# Выводы по Java – коллекциям

Общее количество элементов равно 1 000 000.

Время выполнения основных операций

|  |  |  |  |
| --- | --- | --- | --- |
|  | Добавление в середину, нс | Добавление вконец, нс | Удаление из середины, нс |
| ArrayList | 5 519 169 | 1 538 | 5 991 841 |
| LinkedList | 5 190 233 | 1 964 | 2 901 885 |
| HashSet | 2 522 | 3 351 | 7 479 |
| LinkedHashSet | 1 142 | 4 289 | 6 736 |
| TreeSet | 1 486 | 5 807 | 73 951 |
| HashMap | 1 147 | 3 150 | 5 275 |
| LinkedHashMap | 1 884 | 2 381 | 4 032 |
| TreeMap | 2 184 | 7 183 | 9 175 |

1. Вставка и удаление в середину коллекции происходит быстрее в коллекции LinkedList, чем в ArrayList, так как в данном случае в ArrayList происходит перезапись всех данных «правее» добавляемого. Кроме того, при удалении элементов, размер массива явно не уменьшается. Поэтому LinkedList выгоднее использовать, когда необходимы активные внесения данных в середину списка.
2. В Set коллекциях нет повторяющихся элементов. В LinkedHashSet элементы расположены в порядке добавления. В TreeSet коллекции элементы упорядочены. Наименьшее время добавления и удаления из середины множества – у LinkedHashSet. Наибольшее время удаления из середины множества - у TreeSet. TreeSet лучше использовать в случаях, когда необходимо отсортированное множество, а LinkedHashSet – когда часто добавляем элементы.
3. В Map коллекции хранится два объекта: ключ и значение. Ключи – уникальные объекты. Самая затратная по времени реализация – это TreeMap. Зато TreeMap хранит значения ключей в отсортированном порядке, а LinkedHashMap хранит ключи в порядке вставки. Если необходима постоянная вставка элементов, то лучше для этого подойдет LinkedHashMap.