فصل ۶ ـ پریا مبلغی

۱) مرتب سازی درجی ، چون‌که این الگوریتم زمانی که بخشی از آرایه مرتب است بهتر عمل می‌کند

۳) چون‌که بیشتر آرایه مرتب است بهتر است از درجی استفاده کنیم

$O(n + n\log n)$ (زیر $n\log n$ علامت کشیده شده و بالای آن نوشته شده: $n\log n$)

۴) با پیچیدگی nlogn آن را مرتب می‌کنیم ، سپس با پیچیدگی n ، تعداد تکرار را به دست می‌آوریم
یا تعداد تکرار هر عدد را به عنوان index یک آرایه در نظر گرفته و ذخیره می‌کنیم

۵) الف) دو عدد اول و آخر را جمع می‌کنیم ، اگر بیشتر بود از سمت راست (از انتها) یکی به چپ می‌رویم و
اگر کمتر بود از سمت چپ (از ابتدا) یکی به راست می‌رویم

ب) با nlogn مرتب می‌کنیم و الگوریتم (الف) را اجرا می‌کنیم

۶) ۱) i=0 ۲) مرتب سازی آرایه یک عددی a[i] به روش درجی ۳) i++ ۴) اگر i < a.length برو به ۲

۷) مقایسه و sort حبابی ـ برگرداندن عنصر بزرگ‌تر به آخر

⑧

```
IDk ( int [] a) {
    stack s = new stack ();
    s.push (a.length);
    s.push (0);
    while ( ! s.isEmpty ()) {
        int start = s.pop ();
        int end = s.pop ();
        int b = place ( start, end, a);
        if (b > start + 1) {
            s.push (b - 1);
            s.push (start); }
        if (b < end - 1) {
            s.push (end);
            s.push (b + 1); }}}
```

⑨ اگر $n = N - k$ آنگاه خواهیم داشت

Sort (A, 1, N)

Sort (a, K+1, N)

Sort (A, 1, k+1)

⑩ اگر $k > \log n$، آرایه را مرتب می‌کنیم، و k عنصر ابتدای را برمی‌داریم $O(n \log n)$

اگر $k < \log n$، k مرحله‌ی اول مرتب‌سازی حبابی را انجام می‌دهیم $O(nk)$