После запуска приложения и создания массива, среднее количество объектов типа int[] примерно равно 1,6 млн единиц.

Сортировку проводим как в задании - последовательно:

1. «слияние»,
2. «вставка»,
3. «пузырёк».

Итак, первым мы протестируем ParallelGC.

Получены следующие результаты:

Начало работы приложения: 19:05:15.472536400;

Первая сортировка интервал: 19:05:37.264577700 - 19:05:37.321571300

Вторая сортировка интервал: 19:05:47.712264200 - 19:05:50.781754

Третья сортировка интервал: 19:05:57.850415500 - 19:07:11.433881900

Итого, выходит:

На сортировку слиянием ушло всего 57 миллисекунд, на сортировку вставкой – чуть больше 3 секунд, тогда как на сортировку пузырьком ушло аж почти 74 секунды. Разница колосальная!

Что мы можем увидеть из графиков?

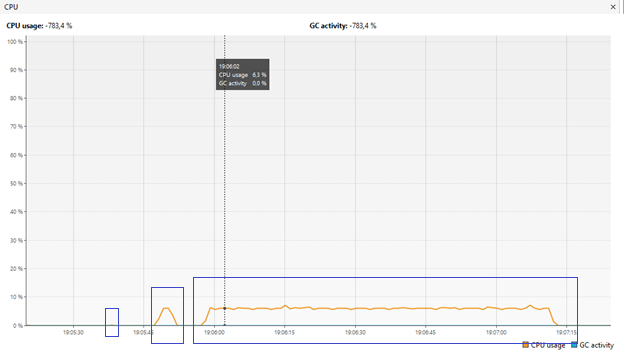
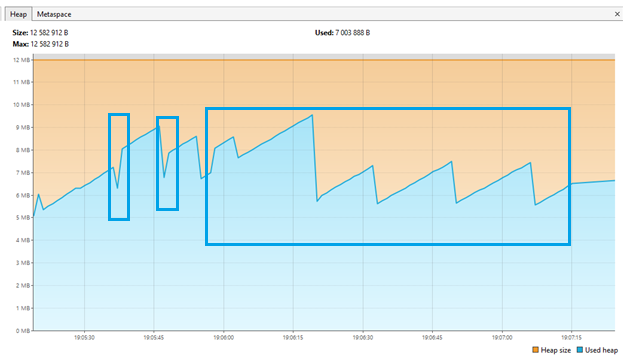
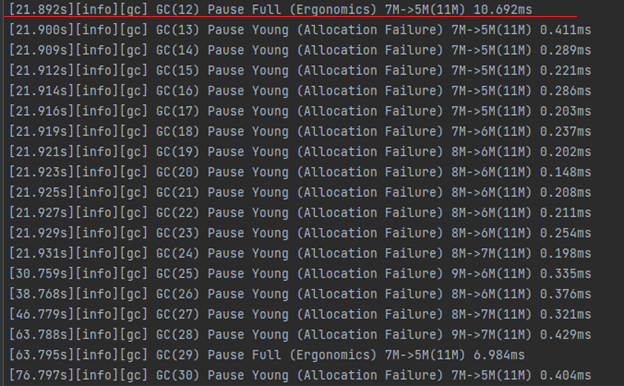


График загрузки CPU показывает, что сортировка слиянием потребовала настолько мало ресурсов, что это почти не видно на графике, сортировка вставкой уже выглядит как что-то адекватное, а вот «пузырёк» выглядит монструозно и занимает так много времени. При этом ни одна из сортировок не задействовала и 10% ресурсов процессора. Ок, посмотрим загрузку памяти.



Использование хипа выглядит интереснее. Мы видим скачки в количестве элементов и последующие сборки мусора (после проведения сортировки) в случае сортировок слиянием и вставкой. Однако, за время сортировки пузырьком, происходит аж пять сборок мусора, одна из которых – полная (резкое падение на графике внутри третьего прямоугольника). Что любопытно, судя по логам, прямо перед первой сортировкой (слиянием) также была полная сборка мусора. Я могу предположить, что в хип не влезла копия нашего сортируемого массива, и JVM приняла решение почистить объекты.



Давайте рассмотрим в том же ключе работу G1GC.

По интервалам:

Начало работы приложения: 19:39:26.427309200;

1. «слияние» - 19:40:05.878947100 - 19:40:05.936937300
2. «вставка» - 19:40:17.993226800 - 19:40:21.144668600
3. «пузырёк» - 19:40:30.295234300 - 19:41:45.260860600

В результате:

На сортировку слиянием ушло 58 миллисекунд, на сортировку вставкой - 3,152 секунды, а на сортировку пузырьком – 75 секунд.

Кажется, смена Garbage Collector’а на G1 не дала в выигрыше скорости сортировок.

Посмотрим графики:

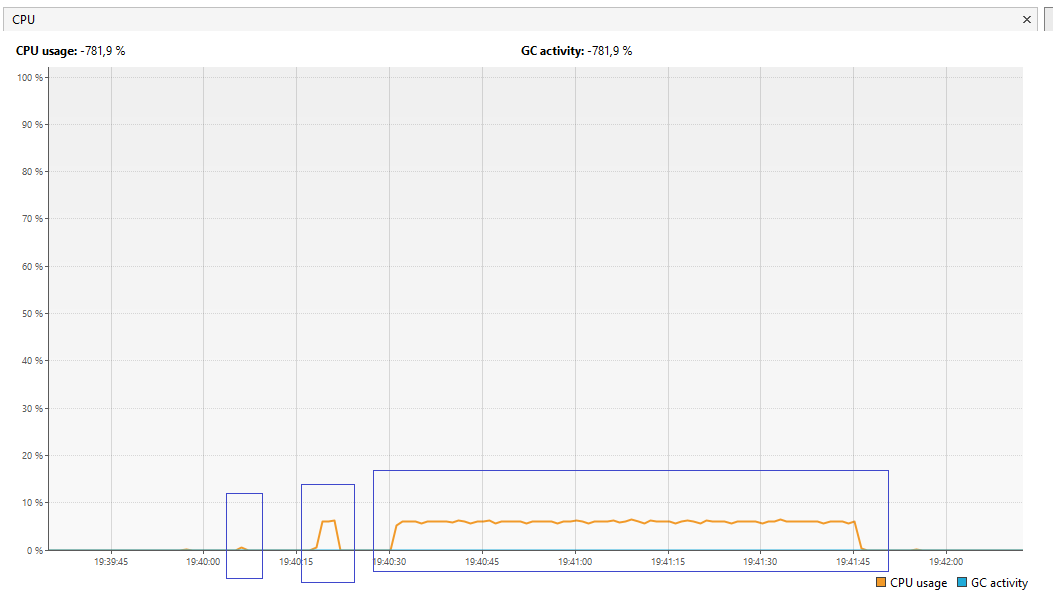
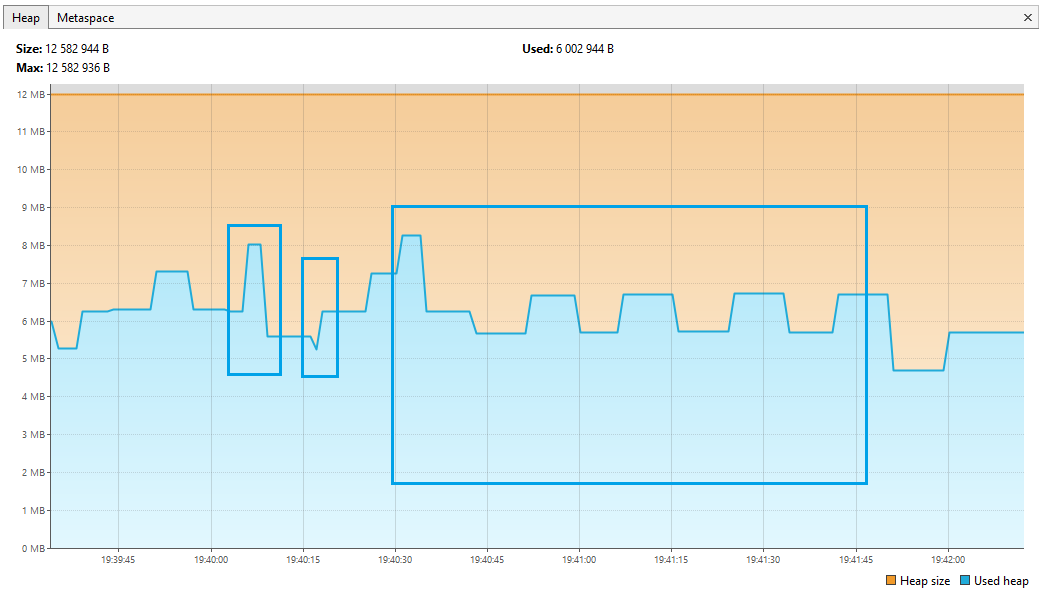
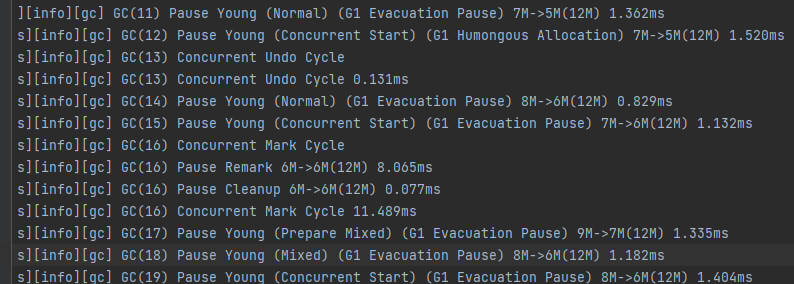


График загрузки CPU очень похож на аналогичный график, который мы получили в прошлый запуск приложения, когда использовали Parallel GC. Не могу сказать ничего нового по этому поводу, поэтому пойдём дальше.



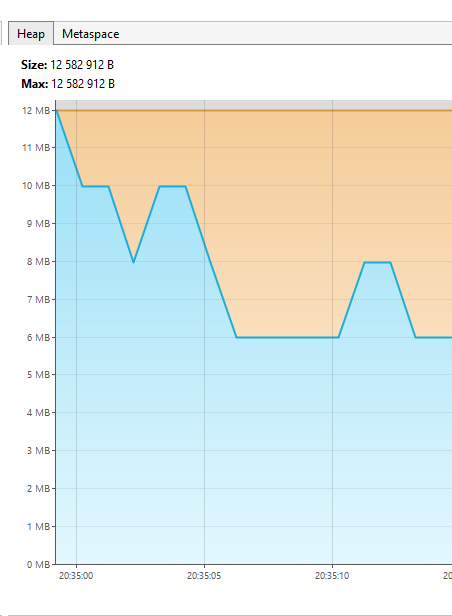
А вот график загрузки хипа выглядит совершенно иначе. Эти плато на вершинах зубцов выглядит любопытно. Давайте заглянем в логи:

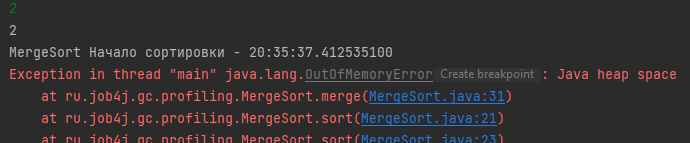


Я могу предположить, что именно так – в виде «плато» на вершинах «зубцов» – выглядит параллельная очистка от мусора, без остановки работы приложения.

Но давайте посмотрим, что нам покажет ZGC.

Ой! Мы сразу сталкиваемся с тем, что при нынешнем хипе память забита сразу со старта, и сразу со старта начинает работать Garbage Collector. И даже попытка простой сортировки слиянием выбивает программу с ошибкой.





Что ж, увеличим хип в два раза и попробуем ещё раз.



Всё получилось! Посмотрим на результаты.

Интервалы:

Начало работы приложения: 20:47:24.539138400;

1. «слияние» - 20:47:49.856146800 - 20:47:49.915404800
2. «вставка» - 20:47:56.986822400 - 20:48:00.117884400
3. «пузырёк» - 20:48:07.542439200 - 20:49:19.306776200

Временные затраты на сортировки плюс/минус те же, что и в прошлый раз, так что сразу перейдём к интересному:

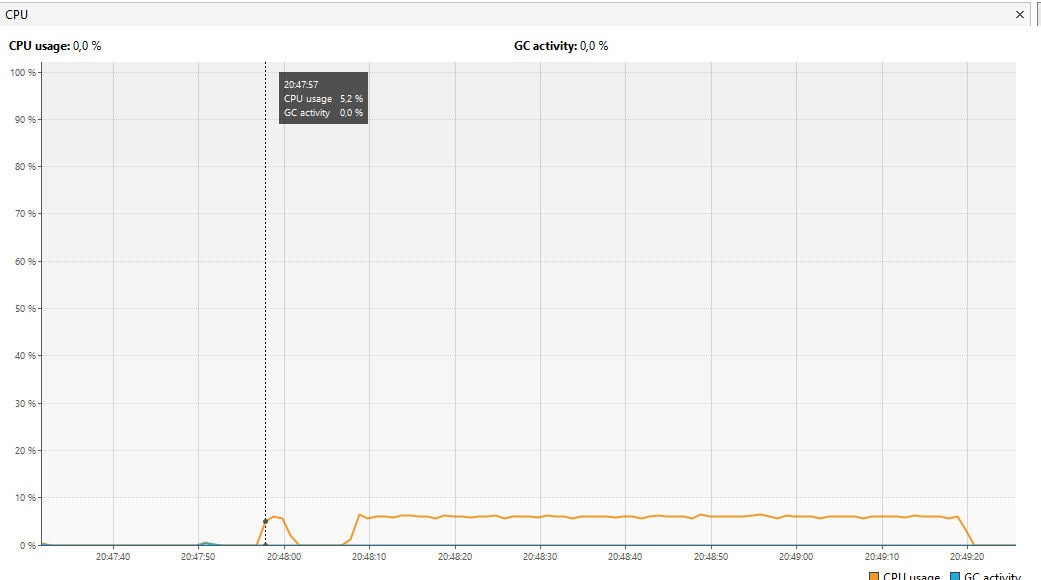
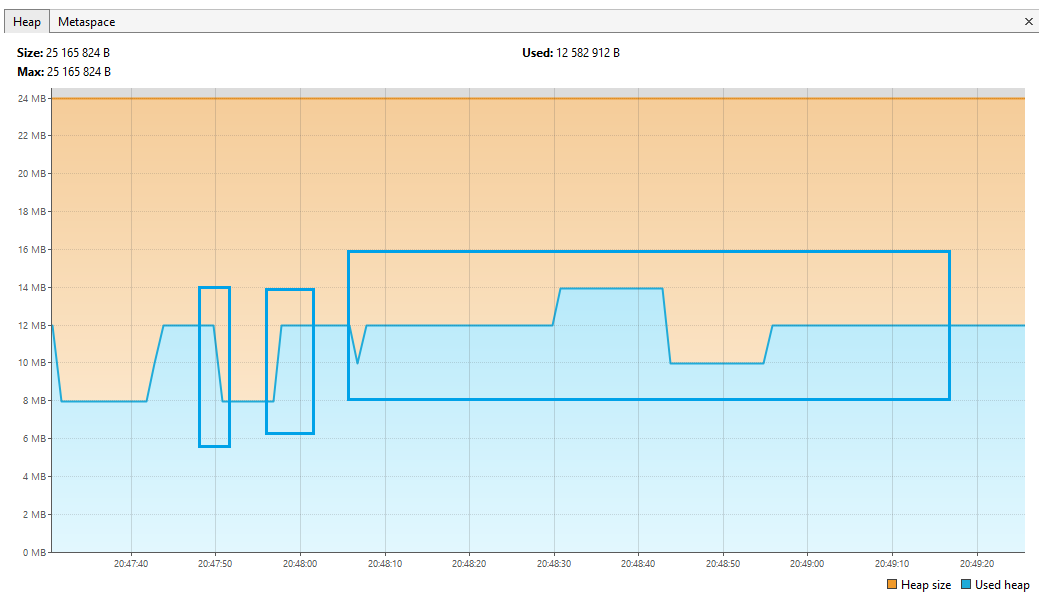


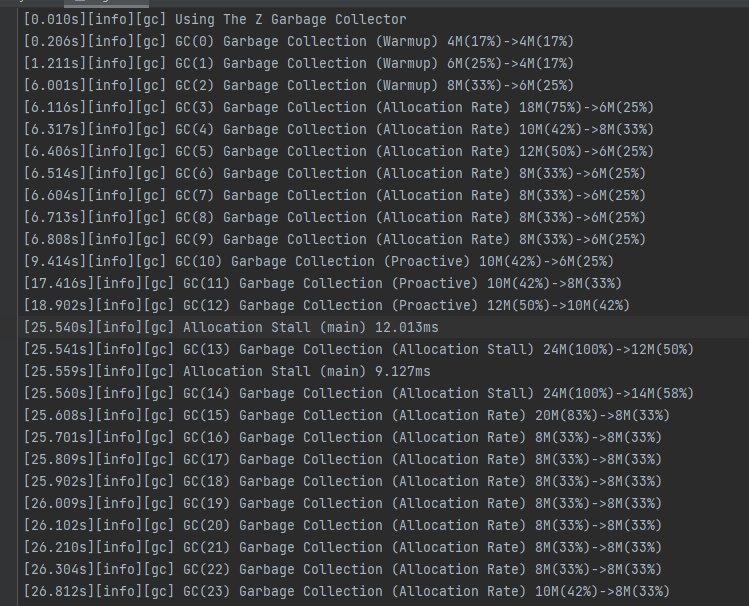
График загрузки процессора почти такой же как раньше, но вот память…



Как вам, а? Да если бы у меня не было чётких таймкодов, я бы и не смог точно сказать в какой момент времени какая сортировка идёт, и идёт ли какая-то вообще!

Поразительно, но ZGC, пусть изначально сжирает больше памяти, при работе приложения почти не показывает никаких скачков и провисаний.

А что у нас в логах?



Разогрев вначале, который выдал нам ошибку в прошлый раз (при малом хипе), но дальше мы не видим существенных остановок программы. Всё гладенько и без резких заполнений и очисток. Поразительно!