*Выводы на основании полученных результатов во втором задании(Test Performance of Collections):*

**a.** В результате сравнения основных операций(добавление, вставка, удаление элементов) для **ArrayList и LinkedList**:

(100000 элементов), time = [ns]

16359 ArrayList add time

20421 LinkedList add time

95197 ArrayList insert time

28370 LinkedList insert time

62483 ArrayList remove time

24369 LinkedList remove time

Вывод: Вставка в конец осуществляется в целом примерно одинаково. Для задачи с операциями динамического удаления или вставки в центр коллекции лучше выбирать LinkedList.

**b.** В результате сравнения основных операций(добавление, вставка, удаление элементов) для **HashSet, LinkedHashSet, TreeSet**:

(~125000 элементов), time = [ns]

5672 HashSet add time

4892 LinkedHashSet add time

3476 TreeSet add time

19808 HashSet remove time

6712 LinkedHashSet remove time

37061 TreeSet remove time

Вывод: Вставка в конец осуществляется быстрее в TreeSet, затем LinkedHashMap и медленне HashSet. Удаление быстрее осуществляется в LinkedHashMap, затем идет HashSet и на последнем месте TreeSet.

В целом оптимальным выбором является LinkedHashSet.

**c.** В результате сравнения основных операций(добавление, вставка, удаление элементов) для **HashMap LinkedHashMap, TreeMap**:

(100000 элементов)

29661 HashMap add time

30094 LinkedHashMap add time

31046 TreeMap add time

45721 HashMap insert time

110630 LinkedHashMap insert time

167678 TreeMap insert time

38239 HashMap remove time

36250 LinkedHashMap remove time

94262 TreeMap insert time

Вывод: Вставка в конец осуществляется вц елом одинаково, добавление в середину быстрее для HashMap. Удаление элементов для HashMap и LinkedHashMap почти одинаково. Оптимальным выбором является HashMap.

При переопределении equals и hashcode мы получаем прирост в производительности.