АНАЛИЗ НА ЗАДАЧА
ПАЛАЧИНКОВО СОРТИРАНЕ

Основната идея за решаване на задачата е базирана на наблюдението, че с най-много две операции можем да преместим най-голямото число от редицата на първо място и след това, обръщайки цялата пермутация – на последно, където всъщност трябва да бъде. Прилагаме този метод за останалата част от редицата, докато не сортираме цялата пермутация.

За наивна имплементация на това решение са предвидени 40 точки. Авторовото решение използва структурата от данни Implicit Treap, която позволява изпълнение на заявки за намиране на число и за обръщане на интервал за логаритмична сложност.

В този анализ ще отбележим някои важни моменти в реализацията на решението. Implicit Treap (Декартово дърво с неявни ключове) е двоично дърво за търсене, в което всеки връх има ключ, отговарящ на номера му в обхождането ляво-корен-дясно, и произволно избран приоритет. Приоритетът на даден връх е по-голям от приоритетите на децата му, а произволният му избор има за цел да балансира дървото. Основните операции, с които оперира Treap-ът, са Split и Merge – съответно разделят и сливат структурата или части от нея. Това предизвиква промяна в ключовете на върховете, ето защо те се преизчисляват в процеса на изпълнение на заявките.

Заявката за обръщане на интервал се осъществява с Lazy Propagation. Първо трябва с операцията Split да отделим само частта от Treap-а, която искаме да обърнем. След това е достатъчно да отбележим, че коренът и неговото поддърво трябва да бъдат обърнати. Lazy-стойността се пренася към следващото ниво на дървото като разменим указателите към двете деца и променим техните Lazy-стойности.

За заявката за намиране на число е необходимо да имаме указател към върха, който отговаря за него. С обхождане по върховете от него до корена, съблюдавайки кои от тях са отбелязани за обръщане, може да намерим позицията му в дървото.