a` by negating the function. For **non**-positive `a`, the smallest number in the transformed array is at an end rather than in the middle. Just merge from the two ends.

```
def sortTransformedArray(self, nums, a, b, c):
    if a > 0:
        flipped = self.sortTransformedArray(nums, -a, -b, -c)
        return [-y for y in flipped[::-1]]
    y = [a*x*x + b*x + c for x in nums]
    return list(heapq.merge(y, reversed(y)))[:len(y)]
```

heapq.merge wants sorted inputs, but it's implemented the obvious way and thus it works here anyway.

A slightly cleaner version, only merging as much as needed (see last two lines):

```
def sortTransformedArray(self, nums, a, b, c):
    if a > 0:
        flipped = self.sortTransformedArray(nums, -a, -b, -c)
        return [-y for y in flipped[::-1]]
    y = [a*x*x + b*x + c for x in nums]
    merge = heapq.merge(y, reversed(y))
    return list(itertools.islice(merge, len(y)))
```

An uglier version, couldn't find nice variable names...

```
def sortTransformedArray(self, nums, a, b, c):
    m = 1 if a < 0 else -1
    y = [(a*x*x + b*x + c)*m for x in nums]
    y = list(heapq.merge(y, reversed(y)))[:len(y)]
    return [y*m for y in y[::m]]
```

python short heapq