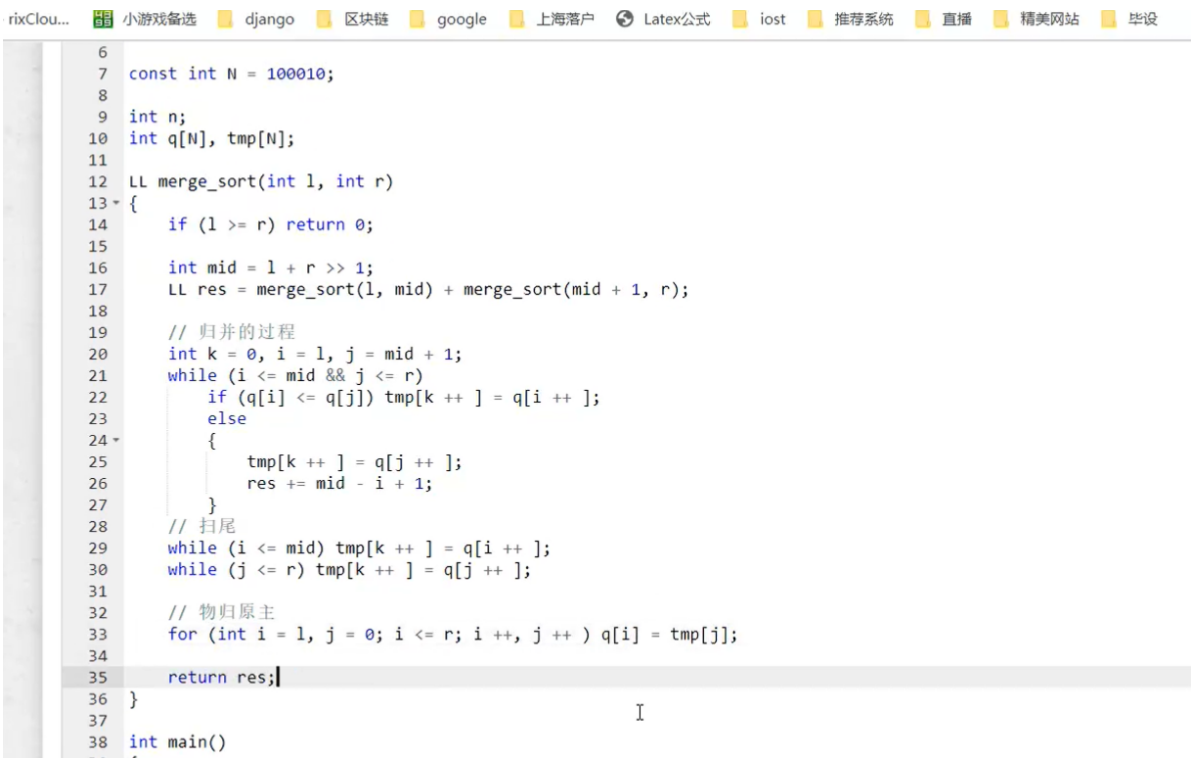
归并排序

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



```cpp
const int N = 100010;

int n;
int q[N], tmp[N];

LL merge_sort(int l, int r)
{
    if (l >= r) return 0;

    int mid = l + r >> 1;
    LL res = merge_sort(l, mid) + merge_sort(mid + 1, r);

    // 归并的过程
    int k = 0, i = l, j = mid + 1;
    while (i <= mid && j <= r)
        if (q[i] <= q[j]) tmp[k ++ ] = q[i ++ ];
        else
        {
            tmp[k ++ ] = q[j ++ ];
            res += mid - i + 1;
        }
    // 扫尾
    while (i <= mid) tmp[k ++ ] = q[i ++ ];
    while (j <= r) tmp[k ++ ] = q[j ++ ];

    // 物归原主
    for (int i = l, j = 0; i <= r; i ++, j ++ ) q[i] = tmp[j];

    return res;
}

int main()
```

```cpp
#include <iostream>

using namespace std;

typedef long long LL;

const int N = 1e5 + 10;

int a[N], tmp[N];

LL merge_sort(int q[], int l, int r)
{
    if (l >= r) return 0;

    int mid = l + r >> 1;
```

```
    LL res = merge_sort(q, l, mid) + merge_sort(q, mid + 1, r);

    int k = 0, i = l, j = mid + 1;
    while (i <= mid && j <= r)
        if (q[i] <= q[j]) tmp[k ++ ] = q[i ++ ];
        else
        {
            res += mid - i + 1;
            tmp[k ++ ] = q[j ++ ];
        }
    while (i <= mid) tmp[k ++ ] = q[i ++ ];
    while (j <= r) tmp[k ++ ] = q[j ++ ];

    for (i = l, j = 0; i <= r; i ++, j ++ ) q[i] = tmp[j];

    return res;
}

int main()
{
    int n;
    scanf("%d", &n);
    for (int i = 0; i < n; i ++ ) scanf("%d", &a[i]);

    cout << merge_sort(a, 0, n - 1) << endl;

    return 0;
}
```