Introsort   
(Интроспективная сортировка)

1. Introsort или интроспективная сортировка — [алгоритм сортировки](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B8), предложенный [Дэвидом Мюссером](https://ru.wikipedia.org/w/index.php?title=%D0%94%D1%8D%D0%B2%D0%B8%D0%B4_%D0%A0._%D0%9C%D1%8E%D1%81%D1%81%D0%B5%D1%80&action=edit&redlink=1) в 1997 году. Он использует [быструю сортировку](https://ru.wikipedia.org/wiki/%D0%91%D1%8B%D1%81%D1%82%D1%80%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) и переключается на [пирамидальную сортировку](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%80%D0%B0%D0%BC%D0%B8%D0%B4%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), когда глубина [рекурсии](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BA%D1%83%D1%80%D1%81%D0%B8%D1%8F) превысит некоторый заранее установленный уровень (например, [логарифм](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B0%D1%80%D0%B8%D1%84%D0%BC) от числа сортируемых элементов). Этот подход сочетает в себе достоинства обоих методов с худшим случаем [O](https://ru.wikipedia.org/wiki/%D0%9E-%D0%BD%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D1%8F)(*n* log *n*) и быстродействием, сравнимым с быстрой сортировкой. Так как оба алгоритма используют сравнения, этот алгоритм также принадлежит классу [сортировок на основе сравнений](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B8_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B5_%D1%81%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B9).
2. Алгоритм

Интросортировка начинается с быстрой сортировки, и если глубина рекурсии превышает определенный предел, она переключается на Heapsort, чтобы избежать временной сложности быстрой сортировки O (N 2 ) в худшем случае. Он также использует сортировку вставкой, когда количество сортируемых элементов значительно меньше.

Итак, сначала он создает раздел. Отсюда три случая.

Если размер раздела таков, что существует возможность превышения предела максимальной глубины, то Introsort переключается на Heapsort. Мы определяем максимальный предел глубины как 2 \* log (N)

Если размер раздела слишком мал, тогда быстрая сортировка затухает до сортировки вставкой. Мы определяем это ограничение как 16 (из-за исследований). Так что, если размер раздела меньше 16, мы сделаем сортировку вставкой.

Если глубина рекурсии не достигла предела и подмассив не совсем мал (то есть между 16 и 2 \* log (N)), тогда он выполняет простую быструю сортировку.

1. Сложность по времени

Лучший случай — O (N log N)  
Средний случай-O (N log N)  
В худшем случае — O (N log N)

Так как Quicksort может иметь худшую временную сложность O (N 2 ) и это также увеличивает пространство стека рекурсии (O (log N), если применяется хвостовая рекурсия), поэтому, чтобы избежать всего этого, нам нужно переключить алгоритм с Quicksort на другой если есть вероятность худшего случая. Таким образом, Introsort решает эту проблему, переключаясь на Heapsort.

Также из-за большего постоянного фактора быстрая сортировка может работать даже хуже, чем алгоритм сортировки O (N2), когда N достаточно мало. Таким образом, Introsort переключается на сортировку вставкой, чтобы уменьшить время выполнения сортировки.

1. Таблица количества входных данных и количества итераций

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количество элементов | Количество итераций | Количество элементов | Количество итераций | Количество элементов | Количество элементов |
| 200 | 1775 | 3600 | 649820 | 7000 | 1735295 |
| 400 | 5827 | 3800 | 725736 | 7200 | 1836541 |
| 600 | 12300 | 4000 | 804707 | 7400 | 1943976 |
| 800 | 21724 | 4200 | 890454 | 7600 | 2052990 |
| 1000 | 33947 | 4400 | 979535 | 7800 | 2176083 |
| 1200 | 48311 | 4600 | 1073440 | 8000 | 2291379 |
| 1400 | 64533 | 4800 | 1173315 | 8200 | 2413697 |
| 1600 | 84243 | 5000 | 1275496 | 8400 | 2539243 |
| 1800 | 107136 | 5200 | 1382663 | 8600 | 2664062 |
| 2000 | 132057 | 5400 | 1492716 | 8800 | 2793666 |
| 2200 | 162603 | 5600 | 1609565 | 9000 | 2929566 |
| 2400 | 195103 | 5800 | 1734268 | 9200 | 3066755 |
| 2600 | 227872 | 6000 | 1860675 | 9400 | 3207463 |
| 2800 | 266673 | 6200 | 1993299 | 9600 | 3343459 |
| 3000 | 306357 | 6400 | 2129616 | 9800 | 3486843 |
| 3200 | 348539 | 6600 | 2270617 | 10000 | 3634052 |
| 3400 | 392828 | 6800 | 2416740 |  |  |

Таблица количества входных данных и затраченного времени

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количесво Элементов | Времени затрачено | Количесво Элементов | Времени затрачено | Количесво Элементов | Времени затрачено |
| 200 | 1452000 | 3600 | 379800 | 7000 | 598200 |
| 400 | 227700 | 3800 | 412200 | 7200 | 575000 |
| 600 | 375100 | 4000 | 519200 | 7400 | 607200 |
| 800 | 365600 | 4200 | 452100 | 7600 | 804300 |
| 1000 | 819200 | 4400 | 460600 | 7800 | 864700 |
| 1200 | 472600 | 4600 | 503000 | 8000 | 691200 |
| 1400 | 237900 | 4800 | 418200 | 8200 | 661900 |
| 1600 | 200100 | 5000 | 536400 | 8400 | 938300 |
| 1800 | 236400 | 5200 | 428500 | 8600 | 906000 |
| 2000 | 258400 | 5400 | 439700 | 8800 | 951200 |
| 2200 | 295600 | 5600 | 989800 | 9000 | 940500 |
| 2400 | 475100 | 5800 | 588500 | 9200 | 1004700 |
| 2600 | 517200 | 6000 | 619000 | 9400 | 1069500 |
| 2800 | 723000 | 6200 | 639300 | 9600 | 1068300 |
| 3000 | 425100 | 6400 | 517500 | 9800 | 862500 |
| 3200 | 500400 | 6600 | 547600 | 10000 | 821200 |
| 3400 | 493700 | 6800 | 616600 |

1. График зависимости кол-ва элементов и кол-ва итераций

График зависимости кол-ва элементов и времени работы программы