{% note info %} **摘要** Title: 788. 逆序对的数量 Tag: 归并排序、逆序对、树状数组 Memory Limit: 64 MB Time Limit: 1000 ms {% endnote %}

Powered by:**NEFU AB-IN**

Link

@TOC

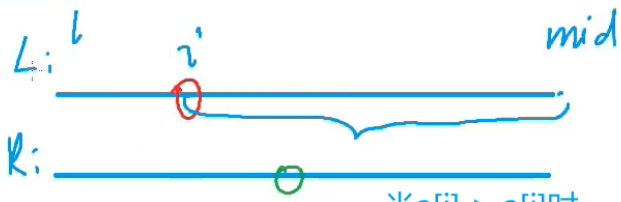
788. 逆序对的数量

- 题意

> 给定一个长度为 n 的整数数列，请你计算数列中的逆序对的数量。 逆序对的定义如下：对于数列的第 i 个和第 j 个元素，如果满足 i<j 且 a[i]>a[j]，则其为一个逆序对；否则不是。

- 思路

1. 左半边内部的逆序对数量：merge_sort(L, mid)
2. 右半边内部的逆序对数量：merge_sort(mid + 1, R)
3.



当a[i] > a[j]时
j与a[i]（包括a[i]）右边的数都形成逆序对

$s_j = mid - j + 1$

归并排序：

1. [L, R] => [L, mid], [mid + 1, R]
2. 递归排序[L, mid]和[mid + 1, R]
3. 归并，将左右两个有序序列合并成一个有序序列

- 代码

```
'''
Author: NEFU AB-IN
Date: 2022-02-18 15:25:02
FilePath: \ACM\Acwing\788.py
LastEditTime: 2022-02-18 15:36:14
'''

a = []
```

```
def merge(l, r):
    if l >= r:
        return 0
    mid = l + r >> 1
    res = merge(l, mid) + merge(mid + 1, r)

    i, j = l, mid + 1
    tmp = []

    while i <= mid and j <= r:
        if a[i] <= a[j]:
            tmp.append(a[i])
            i += 1
        else:
            tmp.append(a[j])
            j += 1
            res += mid - i + 1
    tmp += a[i:mid + 1]
    tmp += a[j:r + 1]
    a[l:r + 1] = tmp
    return res


n = int(input())
a = list(map(int, input().split()))

print(merge(0, n - 1))
```

所以这样也可以，**用总的数对减去正序对（包括相等的）**，也就是逆序对

```
'''
Author: NEFU AB-IN
Date: 2022-02-18 15:25:02
FilePath: \ACM\Acwing\788.py
LastEditTime: 2022-02-18 15:48:43
'''

a = []


def merge(l, r):
    if l >= r:
        return 0
    mid = l + r >> 1
    res = merge(l, mid) + merge(mid + 1, r)

    i, j = l, mid + 1
    tmp = []
```

```
    while i <= mid and j <= r:
        if a[i] <= a[j]:
            tmp.append(a[i])
            i += 1
            res += r - j + 1
        else:
            tmp.append(a[j])
            j += 1
    tmp += a[i:mid + 1]
    tmp += a[j:r + 1]
    a[l:r + 1] = tmp
    return res


n = int(input())
a = list(map(int, input().split()))

print(n * (n - 1) // 2 - merge(0, n - 1))
```

树状数组求逆序对

所以规律如下：(**i从1开始**) **[比我小]： $sum(a[i]-1)$ [小于等于我] : $sum(a[i])$ [比我大] : $i-sum(a[i])$ (逆序对) [大于等于我] : $i-sum(a[i]-1)$**

```
'''
Author: NEFU AB-IN
Date: 2022-02-18 16:31:39
FilePath: \ACM\Acwing\788.1.py
LastEditTime: 2022-02-18 16:47:40
'''
from bisect import bisect_left
from copy import deepcopy

N = int(1e5 + 100)
tr = [0] * N
b = []
lowbit = lambda x: x & -x


def add(x, d):
    while x <= n:
        tr[x] += d
        x += lowbit(x)


def getsum(x):
    res = 0
    while x > 0:
        res += tr[x]
        x -= lowbit(x)
```

```
    return res


if __name__ == "__main__":
    n = int(input())
    xs = list(map(int, input().split()))
    res = 0

    a = deepcopy(xs)
    xs = list(set(xs))
    xs.sort()

    for i in range(n):
        b.append(bisect_left(xs, a[i]) + 1)  #离散化的值不能等于0
    for i in range(n):
        add(b[i], 1)
        res += (i + 1 - getsum(b[i]))  #因为i是从0开始的所以+1
    print(res)
```