Вывод по таблицам и графикам сравнения разных алгоритмов сортировки

Метод	Преимущества	Недостатки
selection	Время сортировки практически никак не зависит от величины и расположения элементов массива Быстрее bubble на случайных массивах	Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время
bubble	Время сортировки в практически никак не зависит от элементов расположения элементов массива	Медленнее на почти отсортированных массивах, чем bubble_1 и bubble_2 Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время большинстве случаев
bubble_1	Все преимущества bubble, но также низкое время работы при сортировке практически отсортированного массива	Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время
bubble_2	Все преимущества bubble_1 но также ускорение работы в случаях массива, не близкому к обратно-отсортированному	Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время
simple_insertion	Быстрее, чем все вышеперечисленные сортировки в большинстве случаев Достаточно быстрая на практически отсортированных массивах	Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время по сравнению с сортировками за логарифмическое
binary_insertion	Быстрее линейной при больших размерах массива. Достаточно быстрая на практически отсортированных массивах	Медленнее на массивах с произвольным расположением элементов, чем сортировки не за квадратичное время по сравнению с сортировками за логарифмическое. Медленнее линейной при маленьких размерах массивах
counting	Скорость сортировки не зависит от расположения чисел, а только от их величины Зависит линейно от размера Очень быстрая при маленьких величинах чисел (в работе максимальный размер числа равен 4000, так что и на больших числах сортировка очень быстрая)	Использование большого количества доп. памяти Медленная на очень больших числах

radix	Практически все преимущества counting, а также быстрее её на больших числах	Использование большого количества доп. памяти Является более медленной на маленьких числах, чем counting Медленная на числах в несколько крат превышающих размер массива
merge	Не зависит от расположения элементов в массиве, является достаточно быстрой при большом размере массива	Медленная при маленьком размере массива, а также проигрывает сортировкам, которые являются быстрыми на практически отсортированном массиве.
quick	Является достаточно быстрой при маленьком размере массива Быстрая на практически отсортированном массиве	Медленная на массивах с большим количеством элементов
heap	Показывает одни из самых лучших результатов на всех размерах массива	Медленнее shell_ciura во всех случаях и shell_shell (на маленьких размерах массива), но использует доп память, а также медленнее цифровой и подсчетом на маленьких числах
shell_shell	Самая быстрая квадратичная сортировка на произвольном расположении элементов в массиве По скорости не уступает логарифмическим сортировкам (на взятых размерах массива)	Медленнее shell_ciura
shell_ciura	Показывает очень высокие показатели при сортировке всех видов массива	Неприменима для больших размеров массива Медленнее, чем counting и radix при маленьких числах

Графики измерения времени и элементарных операций отличаются по нескольким причинам:

- устройство, на котором происходит компиляция, может быть загружено другими процессами
- разные элементарные операции занимают разное время, что также влияет на результат тестирования
- могут возникать случайные выбросы, которые искажают результат тестирования

Тестирование лучше проводить на практически незагруженном другими процессами устройстве, а также проводить сортировку несколько раз и усреднять полученный результат