

2021-10-28

Today Learned:

[Introduction of Algorithms lecture 1](#)

(Corresponding to the first and second chapters of the book)

Today Exercise:

Merge-Sort(From AOAPC II Page225-226)

Source Code:

```
// merge_sort.c
// author: Rujia Liu, Jerry Zhang (completing the code)

#include <stdio.h>
void merge_sort(int* A, int x, int y, int* T);

int main()
{
    int A[10] = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 };
    int T[10] = { 0 };
    merge_sort(A, 0, 10, T);
    for (int i = 0; i < 10; i++) printf("%d ", A[i]);
    printf("\n");
    return 0;
}

void merge_sort(int* A, int x, int y, int* T)
{
    if (y-x > 1) {
        int m = x + (y-x) / 2;
        int p = x; // starting index
        int q = m; // middle index
        int i = x; // counter
        merge_sort(A, x, m, T);
        merge_sort(A, m, y, T);
        while (p < m || q < y) {
            if (q ≥ y || (p < m && A[p] ≤ A[q])) {
                T[i++] = A[p++];
            }
            else {

```

```

        T[i++] = A[q++];
    }
}
for (i = x; i < y; i++) { A[i] = T[i]; }
}

```

Coding Notes:

merge sort:

array A: [1 | 3 | 5 | 1] with indices x , m , y above. Pointers p and q are shown below, with a 'compared' label between them.

array T: [1 | 3 | ...] with index i above.

```

while (p < m || q < y) {
    if (there is an unsorted array in either of two arrays)
        if (q >= y || (p < m && A[p] <= A[q]))
            T[i++] = A[p++];
        else T[i++] = A[q++];
}
for (i = x; i < y; i++) A[i] = T[i];
Copy back to array A from T (Temp array).

```

Annotations for the merge logic:

- $q \geq y$: Right array is empty
- $p < m$: Left array is not empty
- $A[p] \leq A[q]$: we need compare their values.
- Left unE. Right unE.