

Algorithms for Sorting

Insertion Sort

Input: unordered array $A[1..n]$.

Output: ordered array $A[1..n]$

Key idea: from the second element to the last element of the array, we insert each element $A[j]$ into the previously sorted subarray $A[1..j-1]$. For the insertion process of each element $A[j]$, do the following:

- Compare $A[j]$ with $A[j-1]$, to test whether $A[j] > A[j-1]$ or not.
- If yes, insert $A[j]$ after $A[j-1]$
- If no, compare $A[j]$ with the previous element of $A[j-1]$
- Repeat this process until we find the proper position for $A[j]$.

For example, we want to sort this array using insertion sort algorithm

5	2	4	6	1	3
---	---	---	---	---	---

1. Compare 2 with 5, to see if $2 > 5$?
2. No. And 5 is the first element, so we insert 2 before 5.

2	5	4	6	1	3
---	---	---	---	---	---

3. Compare 4 with 5, to see if $4 > 5$?
4. No. Compare 4 with 2, if $4 > 2$?
5. Yes! Therefore insert 4 after 2.

2	4	5	6	1	3
---	---	---	---	---	---

6. Compare 6 with 5, if $6 > 5$?
7. Yes. Insert 6 after 5.

2	4	5	6	1	3
---	---	---	---	---	---

8. Compare 1 with 6, if $1 > 6$?
9. No. Compare 1 with 5, if $1 > 5$?
10. No. If $1 > 4$?
11. No. If $1 > 2$?
12. No. And 2 is the first element, so we insert 1 before 2.

1	2	4	5	6	3
---	---	---	---	---	---

13. Compare 3 with 6, if $3 > 6$?
14. No. If $3 > 5$?
15. No. If $3 > 4$?
16. No. if $3 > 2$?
17. Yes. Insert 3 after 2.

1	2	3	4	5	6
---	---	---	---	---	---

And we've done.

Question: How many comparisons in total?
Simply count it out: 12.

Merge Sort

Input: 2 ordered subarrays $L[1..n]$ and $R[1..m]$

Output: 1 complete ordered $A[1..m+n]$

Key idea: Since the 2 subarrays are already sorted; we could just compare the elements of L with the elements of R one by one, from the beginning to the end. And merge the two subarrays together.

- a. Set a pointer at the first element of each subarray.
- b. Compare $L[1]$ with $R[1]$, to see if $L[1] > R[1]$.
- c. If yes, put $R[1]$ at the first place of the final array. Move the pointer of R to the next element $R[2]$.
- d. If no, put $L[1]$ at the first place of the final array. Move the pointer of L to the next element $L[2]$.
- e. Compare the 2 new elements which the pointers points. Keep doing these step until the end of both subarrays.