

Cars

(Решение)

Като последна задача в темата беше скрита една от лесните задачи. Тя е по-скоро аналитична, отколкото кодерска - идеята ѝ отнема много повече време от имплементацията (подобно на `KthElement`).

Първото нещо, което трябва да видим, е какво точно се иска от нас. Дадена е пермутация на числата от 0 до N , искаме да направим пермутацията $(0, 1, \dots, N)$, като можем да разменяме местата на нулата и произволно друго число.

Трябва да забележим, че ако едно число (с изключение на нулата) е на мястото си, то можем да не го мърдаме повече оттам. Съответно, ако всички числа са си на мястото сме готови. Ако ли пък не, то има поне две, които не са си на местата (забележете, че няма как само едно да не си е на мястото).

Всяко число, което не си е на мястото, ще иска да "отиде" на точно определена позиция. Примерно в $(2, 1, 0)$ първото число "иска" да отиде на мястото на третото, и третото иска да отиде на мястото на първото. Второто число вече си е на мястото и не иска да ходи никъде - съответно не го мърдаме. Като друг пример нека разгледаме $(1, 2, 0, 4, 3)$. Тук вече имаме зависимост на повече от две числа - 1 иска да отиде на мястото на 2, 2 иска да отиде на мястото на 0 и 0 иска да отиде на мястото на 1. Забележете, че тръгвайки от едно число (което не си е на мястото) ще се образува циклична зависимост между две или повече числа. Тъй като входът е пермутация на числата от 0 до N , то тази зависимост ще е чист цикъл - тоест няма да има "израстъци" по него, тъй като всяко число се среща точно по веднъж.

Като цяло, всяка пермутация може да бъде разбита на независими цикли. Оправяйки един цикъл, ние нито разбъркваме останалите такива, нито числа, които вече са си на мястото. Това е и оптималният ред, в който да местим колите - цикъл по цикъл.

За да оправим един цикъл ще са ни нужни приблизително толкова мърдания, колкото коли има в него, евентуално плюс едно. Трите случая са:

1. Ако нулата участва в цикъла, то просто местим колите една по една, местейки и нулата. В края всяка кола ще си е на мястото, както и нулата ще си е на мястото. За цикъл от този тип ни трябват точно толкова местения, колкото коли има (тоест броят на елементите в цикъла минус едно).

2. Ако нулата не участва в цикъла, то нямаме свободно място и трябва да мръднем една от колите на мястото на нулата (независимо къде се намира тя в момента). След това местим всички останали коли в цикъла. Накрая връщаме колата, която сме преместили в началото, на вече освободеното ѝ крайно място. Така нулата се връща обратно там, където е била в началото. За цикъл от този тип ни трябва 1 месетене в началото + местене на всички коли без тази, която сме преместили в началото + още едно местене на първата. Тоест общо дължината на цикъла + 1.
3. Коли, които вече са си на мястото образуват цикъл с дължина 1, но тях не трябва да броим, тъй като изобщо не се налага да ги местим.

Ии това е. Просто трябва да преброим циклите в пермутацията с дължина по-голяма от 1, да проверим, дали нула присъства във всеки от тях и да прибавим съответния брой операции, които са ни нужни, за да фикснем всеки от тях. Сложността на този алгоритъм е $O(N)$, където N е броят коли.