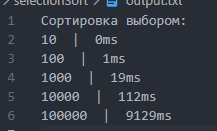
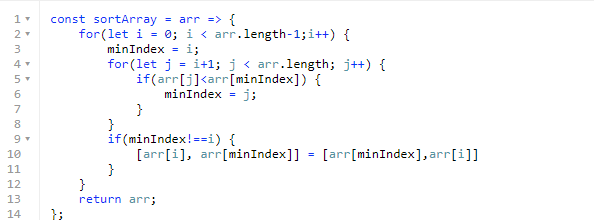
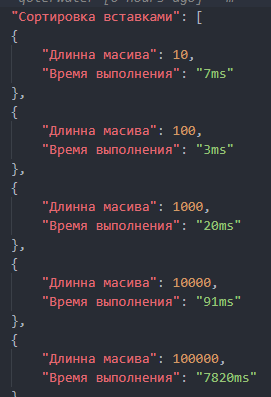
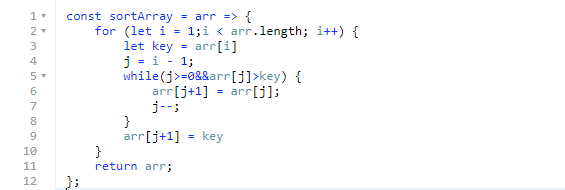
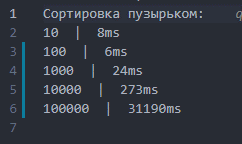
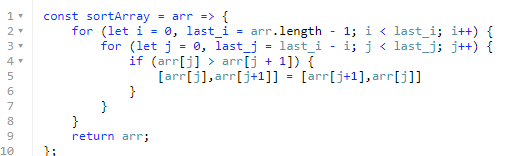
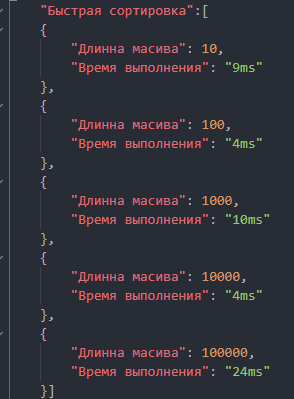
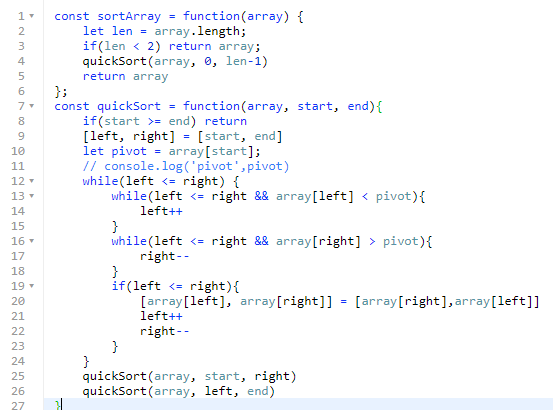
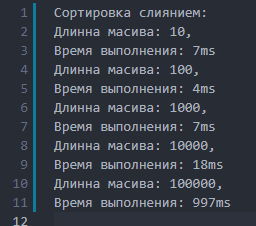
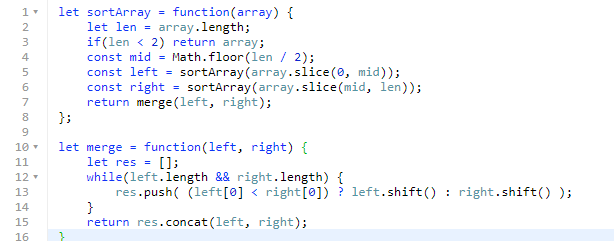
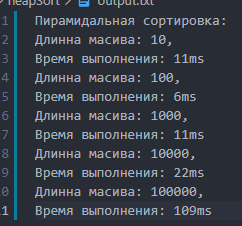
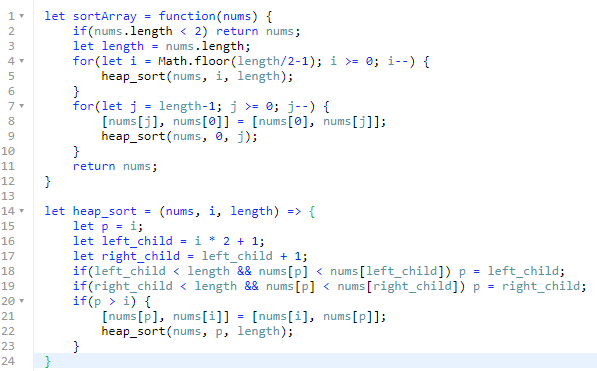
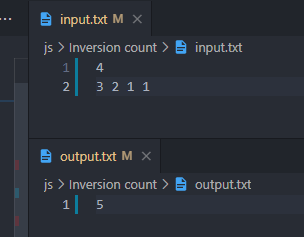
1. Сортировка выбором — на каждом шаге мы добавляем минимальный элемент в левую часть массива. Находим минимальный элемент в неотсортированной части и меняется местами с первым элементов в неотсортированной части. И так продолжается, пока не закончится корневой цикл
2. Сортировка вставками – в начале цикла устанавливается ведущий элемент, сохраняем его в отдельную переменную *key*, дальше алгоритм находит куда должен встать текущий элемент массива в его отсортированную часть. После окончания внутреннего цикла переменная *key* перемещается на освободившееся место
3. Сортировка пузырьком – Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется перестановка элементов. Проходы по массиву повторяются {\displaystyle N-1}*N - 1* раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован. 
4. Быстрая сортировка – одна из самых быстрых сортировок, она является улучшенным вариантом сортировки пузырьком. Алгоритм сортировки:
   1. Выбрать опорный элемент.
   2. Отсортировать все остальные элементы с опорным, все, что меньше – слева, а что больше и равные – справа.
   3. Запустить функцию сортировки отдельно для списка который получился слева от опорного и для списка, который получился справа от опорного.



1. Сортировка слиянием – сперва массив делится на две равные части, запуская эту функцию рекурсивно, пока в каждой части не останется по 2 элемента. Затем алгоритм переходит к функции слияния, которая сравнивает первый элемент из левой половины и первый элемент из правой половины. Если элемент из левой половины меньше, то он добавляется в результирующий массив, иначе добавляется правый элемент. Функция слияния запускается ко всем оставшимся долям массива, в конечном итоге сливая их в один массив.
2. Пирамидальная сортировка – сперва мы преобразовываем список в бинарное дерево Max Heap, в нём наибольший элемент будет на вершине дерева. Затем этот элемент меняется местами с последним элементом в списке, после чего дерево перестраивается, в нем на вершине будет уже следующий максимальный элемент, который ставится в предпоследнее место списка, и так далее до момента, пока все элементы дерева не будут уделаны. 
3. Количество инверсий - Это измененная сортировка слиянием, все изменения скрыты внутри слияния. При сортировке алгоритм удаляет инверсии. При слиянии алгоритм подсчитывает количество удаленных инверсий (можно сказать, отсортированных). Единственный момент, когда инверсии удаляются, - это когда алгоритм берет элемент из правой части массива и объединяет его с основным массивом. Число инверсий, удаленных этой операцией, равно числу элементов, оставшихся от левого массива, подлежащего объединению. (<https://contest.yandex.ru/contest/26597/run-report/51782844/>) 
4. К-ая порядковая статистика – K-ой порядковой статистикой N-элементной последовательности AN называется число AK, которое будет стоять на K-ом месте после упорядочивания элементов этой последовательности по возрастанию. Последовательность AN задаётся следующим образом:  
   A1 = P,  
   Ai = (Ai-1 ⋅ Q) mod V.

Массив сортируется быстрой сортировкой после генерации, затем в выходной файл записывается AK

(<https://contest.yandex.ru/contest/26597/run-report/51790262/>)

