Анализ на решението

Подзадача 1

При малките ограничения е достатъчно да симулираме всяка операция директно и след всяка да преброим инверсиите с двоен цикъл. Това дава решение с квадратична проверка на инверсиите за всяка стъпка.
Сложност $O(m \cdot n^2)$

Подзадача 2

Тук е достатъчно да преброим инверсиите само веднъж в началото. След това симулираме операциите, като знаем, че размяната на два съседни елемента променя броя на инверсиите с точно 1.
Сложност $O(n^2 + m \cdot n)$

Подзадача 3

Тук имаме само една операция, но n вече е голямо, затова броенето на инверсиите с $O(n^2)$ е бавно. Вместо това намираме броя на инверсиите с Фенуик дърво или интервално дърво за $O(n \log n)$, като обхождаме елементите отляво надясно и поддържаме колко по-големи елемента са видяни до момента.
Сложност $O(n \log n)$

Подзадача 4

Операциите са последователни: $2, 3, 4, \ldots, n$. Разглеждаме елемент на позиция i и нека пред него има k по-големи числа. Този елемент ще участва в k последователни размени като по-малкото число и при всяка от тях ще намалява броя на инверсиите с 1. Това се случва на операции $i-1, i, \ldots, i+k-2$.

Идеята е да направим масив от разлики: на позиция $i-1$ добавяме $+1$ (започва да намалява), а на позиция $i+k-1$ добавяме -1 (спира да намалява). След като натрупаме всички такива интервали, пресмятаме префиксни суми и получаваме с колко намаляваме броя на инверсиите на всяка стъпка. Така получаваме обща сложност $O(n \log n + n)$.

Подзадача 5

Операциите вече не са последователни, но идеята остава да броим всяка инверсия от страната на по-малкото число. Поддържаме приоритетна опашка с елементите от текущия префикс и броя инверсии, в които участват като по-малки. При нова операция разширяваме префикса, като добавяме новите елементи и техните инверсии.

След това премахваме от опашката всички елементи с нулев брой инверсии. Броят на инверсиите намалява с броя елементи в опашката, а броят инверсии на всеки от тях намалява с 1. Директното намаляване за всички

елементи е скъпо, затова поддържаме в опашката ключ инверсии+позиция, което гарантира, че на върха стои елементът, който първи ще стане нула и ще бъде изваден. Така получаваме сложност $O(n \log n + m \log n)$.