* اثبات درستی الگوریتم مرتب سازی درجی - نسخه بازگشتی :

* خط ۶ تا ۸ الگوریتم شبه کد قبل با استفاده از مفهوم Loop Invariant قابل اثبات است.

* برای کل الگوریتم هم از اثبات استقرایی استفاده می‌کنیم

← پایه استقرا: آرایه ۱ عنصری یک آرایه مرتب است.

← گام استقرا: k-1 عنصر اول آرایه مرتب باشد + درستی خط ۶ تا ۸ الگوریتم ← k عنصر اول آرایه مرتب است.

* مرور کلی برای محاسبه پیچیدگی زمانی الگوریتم مان بازگشتی :

$$T(n) = a\,T(n/b) + D(n) + C(n)$$

D(n) → Divide

C(n) → Combination

$T(1) = c$ ثابت

جایگذاری :

D(n) : زمان اجرای مرحله تقسیم برای n عنصر است،

C(n) : زمان اجرای مرحله ترکیب برای n عنصر است،

ضریب a : بیانگر تعداد زیرمساله‌هایی است که باید حل شود،

ضریب $1/b$: بیانگر آن است اندازه زیرمساله، $1/b$ اندازه مساله اصلی است.

* تحلیل پیچیدگی الگوریتم مرتب سازی درجی - نسخه بازگشتی :

$$T(n) = a\,T(n/b) + \underbrace{D(n)}_{O(1)} + \underbrace{C(n)}_{O(n)}$$

$a = 1$

$b = \frac{n}{n-1}$

$$T(n) = T(n-1) + O(n)$$

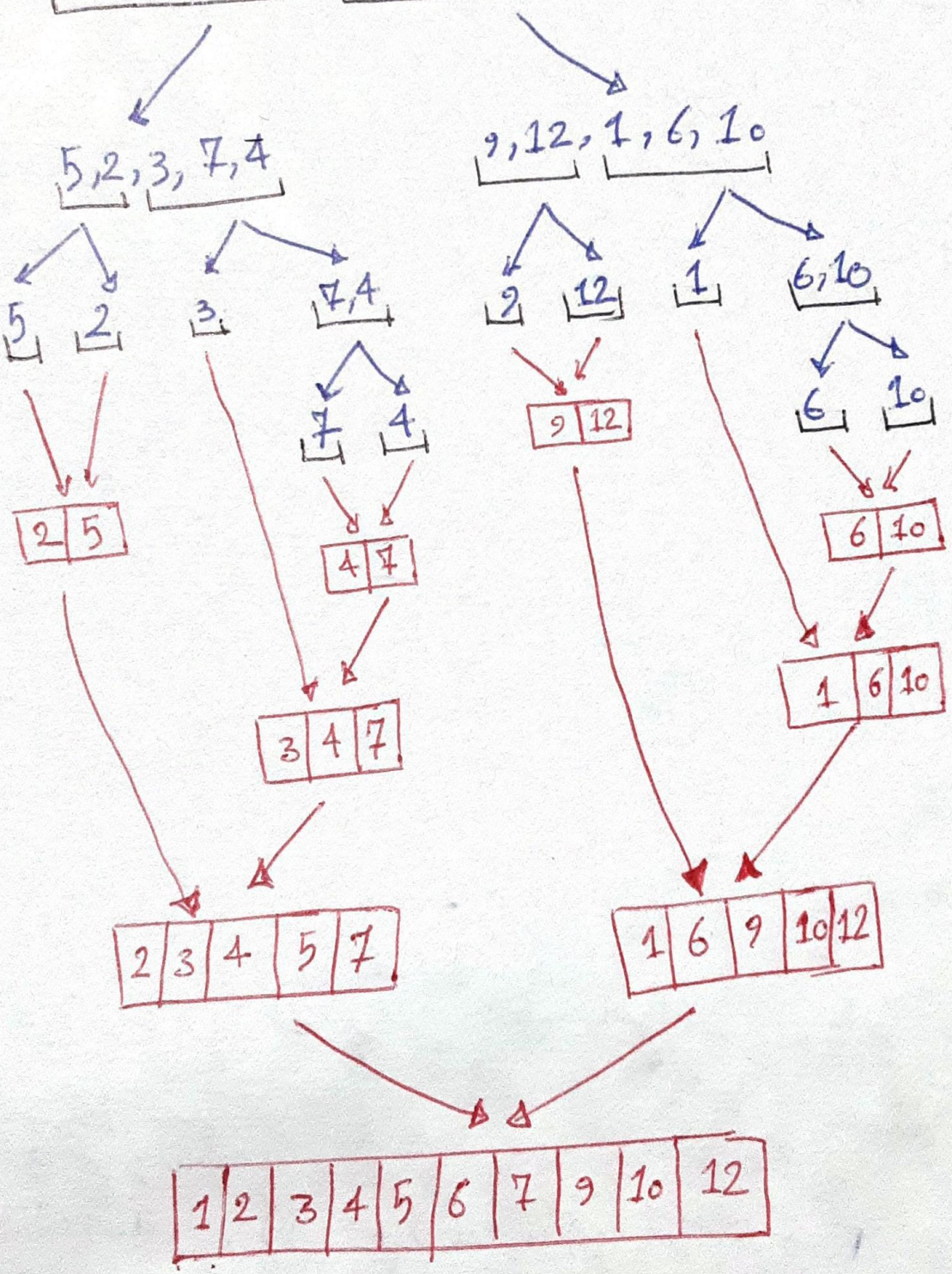
$$\Rightarrow T(n) = n * O(n) = O(n^2)$$

مرتب سازی ادغامی (Merge sort): (پیچیدگی زمانی در بهترین، بدترین و متوسط $\Theta(n \log_2 n)$)

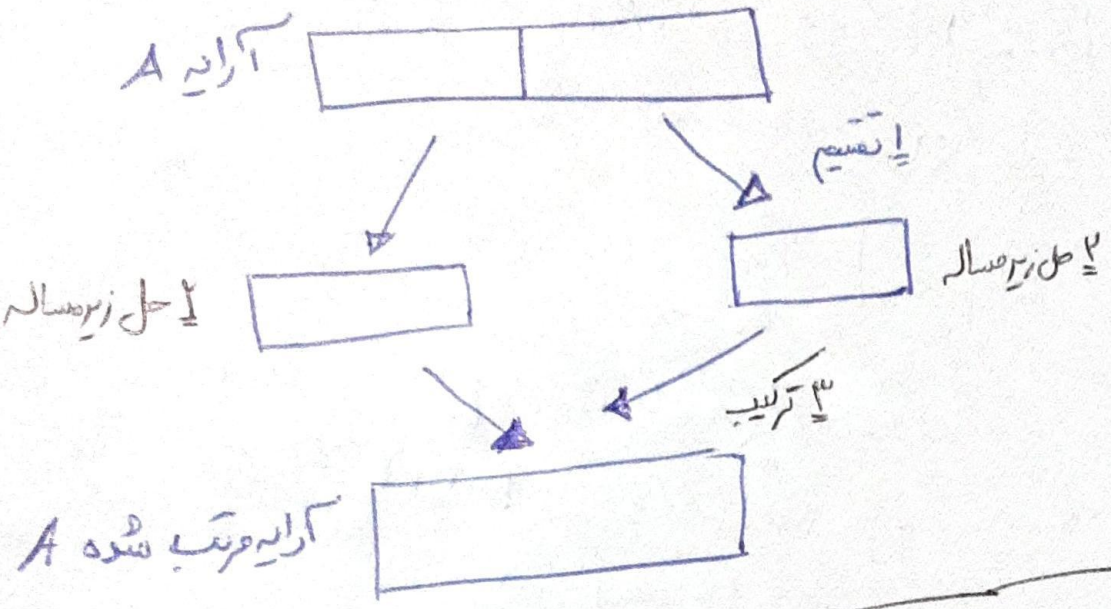
الگوریتمی است که از روش کلی تقسیم و حل برای مرتب سازی استفاده می کند.

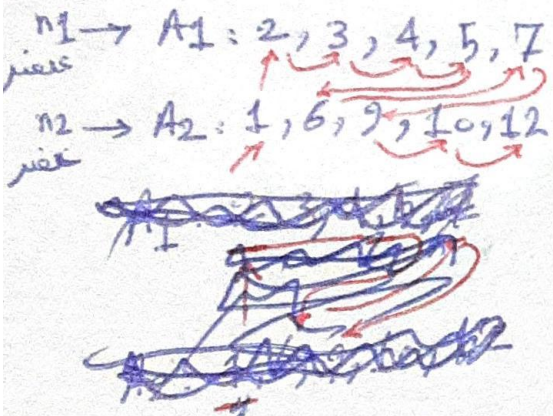
در ادامه الگوریتم مرتب سازی ادغامی با یک مثال تشریح می شود.

آرایه A: 5, 2, 3, 7, 4, 9, 12, 1, 6, 10



روال کلی در الگوریتم مرتب سازی ادغامی :





شهودی روی ادغام دو آرایه مرتب شده :

مثال :

$A_{12} = 1, 2, 3, 4, 5, 6, 7, 9, 10, 12$

تعداد مقایسه ها برای ادغام ←
- حداقل برابر n_1 زمانیکه تمام عناصر A_1 کوچکتر از A_2 باشد ($n_1 \leq n_2$ با توجه به فرض کردن)
- حداکثر $n_1 + n_2 - 1$ زمانیکه عناصر A_1 و A_2 یکی در میان از هم کوچکتر و بزرگتر باشند.

نکته ۱ : دو آرایه مرتب، در زمان خطی مرتب می شوند، هزینه ادغام $O(n_1+n_2)$

نکته ۲ : برای ادغام دو زیر آرایه A_1 و A_2 از آرایه A ← آیا نیاز به در نظر گرفتن آرایه اضافی داریم یا خیر؟

تمرین ↑

تلاش ها برای انجام این عمل (In Place Merge-Sort) منجر به عموماً

از این به بعد در ادامه این الگوریتم می شود.