1. Алгоритмы 1 и 3 ничего не вернут, если элемент, встречающийся чаще, чем n / 2 раз отсутствует. Пример: 1 2 1 2 1 2

1 алгоритм вообще ничего не вернет, 2 алгоритм элемент «2» (с = 1, ind = 4), 3 алгоритм тоже ничего не вернет, а просто отсортирует и получим 1 1 1 2 2 2.

Но если будет элемент, встречающийся чаще чем n / 2 раз, то все алгоритмы вернут его!

2. В 1-ом есть вложенный цикл, как и внутренний совершающий n итераций => O(n^2)

Во 2-ом один цикл длины n => O(n)

В 3-ем есть некая сортировка и цикл до n (O(n)), но так как вряд-ли есть сортировка за O(n), то сложность этого алгоритма будет совпадать со сложностью сортировки.

3. В первом цикле стоит просто вывести сообщение, если нет такого c > n / 2

else return «no such element»

Во втором следует ещё раз пройти по циклу, чтобы убедиться, что мы нашли именно преобладающий элемент (завести счётчик и проверить сколько раз мы его встретили и что это правда > n / 2)

В третьем нужно так же добавить сообщение, если такого элемента нет и поэтому блок с вызовом return надо вынести за цикл.

4. К 1-му добавили ещё один вызов return, вне циклов — O(1). Прежняя скорость сохранилась

Во 2-ом получили O(n) + O(n) = O(n)

В 3-ем аналогично первому и оценка не ухудшилась.