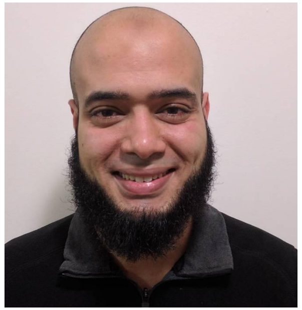
Algorithms
D&C Homework

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



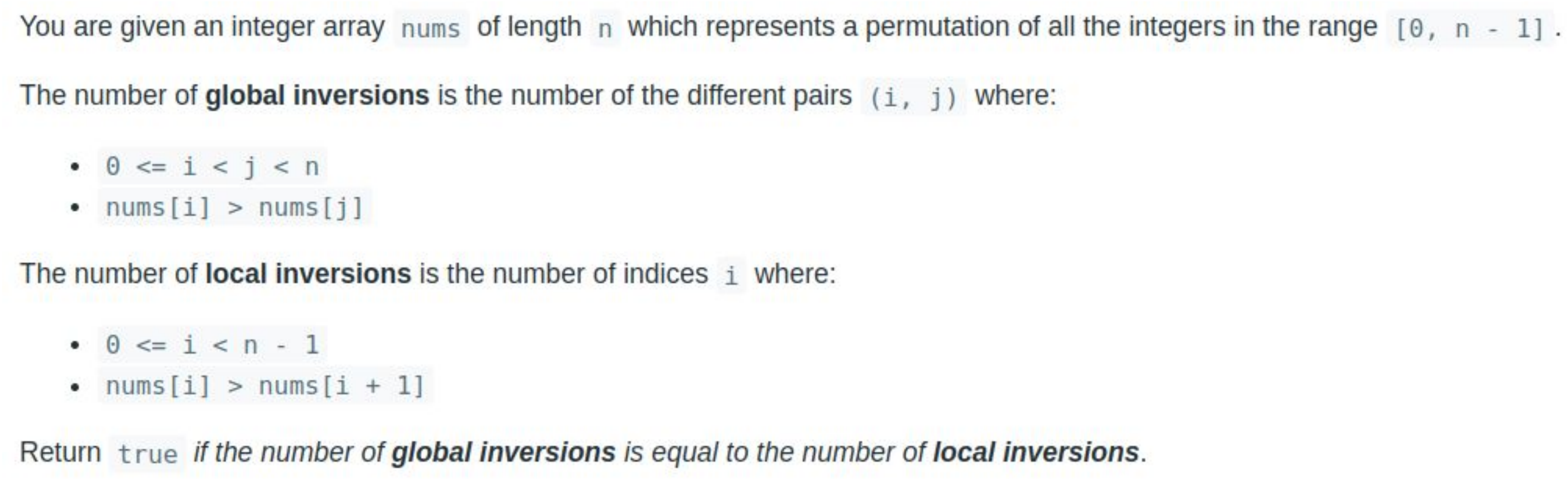
Background: Counting Inversion

- Given an array, the inversion count is the total number of pairs with indices i and j such that: i < j and a[i] > a[j]
  - In other words, number a is before number b but its value is bigger
- Example: 9, 3, 2, 0, 20
  - There are 6 inversions
  - 9 has 3 inversions with {3, 2, 0}
  - 3 has 2 inversions with {2, 0}
  - 2 has 1 inversion with 0
- In terms of sorting, why is that useful?

Background: Counting Inversion

- A sorted array has 0 inversions.
- A **reverse** sorted array of N elements has (N*(N-1))/2 inversions (the maximum)
- So inversion count is an **indicator** for degree of sortedness

Problem #1: LeetCode 775 - Global and Local Inversions



You are given an integer array `nums` of length `n` which represents a permutation of all the integers in the range `[0, n - 1]`.

The number of **global inversions** is the number of the different pairs `(i, j)` where:

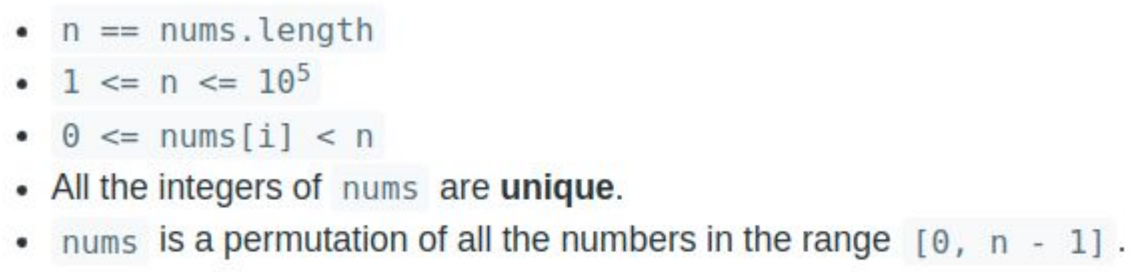
- `0 <= i < j < n`
- `nums[i] > nums[j]`

The number of **local inversions** is the number of indices `i` where:

- `0 <= i < n - 1`
- `nums[i] > nums[i + 1]`

Return `true` *if the number of **global inversions** is equal to the number of **local inversions***.

- **Note**: counting the actual inversions won’t fit in 32 bit
    - Why? Max inversions in 10^5



- `n == nums.length`
- `1 <= n <= 10`^{5}
- `0 <= nums[i] < n`
- All the integers of `nums` are **unique**.
- `nums` is a permutation of all the numbers in the range `[0, n - 1]`.

Example 1:

```
Input: nums = [1,0,2]
Output: true
Explanation: There is 1 global inversion and 1 local inversion.
```

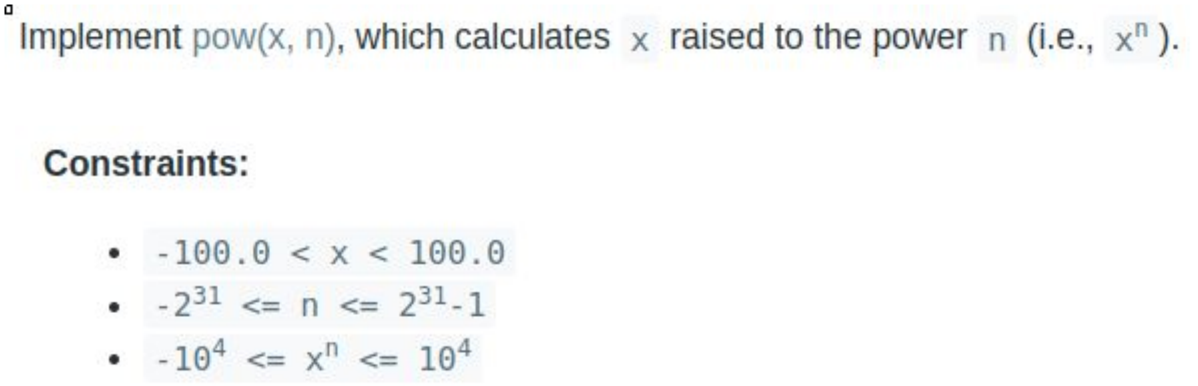
Example 2:

```
Input: nums = [1,2,0]
Output: false
Explanation: There are 2 global inversions and 1 local inversion.
```

Problem #2: Handling Duplicates

- The lecture code is handling array of unique values
- Please change the code to quicksort the data even if there are duplicate cases
- Test well your code

Problem #3: LeetCode 50 - Pow(x, n)



Implement pow(x, n), which calculates `x` raised to the power `n` (i.e., x^n).

Constraints:

- `-100.0 < x < 100.0`
- $-2^{31} <= n <= 2^{31}-1$
- $-10^4 <= x^n <= 10^4$

Example 1:

Input: x = 2.00000, n = 10
Output: 1024.00000

Example 2:

Input: x = 2.10000, n = 3
Output: 9.26100

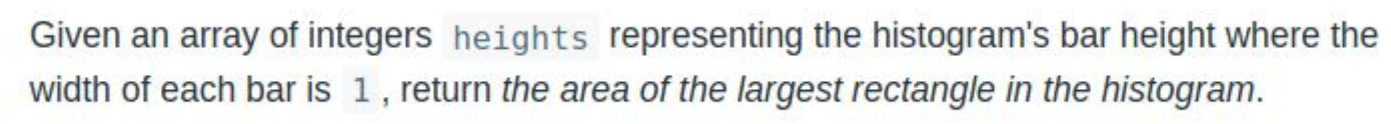
Example 3:

Input: x = 2.00000, n = -2
Output: 0.25000
Explanation: $2^{-2} = 1/2^{2} = 1/4 = 0.25$

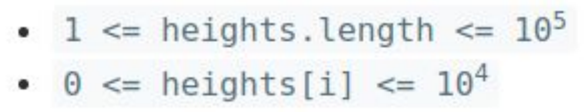
Tips

- For simplicity, assume positive integers only
    - pow(2, 4) = 16
- How can we apply D&C to this problem?
- Say n = 16
    - x^16 vs x^8
    - Given x^8, can we get x^16? x^17?
- Be careful from overflows

Problem #4: LeetCode 84 - Largest Rectangle in Histogram

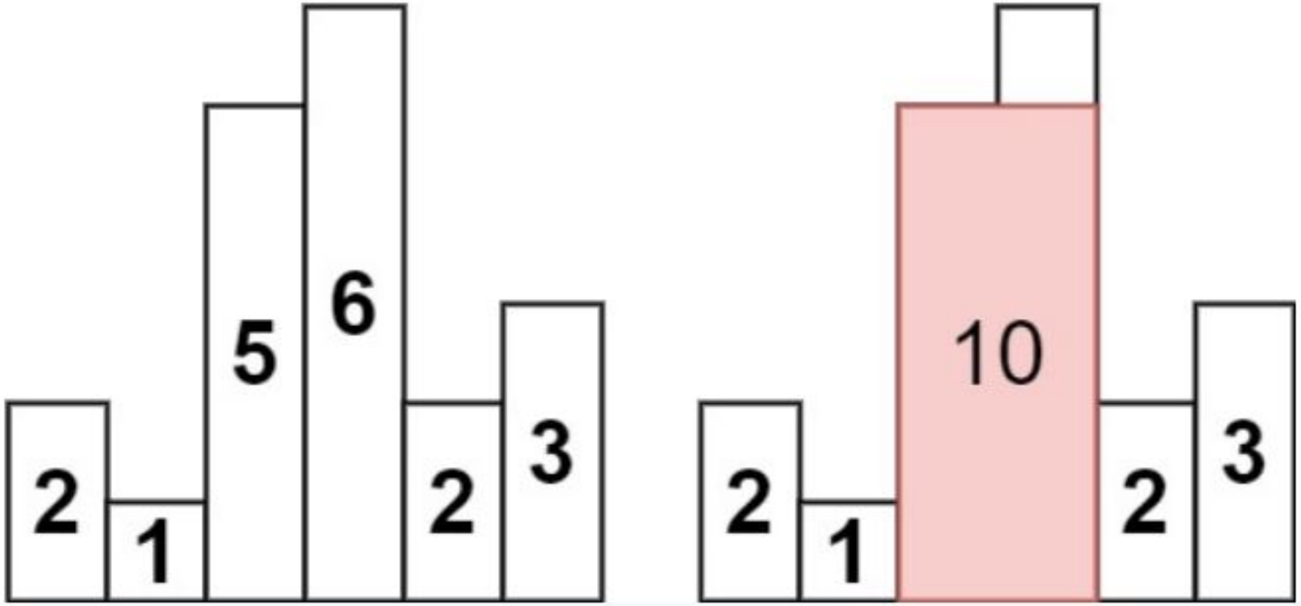


Given an array of integers `heights` representing the histogram's bar height where the width of each bar is `1`, return *the area of the largest rectangle in the histogram*.



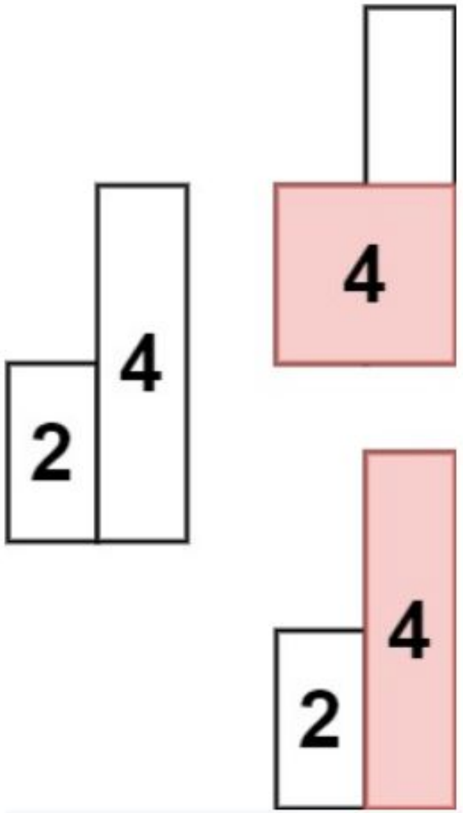
- $1 <= heights.length <= 10^5$
- $0 <= heights[i] <= 10^4$

Example 1:



```
Input: heights = [2,1,5,6,2,3]
Output: 10
Explanation: The above is a histogram where width of each bar is 1.
The largest rectangle is shown in the red area, which has an area =
10 units.
```

Example 2:



```
Input: heights = [2,4]
Output: 4
```

More:

- 2 1 **4** **5** 1 3 3 ⇒ 8
  - 2*4
- 1000 1000 1000 1000 ⇒ 4000
  - 4 * 1000

Hint and Notes

- The ‘Largest Rectangle in Histogram’ is a popular problem with several solutions
  - We will encounter it again in the interviews course
- The hint that can help you is: Given a range, what does the **minimum value** in this range represents?
  - To pass LeetCode time limit, you need to implement this function in O(logn)
  - Attached is a **code template** ready for you with this function.
  - You will only implement the D&C function **getMaxArea**
    - You can make calls getMinIdx(l, r) to get the minimum index in this range
- What is the time complexity?

Problem #5 - CodeForces 560D - Round #313 (Div. 2)

- Two strings a and b of equal length are called equivalent IFF:
    - They are equal.
    - If we split string a into two halves of the same size a1 and a2,
      and string b into two halves of the same size b1 and b2,
      then one of the following is correct:
        - a1 is equivalent to b1, and a2 is equivalent to b2
        - a1 is equivalent to b2, and a2 is equivalent to b1
- Read 2 strings (each on a line), Print **YES** if these two strings are equivalent, and **NO** otherwise.
    - Each of them has the length from 1 to 200 000 and consists of lowercase English letters.
    - The strings have the same length.

Examples

input

```
aaba
abaa
```

output

```
YES
```

input

```
aabb
abab
```

output

```
NO
```

Note

In the first sample you should split the first string into strings "aa" and "ba", the second one — into strings "ab" and "aa". "aa" is equivalent to "aa"; "ab" is equivalent to "ba" as "ab" = "a" + "b", "ba" = "b" + "a".

In the second sample the first string can be splitted into strings "aa" and "bb", that are equivalent only to themselves. That's why string "aabb" is equivalent only to itself and to string "bbaa".

Hint

- Create a **D&C** function canonical(string) that returns the **same** representation of all strings that are equivalent
    - xycde ⇒ xycde
    - xycd ⇒ cdxy
    - xydc ⇒ cdxy
    - yxdc ⇒ cdxy
    - dcxy ⇒ cdxy # All these 4 strings are equivalent. We return cdxy
    - ydxc ⇒ cxdy
- Then, we can trivially check if canonical(str1) == canonical(str2)
- You don’t need to submit on Codeforces. Just do some local tests

Optional

- Learn how to merge k-sorted arrays (see see)
    - In the interviews course, we will learn how to merge k linkedlists
- Think How to sort a huge file which **can't be loaded** in memory
    - Tip: Split to small files. Sort each in memory (quicksort). Use files to merge (sorted files)
- Write a code to do matrix power, similar to the power function solution
    - You can practice that on Fibonacci
- Learn other ways to code the partition function

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."